Hacettepe Üniversitesi
Bilgisayar Mühendisliği Bölümü

BBM371 - Data Management

Big Data

What is Big Data?

Simple Questions

- How long would it take you (together with few friends maybe) to develop an application like WhatsApp?
- Why do you think Facebook bought WhatsApp for $19 Billion?


nature
THE INTERNATIONAL WEEKLY JOURNAL OF SCIENCE
THE BITER BIT
Viral infections for viruses
TROPICAL CYCLONES
The strong get stronger
BLACK HOLE PHYSICS
A new window on the Galactic Centre
BIG DATA
NATUREJOBS
Minnesota musings
SCIENCE IN THE PETABYTE ERA


A tale of two Libyas
The GOP's misinformation campaign
Could your baby be depressed?
TIME
YOUR DATA FOR SALE
Everything about you is being tracked—get over it.
BY JOEL STEIN


POPULAR SCIENCE
THE FUTURE NOW
THE CONTROL CENTERS
OFFICER ALGORITHM
NEW WAYS OF SEEING
PLUS
SPECIAL ISSUE
DATA IS POWER
HOW INFORMATION IS DRIVING THE FUTURE


HBR.ORG
Harvard Business Review
OCTOBER 2012
The True Measures Of Success
10 Rules for Managing Global Innovation
What Ever Happened To Accountability?
GETTING CONTROL OF BIG DATA
How vast new streams of information are changing the art of management
PAGE 59


COUNTRY REPORT
Portugal Casts A Wide Net For Exports
SIBOS SUPERSECTION
How The Blockchain Will Redefine Banking
GULF COOPERATION COUNCIL
Breaking Down Arab Trade Barriers
WORLD'S BEST SUBCUSTODIANS 2015
GLOBAL FINANCE
BIG DATA
The Future Is Now


The Economist
The data deluge
AND HOW TO HANDLE IT: A 14-PAGE SPECIAL REPORT

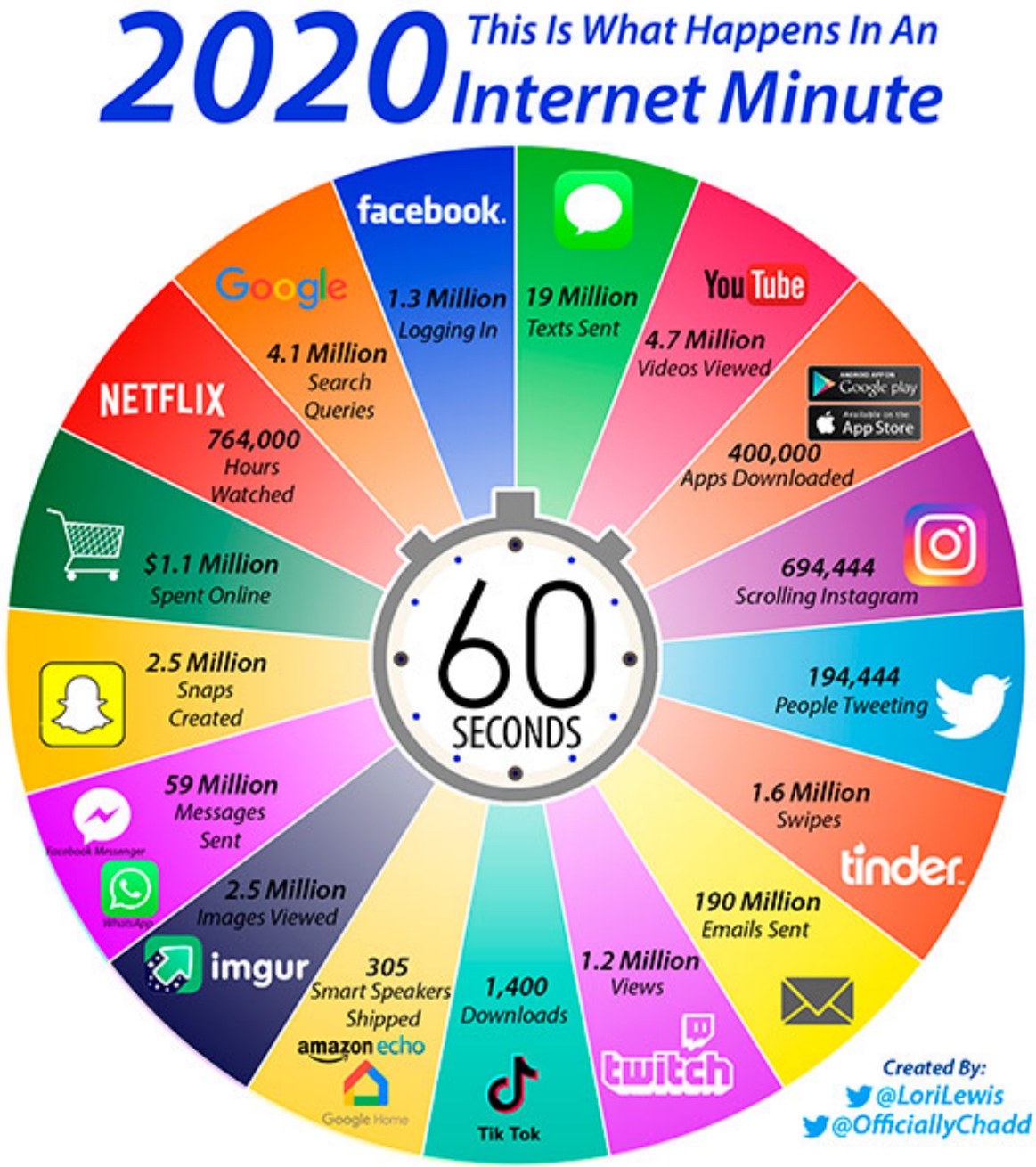
Define Big

- 1 Bit = Binary Digit
- 8 Bits = 1 Byte
- 1000 Bytes = 1 Kilobyte
- 1000 Kilobytes = 1 Megabyte
- 1000 Megabytes = 1 Gigabyte
- 1000 Gigabytes = 1 Terabyte
- 1000 Terabytes = 1 Petabyte
- 1000 Petabytes = 1 Exabyte
- 1000 Exabytes = 1 Zettabyte
- 1000 Zettabytes = 1 Yottabyte
- 1000 Yottabytes = 1 Brontobyte
- 1000 Brontobytes = 1 Geopbyte

Your data (TB/PB)

Your friend's data (TB/PB)

The World (EB)

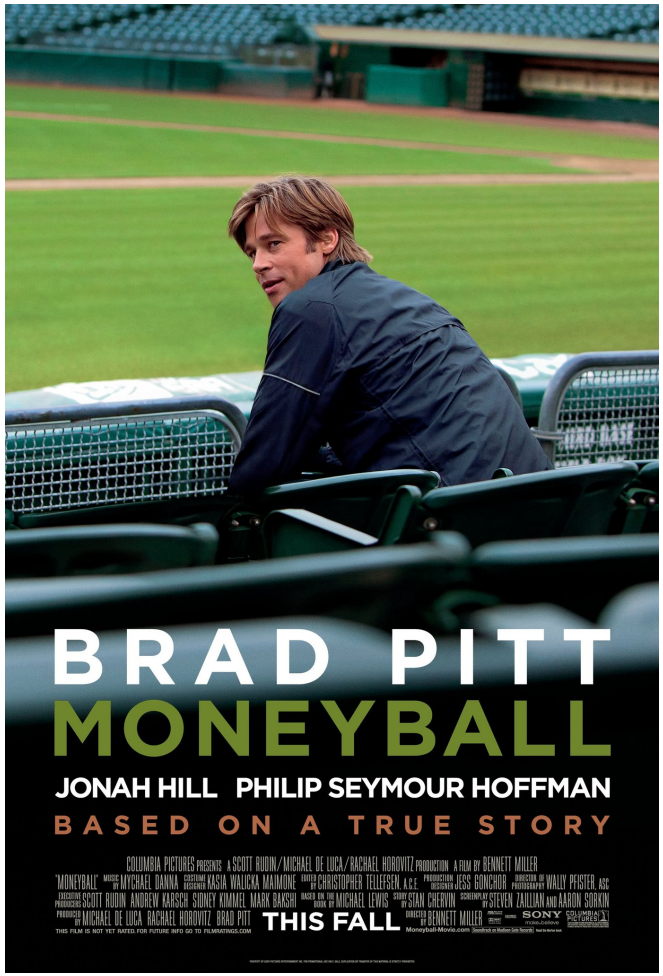


Infographic from Lori Lewis and Chadd Callahan of Cumulus Media

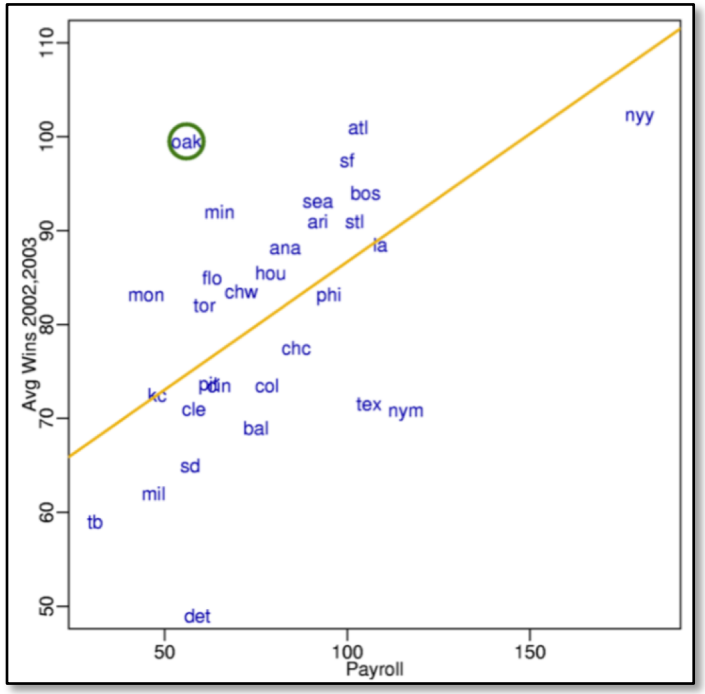
What can You Do with Big Data?

Options are unlimited…

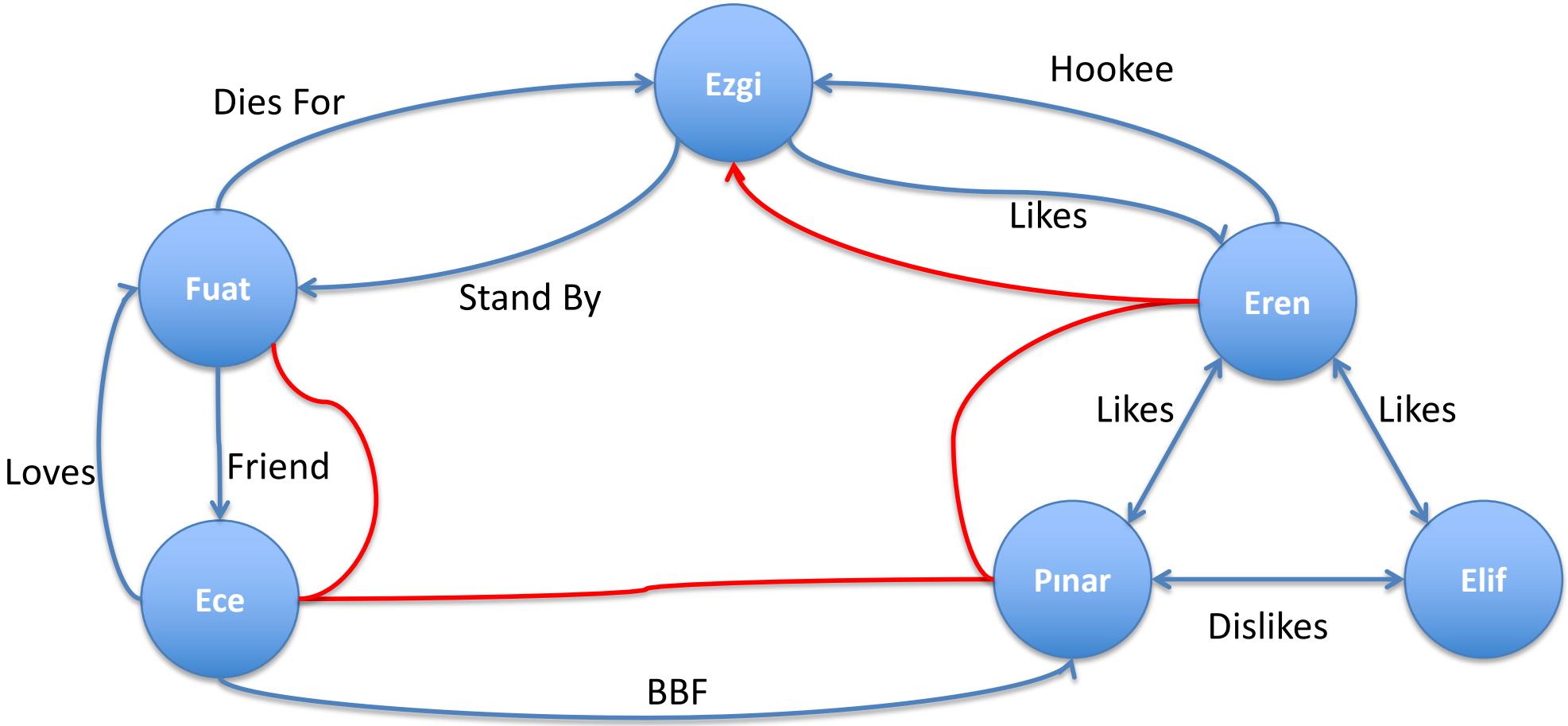
Win the Championship



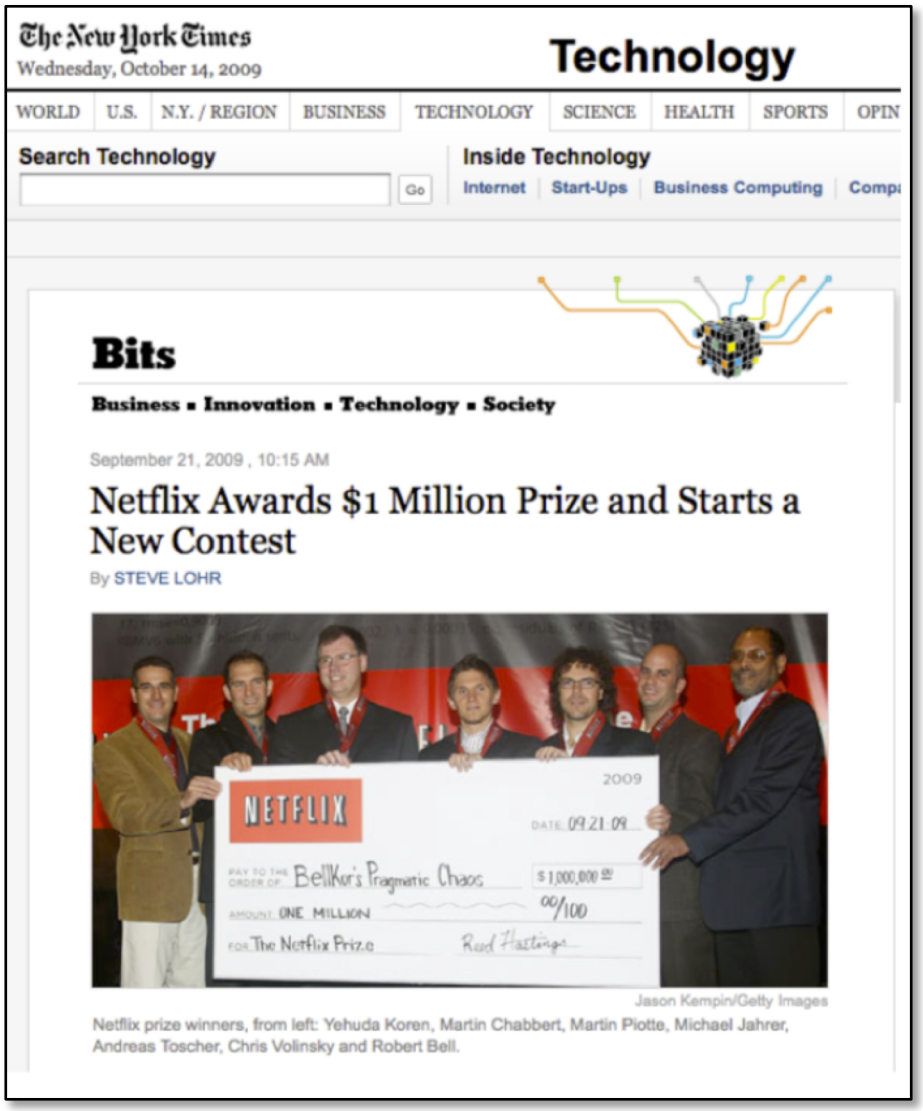
Oakland A's general manager Billy Beane's successful attempt to assemble a baseball team on a lean budget by employing computer-generated analysis to acquire new players.



Discover Complicated Relationships

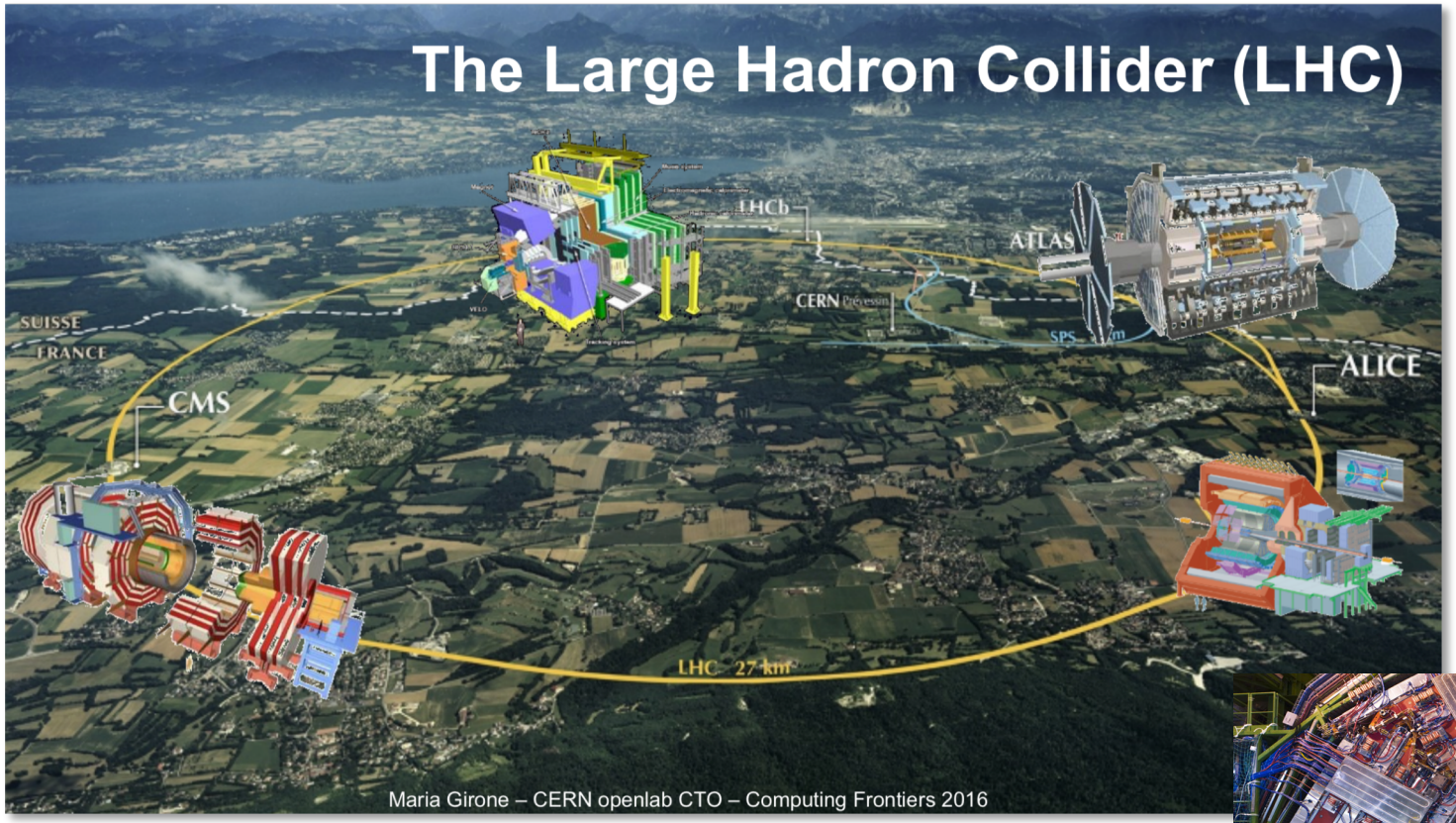


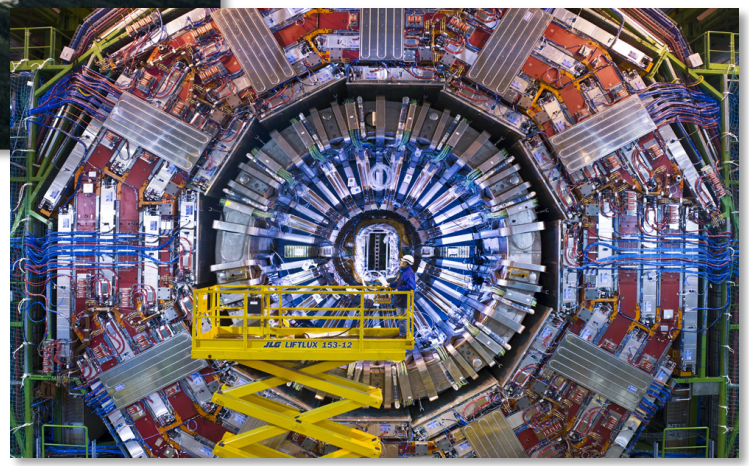
Win Awards: Netflix Challenge



- October 2006: Netflix offers $1M for an improved recommender algorithm.
- 6 years of data for training: 2000-2005
- $1M grand prize for 10% improvement

Do Science: Big Data in Physics

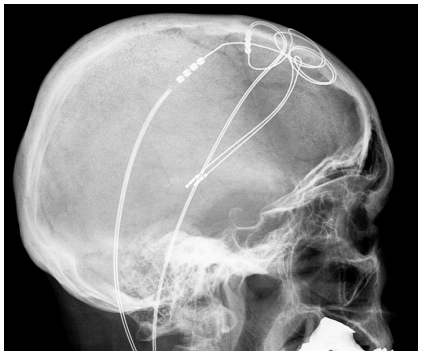


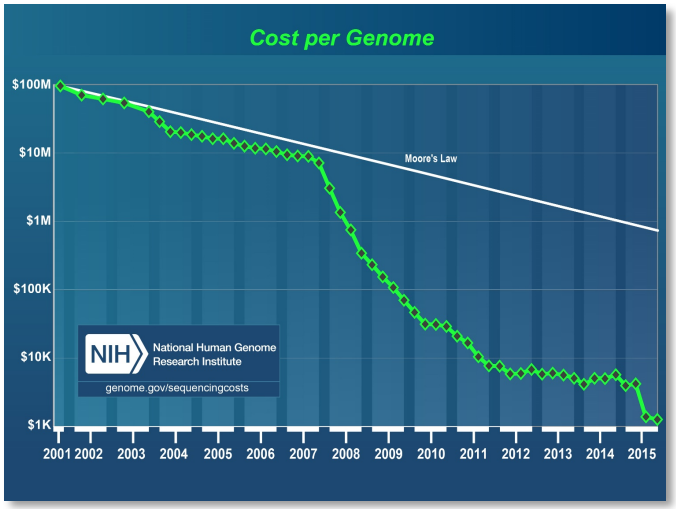


50 Petabytes of data per year!

Care for Health

- Faster and cheaper technology and data storage
- Widespread sensing devices
- An increase in “born” digital data
- Greater availability of data via repositories
- Data sharing mandates





NIH U.S. National Library of Medicine
National Network of Libraries of Medicine
Pacific Northwest Region

Why is Big Data Processing Difficult?

- You need more data than your data warehouse.
  - you need more data than you have
  - logs, Twitter feeds, blogs, customer surveys, ...
- You need to ask the right questions.
  - data alone is silent
- You need technology and organization that help you concentrate on asking the right questions.

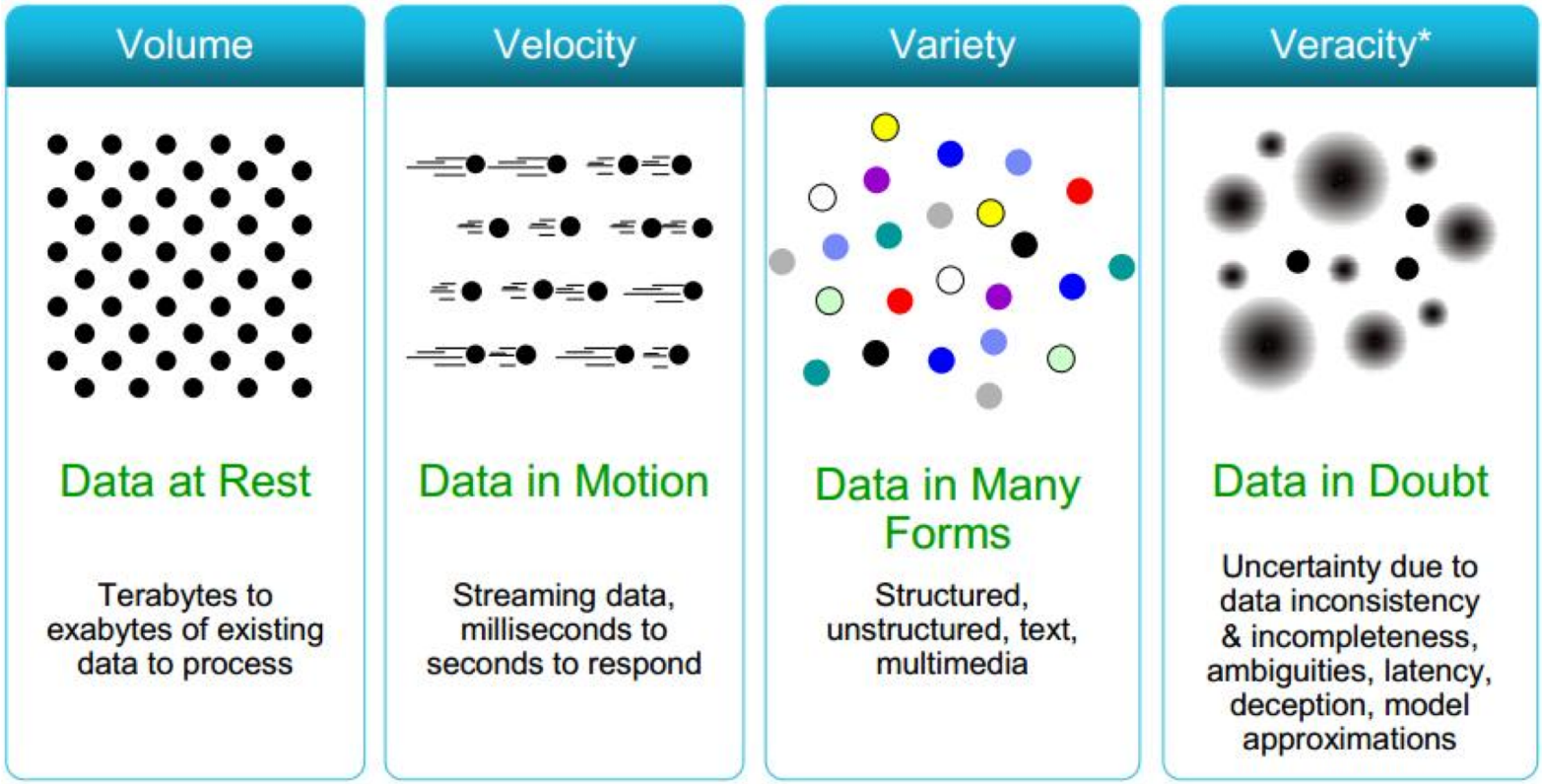
Is bigger = smarter?

- Yes!
  - tolerate errors
  - discover the long tail and corner cases
  - machine learning works much better
- But!
  - more data, more error (e.g., semantic heterogeneity)
  - with enough data you can prove anything
  - still need humans to ask right questions

Big Data Success Story

- Google Translate
  - you collect snipets of translations
  - you match sentences to snipets
  - you continuously debug your system
- Why does it work?
  - there are tons of snipets on the Web
  - there is a ground truth that helps to debug system

Four V's of Big Data



More V's

- Variability
  - Inconsistency in data, inconsistent speed at which big data is loaded into your database
- Validity
  - Similar to veracity, validity refers to how accurate and correct the data is for its intended use
- Vulnerability
  - Big data brings new security question
- Volatility
  - You cannot store data indefinitely anymore
- Visualization
  - Challenge to visualize, need different ways to represent data
- Value
  - The other Vs are meaningless if you don't derive business value from the data

Big Data Processing

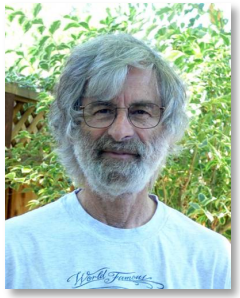
Problem

- How do you scale up applications?
  - Run jobs processing 100's of terabytes of data
  - Takes days to read on 1 computer
- Need lots of cheap computers organized as a cluster
  - Fixes speed problem (15 minutes on 1000 computers), but…
  - Reliability problems
    - In large clusters, computers fail every day
      - 1 Computer, 1 failure in a year
      - 365 computer, 1 failure a day ☺
    - Cluster size is not fixed
- Need common infrastructure
  - Must be efficient and reliable

Distributed Systems

- Allows developers to use multiple machines for a single task
- Programming on a distributed system is much more complex
  - Synchronizing data exchanges
  - Managing a finite bandwidth
  - Controlling computation timing is complicated
- Distributed systems must be designed with the expectation of failure

"A distributed system is one that *prevents you* from *working* because of the failure of a machine that *you* had never heard of." *Leslie Lamport*



Partial Failures

- Failure of a single component must not cause the failure of the entire system only a degradation of the application performance
- Failure should not result in the loss of any data
- If a component fails, it should be able to recover without restarting the entire system
- Component failure or recovery during a job must not affect the final output

Scalability

- Increasing resources should increase load capacity
- Increasing the load on the system should result in a graceful decline in performance for all jobs
  - Not system failure

Solution: Hadoop

- Open Source Apache Project
  - An open-source software framework for storage and large scale processing of data-sets on clusters of commodity hardware.
- Hadoop Core includes:
  - Distributed File System - distributes data
  - Map/Reduce - distributes application
- Runs on
  - Linux, Mac OS/X, Windows, and Solaris
  - Commodity hardware

Btw, there are other solutions. We are going to focus on Hadoop for the moment.

Little Hadoop History

- Hadoop was created by Doug Cutting and Mike Cafarella in 2005.

- Cutting, who was working at Yahoo! at the time, named it after his son's toy elephant.

  – "Being a guy in the software business, we're always looking for names," Cutting said. "I'd been saving it for the right time."

Example Hadoop Cluster

- ~20,000 machines running Hadoop
- Largest clusters are currently 2000 nodes
- Several petabytes of user data
- Hundreds of thousands of jobs every month

Numbers are probably outdated and growing.

YAHOO! DEVELOPER NETWORK

MY PROJECTS | TOOLS & SERVICES | RESOURCES

YDN / Hadoop and Distributed Computing at Yahoo! / Hadoop Sorts a Petabyte in 16.25 Hours and a Terabyte in 62 S

Yahoo! Developer Network Blog

« Previous | Main | Next »

MAY 11, 2009

Hadoop Sorts a Petabyte in 16.25 Hours and a Terabyte in 62 Seconds

We used **Apache Hadoop** to compete in **Jim Gray's Sort** benchmark. Jim's Gray's sort benchmark consists of a set of many related benchmarks, each with their own rules. All of the sort benchmarks measure the time to sort different numbers of 100 byte records. The first 10 bytes of each record is the key and the rest is the value. The **minute sort** must finish end to end in less than a minute. The **Gray sort** must sort more than 100 terabytes and must run for at least an hour. The best times we observed were:

Bytes	Nodes	Maps	Reduces	Replication	Time
500,000,000,000	1406	8000	2600	1	59 seconds
1,000,000,000,000	1460	8000	2700	1	62 seconds
100,000,000,000,000	3452	190,000	10,000	2	173 minutes
1,000,000,000,000,000	3658	80,000	20,000	2	975 minutes

Hadoop Origin and Idea

- Based on work done by Google in the early 2000s
  - "The Google File System" in 2003
  - "MapReduce: Simplified Data Processing on Large Clusters" in 2004
- The core idea was to distribute the data as it is initially stored
  - Each node can then perform computation on the data it stores without moving the data for the initial processing

Hadoop Components

- Hadoop Distributed file system (HDFS)
  - Single namespace for entire cluster
  - Replicates data for fault-tolerance
- MapReduce framework
  - Executes user jobs specified as “map” and “reduce” functions
  - Manages work distribution & fault-tolerance

HDFS

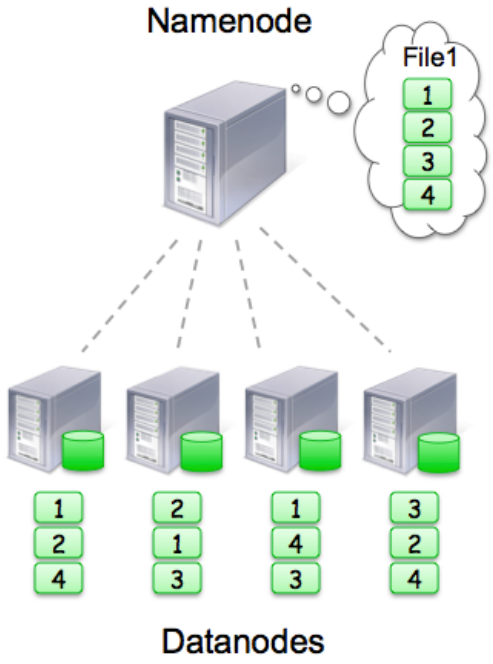
- The Hadoop Distributed File System (HDFS) is a distributed file system designed to run on commodity hardware.
  - HDFS is highly fault-tolerant and is designed to be deployed on low-cost hardware.
  - HDFS provides high throughput access to application data and is suitable for applications that have large data sets.

Assumptions and Goals

- Detection of hardware faults and quick, automatic recovery from them.
- Designed more for batch processing rather than interactive use by users.
- Support for large files.
- Follows a write-once-read-many access model for files.
- Provides interfaces for applications to move themselves closer to where the data is located.
- Designed to be easily portable from one platform to another.

HDFS Pursues a master-slave Model

- NameNode
  - Executes file system namespace operations like opening, closing, and renaming files and directories.
  - Determines the mapping of blocks to DataNodes.
- DataNodes
  - Manage attached storage.
  - Internally, a file is split into one or more blocks and these blocks are stored in a set of DataNodes.
  - The DataNodes are responsible for serving read and write requests from the file system's clients.
  - The DataNodes also perform block creation, deletion, and replication upon instruction from the NameNode.



Data Replication

- HDFS is designed to reliably store very large files across machines in a large cluster.
- It stores each file as a sequence of blocks.
  - The blocks of a file are replicated for fault tolerance.
  - The block size and replication factor are configurable per file.
  - An application can specify the number of replicas of a file.
  - The replication factor can be specified at file creation time and can be changed later.
- Files in HDFS are write-once and have strictly one writer at any time.
- The NameNode makes all decisions regarding replication of blocks.
  - It periodically receives a Heartbeat and a Blockreport from each of the DataNodes in the cluster.
  - Receipt of a Heartbeat implies that the DataNode is functioning properly. A Blockreport contains a list of all blocks on a DataNode.

Block Placement

- Files are split into fixed sized blocks and stored on data nodes (Default 64MB)
- Data blocks are replicated for fault tolerance and fast access (Default is 3)
- Where to put a given block by default?
  - **First copy** is written to the node creating the file (write affinity)
  - **Second copy** is written to a data node within the same rack
  - **Third copy** is written to a data node in a different rack

Replica Selection

- To minimize global bandwidth consumption and read latency, HDFS tries to satisfy a read request from a replica that is **closest to the reader**.

- If there exists a **replica on the same rack** as the reader node, then that replica is preferred to satisfy the read request.

- If an HDFS cluster spans multiple data centers, then a replica that is **resident in the local data center** is preferred over any remote replica.

Disk Failures

- Each DataNode sends a Heartbeat message to the NameNode periodically.
- The NameNode marks DataNodes without recent Heartbeats as dead and does not forward any new IO requests to them.
- Any data that was registered to a dead DataNode is not available to HDFS any more.
- DataNode death may cause the replication factor of some blocks to fall below their specified value.
- The NameNode constantly tracks which blocks need to be replicated and initiates replication whenever necessary.

Re-replication

- The necessity for re-replication may arise due to many reasons:
  - a DataNode may become unavailable.
  - a replica may become corrupted.
  - a hard disk on a DataNode may fail.
  - the replication factor of a file may be increased.

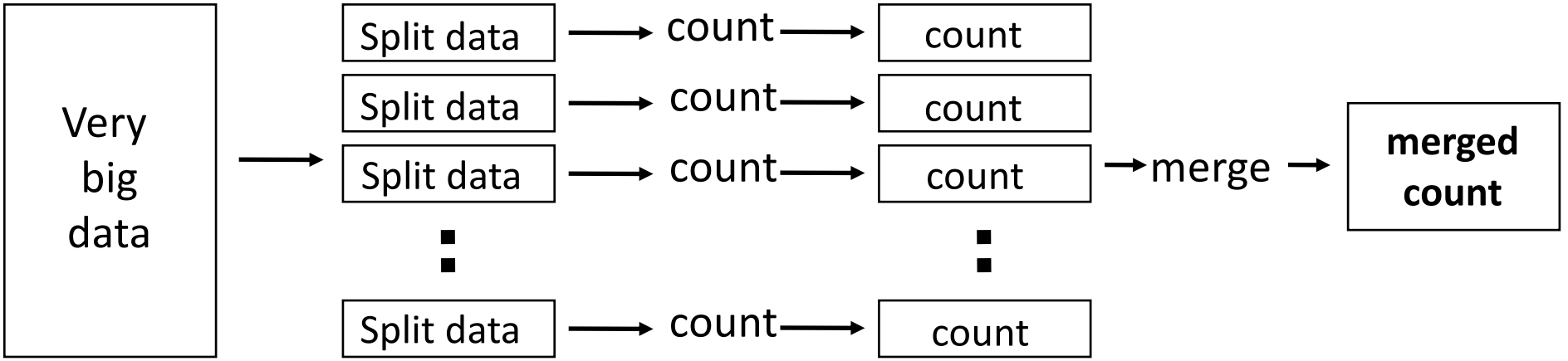
Cluster Rebalancing

- The HDFS architecture is compatible with data rebalancing schemes.
- A scheme might automatically move data from one DataNode to another if the free space on a DataNode falls below a certain threshold.
- In the event of a sudden high demand for a particular file, a scheme might dynamically create additional replicas and rebalance other data in the cluster.
  - These types of data rebalancing schemes are not yet implemented.

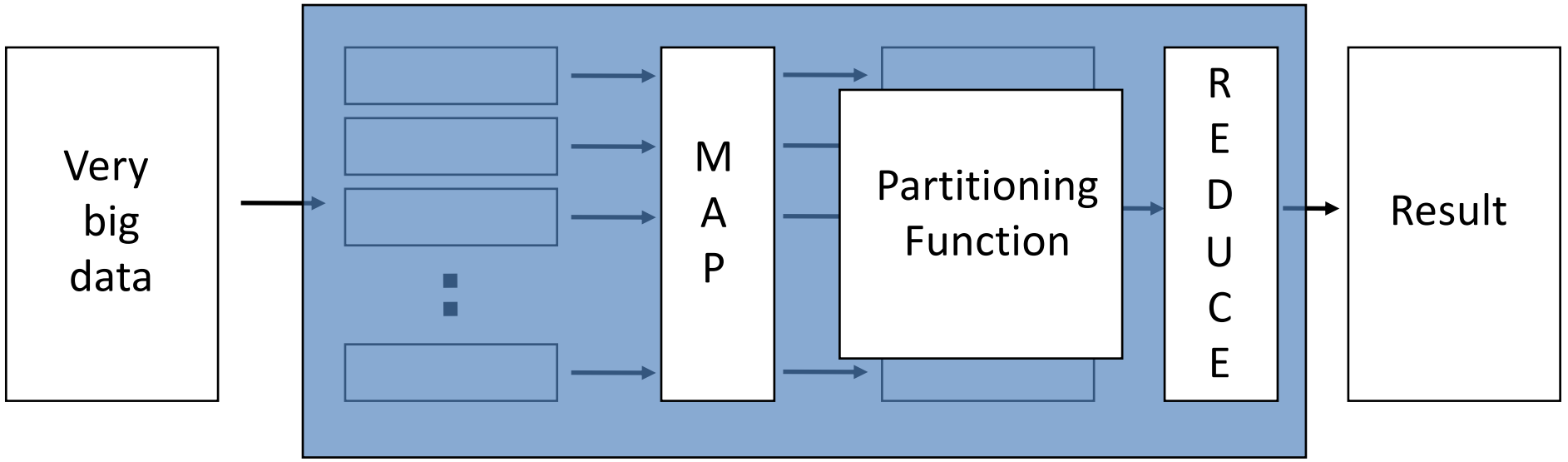
MapReduce

- Platform for reliable, scalable parallel computing
- Abstracts issues of distributed and parallel environment from programmer
  - Programmer provides core logic (via map() and reduce() functions)
  - System takes care of parallelization of computation, coordination, etc.
- Paradigm dates back many decades
  - But very large scale implementations running on clusters with 10^3 to 10^4 machines are more recent
  - Google Map Reduce, Hadoop, ..
- Data storage/access typically done using distributed file systems or key-value stores

Distributed Word Count

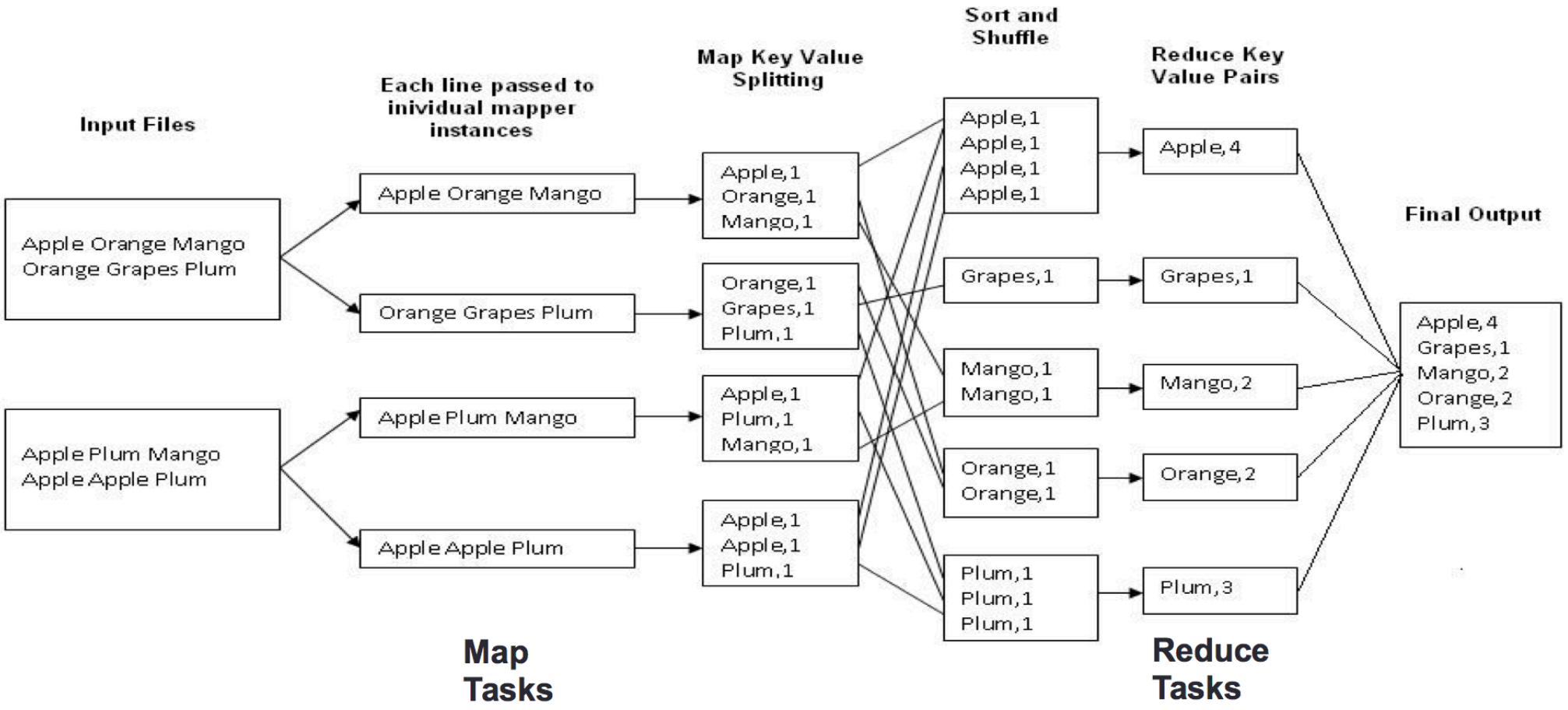


Map + Reduce



- Map:
  - Accepts *input* key/value pair
  - Emits *intermediate* key/value pair
- Reduce :
  - Accepts *intermediate* key/value* pair
  - Emits *output* key/value pair

Word Count Example



Good Starting Point for Hadoop

- Hortonworks provides a sandbox which is provided as a self-contained virtual machine.

- No data center, no cloud service and no internet connection needed!

http://hortonworks.com/products/hortonworks-sandbox/

Smile ☺

- What is Big Data?
  - “Saklamak için dahi odalar dolusu hard disk gerektiren veridir”
- What are four Vs?
  - ”Hocam sabah bakmıştım ama şu an hatırımdan çıkmış durumda walla.”

Example midterm questions and answers ☺