Introduction

Amir H. Payberah
amir@sics.se

Amirkabir University of Technology
(Tehran Polytechnic)

Course Information

Course Objective

- Introduction to main concepts and principles of cloud computing and data intensive computing.

Course Objective

- Introduction to main concepts and principles of cloud computing and data intensive computing.

- How to read, review and present a scientific paper.

Topics of Study

- Topics we will cover include:
  - Cloud platforms
  - Cloud storage, NoSQL and NewSQL databases
  - Cloud resource management
  - Batch processing frameworks
  - Stream processing frameworks
  - Graph processing frameworks

Course Material

- Mainly based on research papers.
- You will find all the material on the course web page:
  http://www.sics.se/∼amir/cloud14.htm

Course Examination

- Mid term exam: 20%
- Final exam: 20%
- Reading assignments: 27%
- Final presentation: 23%
- Final project: 10%
- Lab assignments: 0% :)

Reading Assignments

- Nine reading assignments.

- You should write a review for each paper (at most two pages).

Reading Assignments

- Nine reading assignments.
- You should write a review for each paper (at most two pages).
- For each paper you should:
  - Identify and motivate the problem.
  - Pinpoint the main contributions.
  - Identify positive/negative aspects of the solution/paper.

Reading Assignments

- Nine reading assignments.
- You should write a review for each paper (at most two pages).
- For each paper you should:
  - Identify and motivate the problem.
  - Pinpoint the main contributions.
  - Identify positive/negative aspects of the solution/paper.
- For each paper, you might be given some questions to answer.

Reading Assignments

- Nine reading assignments.
- You should write a review for each paper (at most two pages).
- For each paper you should:
  - Identify and motivate the problem.
  - Pinpoint the main contributions.
  - Identify positive/negative aspects of the solution/paper.
- For each paper, you might be given some questions to answer.
- Students will work in groups of two.

Presentation

- Each group give a 30 minutes talk on a scientific paper.
- The list of papers will be available in the course web page.
- You are also free to choose any other paper, but it should be confirmed.

Lab Assignments and the Project

- Implement simple applications on different frameworks through the course.
- The solution of each lab assignment will be uploaded on the course page, one week after their start dates.
- The final project on top of Spark.

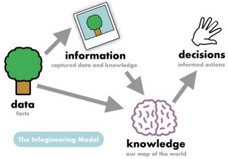
Discussion Forum

- Use the course discussion forum if you have any questions: http://www.sics.se/∼amir/cloud14.htm

Course Overview

Data is not information, information is not knowledge, knowledge is not understanding, understanding is not wisdom.

- Clifford Stoll



Big Data

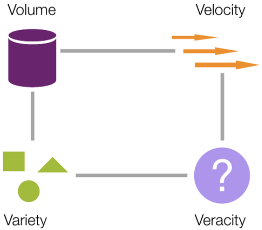
small data

big data

- Big Data refers to datasets and flows large enough that has outpaced our capability to store, process, analyze, and understand.

The Four Dimensions of Big Data

- Volume: data size
- Velocity: data generation rate
- Variety: data heterogeneity
- This 4th V is for Vacillation: Veracity/Variability/Value



Where Does Big Data Come From?

Big Data Market Driving Factors

The number of web pages indexed by Google, which were around one million in 1998, have exceeded one trillion in 2008, and its expansion is accelerated by appearance of the social networks.∗

[∗Wei Fan et al., Mining big data: current status, and forecast to the future, 2013]

Big Data Market Driving Factors

The amount of mobile data traffic is expected to grow to 10.8 Exabyte per month by 2016.*

[*Dan Vesset et al., Worldwide Big Data Technology and Services 2012-2015 Forecast, 2013]

Big Data Market Driving Factors

More than 65 billion devices were connected to the Internet by 2010, and this number will go up to 230 billion by 2020.∗

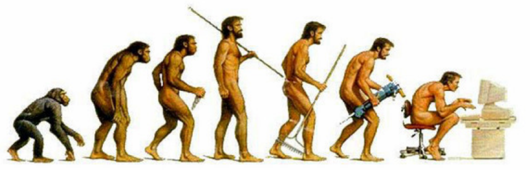
[∗John Mahoney et al., The Internet of Things Is Coming, 2013]

Open source communities

Big Data Market Driving Factors

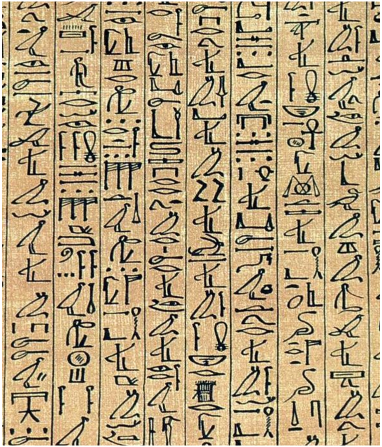
Many companies are moving towards using Cloud services to access Big Data analytical tools.

History of Data



4000 B.C

- Manual recording
- From tablets to papyrus, to parchment, and then to paper

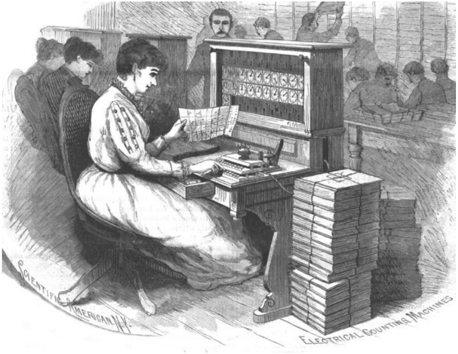


1450

- Gutenberg's printing press

1800's - 1940's

- Punched cards (no fault-tolerance)
- Binary data
- 1890: US census
- 1911: IBM appeared



IBM

1940's - 1950's

- Magnetic tapes

1950's - 1960's

- Large-scale mainframe computers
- Batch transaction processing
- File-oriented record processing model (e.g., COBOL)

1960's - 1970's

- Hierarchical DBMS (one-to-many)
- Network DBMS (many-to-many)
- VM OS by IBM $\rightarrow$ multiple VMs on a single physical node.

1970's - 1980's

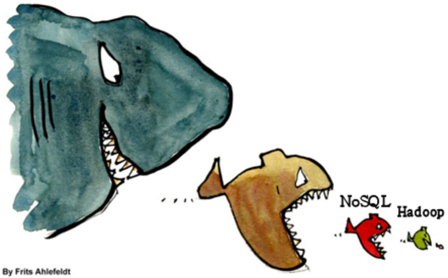
- Relational DBMS (tables) and SQL
- ACID
- Client-server computing
- Parallel processing

1990's - 2000's

- Virtualized Private Network connections (VPN)
- The Internet...

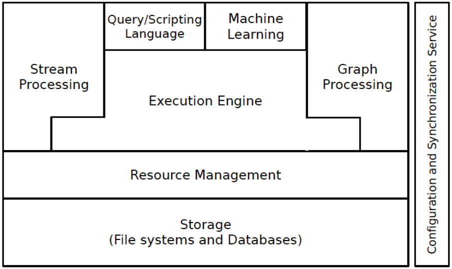
2000's - Now

- Cloud computing
- NoSQL: BASE instead of ACID
- Big Data

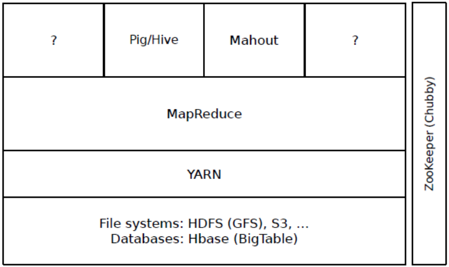


Cloud and Big Data

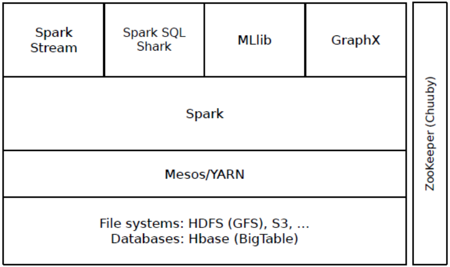
Big Data Analytics Stack



Hadoop Big Data Analytics Stack

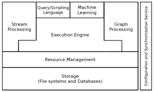


Spark Big Data Analytics Stack



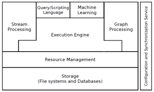
Big Data - File systems

- Traditional file-systems are not well-designed for large-scale data processing systems.



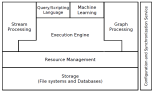
Big Data - File systems

- Traditional file-systems are not well-designed for large-scale data processing systems.
- Efficiency has a higher priority than other features, e.g., directory service.



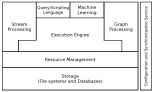
Big Data - File systems

- Traditional file-systems are not well-designed for large-scale data processing systems.
- Efficiency has a higher priority than other features, e.g., directory service.
- Massive size of data tends to store it across multiple machines in a distributed way.



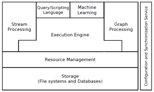
Big Data - File systems

- Traditional file-systems are not well-designed for large-scale data processing systems.
- Efficiency has a higher priority than other features, e.g., directory service.
- Massive size of data tends to store it across multiple machines in a distributed way.
- HDFS/GFS, Amazon S3, ...

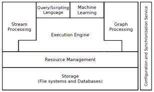


Big Data - Database

- Relational Databases Management Systems (RDMS) were not designed to be distributed.

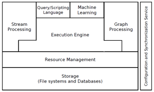


- Relational Databases Management Systems (RDMS) were not designed to be distributed.
- NoSQL databases relax one or more of the ACID properties: BASE



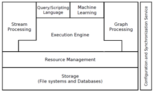
Big Data - Database

- Relational Databases Management Systems (RDMS) were not designed to be distributed.
- NoSQL databases relax one or more of the ACID properties: BASE
- Different data models: key/value, column-family, graph, document.



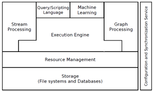
Big Data - Database

- Relational Databases Management Systems (RDMS) were not designed to be distributed.
- NoSQL databases relax one or more of the ACID properties: BASE
- Different data models: key/value, column-family, graph, document.
- Hbase/BigTable, Dynamo, Scalaris, Cassandra, MongoDB, Voldemort, Riak, Neo4J, ...



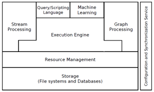
Big Data - Resource Management

- Different frameworks require different computing resources.



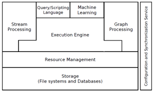
Big Data - Resource Management

- Different frameworks require different computing resources.
- Large organizations need the ability to share data and resources between multiple frameworks.



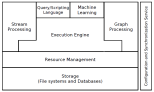
Big Data - Resource Management

- Different frameworks require different computing resources.
- Large organizations need the ability to share data and resources between multiple frameworks.
- Resource management share resources in a cluster between multiple frameworks while providing resource isolation.



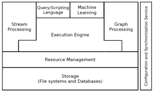
Big Data - Resource Management

- Different frameworks require different computing resources.
- Large organizations need the ability to share data and resources between multiple frameworks.
- Resource management share resources in a cluster between multiple frameworks while providing resource isolation.
- Mesos, YARN, Quincy, ...



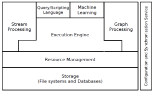
Big Data - Execution Engine

- Scalable and fault tolerance parallel data processing on clusters of unreliable machines.



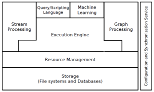
Big Data - Execution Engine

- Scalable and fault tolerance parallel data processing on clusters of unreliable machines.
- Data-parallel programming model for clusters of commodity machines.



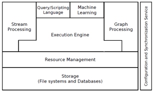
Big Data - Execution Engine

- Scalable and fault tolerance parallel data processing on clusters of unreliable machines.
- Data-parallel programming model for clusters of commodity machines.
- MapReduce, Spark, Stratosphere, Dryad, Hyracks, ...



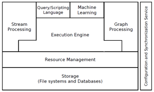
Big Data - Query/Scripting Language

- Low-level programming of execution engines, e.g., MapReduce, is not easy for end users.



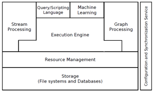
Big Data - Query/Scripting Language

- Low-level programming of execution engines, e.g., MapReduce, is not easy for end users.
- Need high-level language to improve the query capabilities of execution engines.



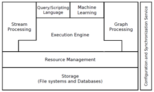
Big Data - Query/Scripting Language

- Low-level programming of execution engines, e.g., MapReduce, is not easy for end users.
- Need high-level language to improve the query capabilities of execution engines.
- It translates user-defined functions to low-level API of the execution engines.



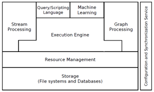
Big Data - Query/Scripting Language

- Low-level programming of execution engines, e.g., MapReduce, is not easy for end users.
- Need high-level language to improve the query capabilities of execution engines.
- It translates user-defined functions to low-level API of the execution engines.
- Pig, Hive, Shark, Meteor, DryadLINQ, SCOPE, ...



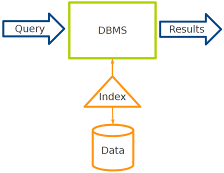
Big Data - Stream Processing

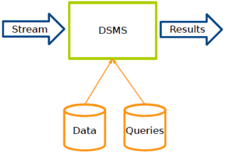
- Providing users with fresh and low latency results.

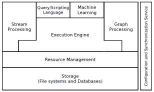


Big Data - Stream Processing

- Providing users with fresh and low latency results.
- Database Management Systems (DBMS) vs. Data Stream Management Systems (DSMS)

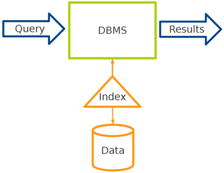


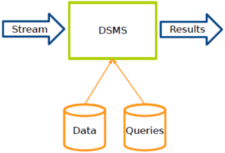




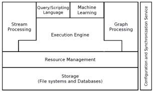
Big Data - Stream Processing

- Providing users with fresh and low latency results.
- Database Management Systems (DBMS) vs. Data Stream Management Systems (DSMS)



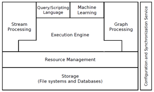


- Storm, S4, SEEP, D-Stream, Naiad, ...



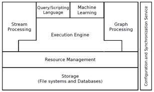
Big Data - Graph Processing

- Many problems are expressed using graphs: sparse computational dependencies, and multiple iterations to converge.



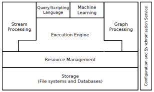
Big Data - Graph Processing

- Many problems are expressed using graphs: sparse computational dependencies, and multiple iterations to converge.
- Data-parallel frameworks, such as MapReduce, are not ideal for these problems: slow



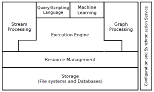
Big Data - Graph Processing

- Many problems are expressed using graphs: sparse computational dependencies, and multiple iterations to converge.
- Data-parallel frameworks, such as MapReduce, are not ideal for these problems: slow
- Graph processing frameworks are optimized for graph-based problems.



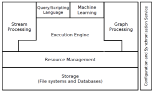
Big Data - Graph Processing

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



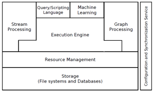
Big Data - Machine Learning

- Implementing and consuming machine learning techniques at scale are difficult tasks for developers and end users.



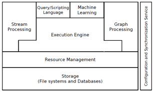
Big Data - Machine Learning

- Implementing and consuming machine learning techniques at scale are difficult tasks for developers and end users.
- There exist platforms that address it by providing scalable machine-learning and data mining libraries.



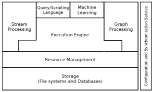
Big Data - Machine Learning

- Implementing and consuming machine learning techniques at scale are difficult tasks for developers and end users.
- There exist platforms that address it by providing scalable machine-learning and data mining libraries.
- Mahout, MLBase, SystemML, Ricardo, Presto, ...



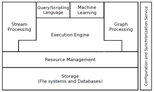
Big Data - Configuration and Synchronization Service

- A means to synchronize distributed applications accesses to shared resources.



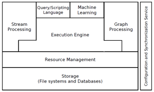
Big Data - Configuration and Synchronization Service

- A means to synchronize distributed applications accesses to shared resources.
- Allows distributed processes to coordinate with each other.



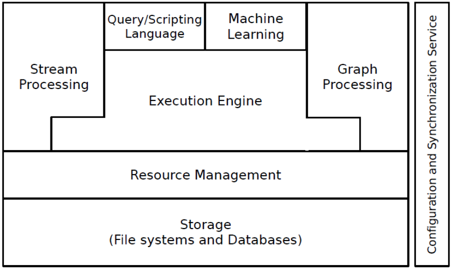
Big Data - Configuration and Synchronization Service

- A means to synchronize distributed applications accesses to shared resources.
- Allows distributed processes to coordinate with each other.
- Zookeeper, Chubby, ...



Summary

Summary


Query/Scripting Language
Machine Learning
Stream Processing
Execution Engine
Graph Processing
Resource Management
Storage
(File systems and Databases)
Configuration and Synchronization Service

Questions?