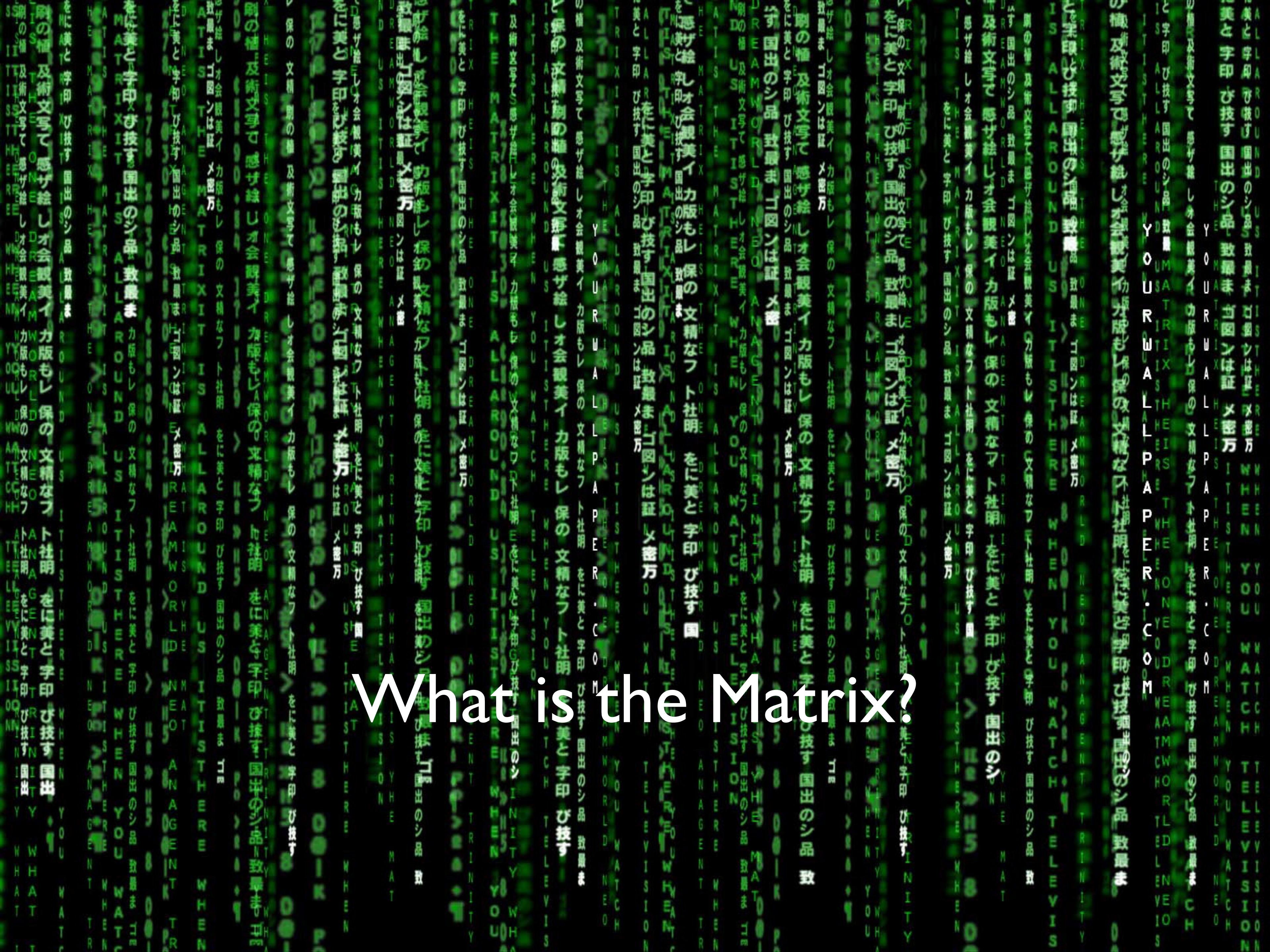
INFM 603: Information Technology and Organizational Context

Session 11: Cloud Computing and Big Data

Jimmy Lin
The iSchool
University of Maryland

Thursday, November 15, 2012



What is the Matrix?

What is cloud computing?

Source: Wikipedia (Clouds)

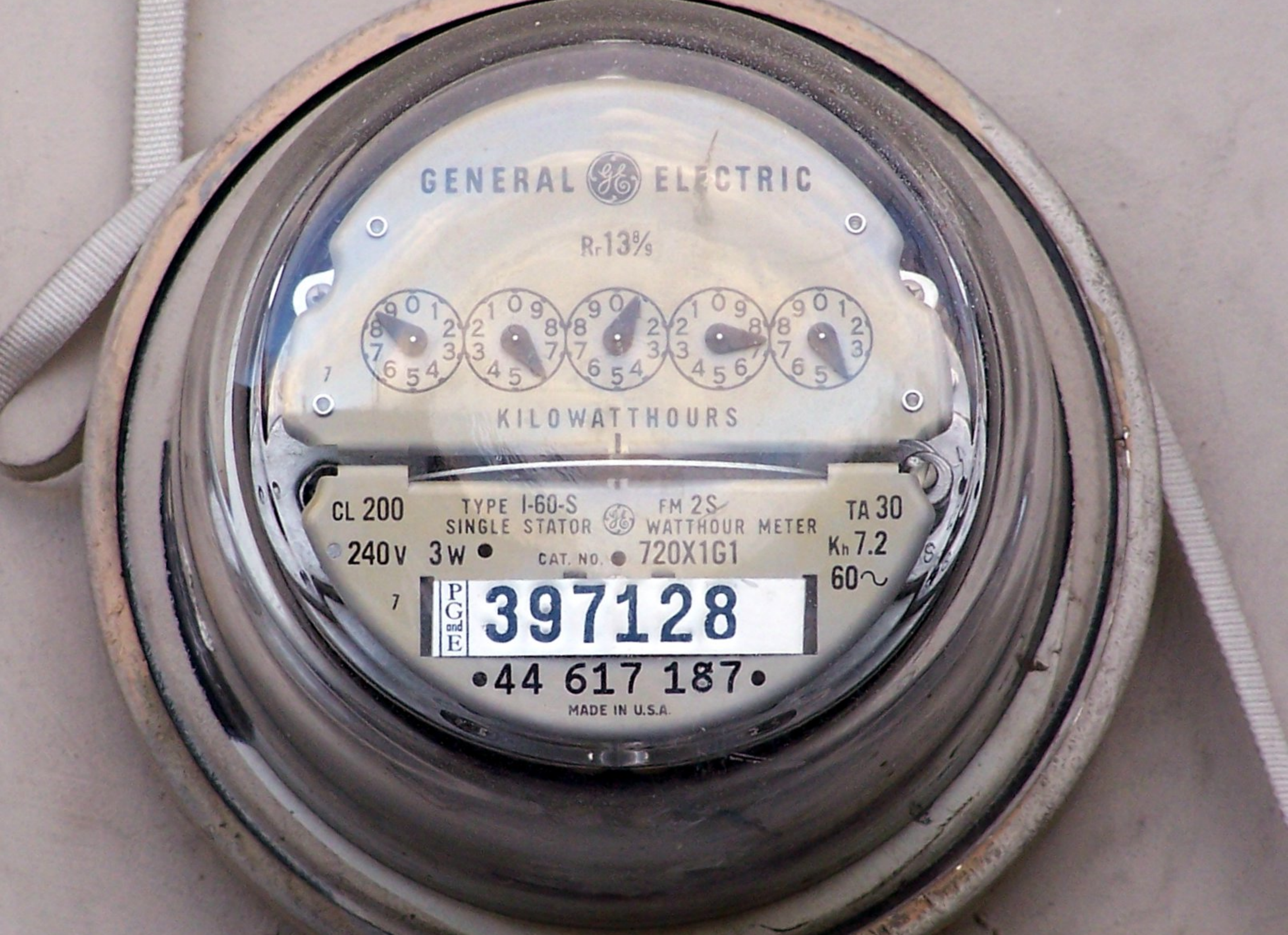
Source: flickr (60in3/2338247189)

The best thing since sliced bread?

- Before clouds…
  - Grids
  - Connection machines
  - Vector supercomputers
  - …
- Cloud computing means many different things:
  - Large-data processing
  - Rebranding of web 2.0
  - Utility computing
  - Everything as a service

Rebranding of web 2.0

- Rich, interactive web applications
  - Clouds refer to the servers that run them
  - AJAX as the de facto standard (for better or worse)
  - Examples: Facebook, YouTube, Gmail, …
- "The network is the computer": take two
  - User data is stored "in the clouds"
  - Rise of the tablets, smartphones, etc.
  - Browser is the OS

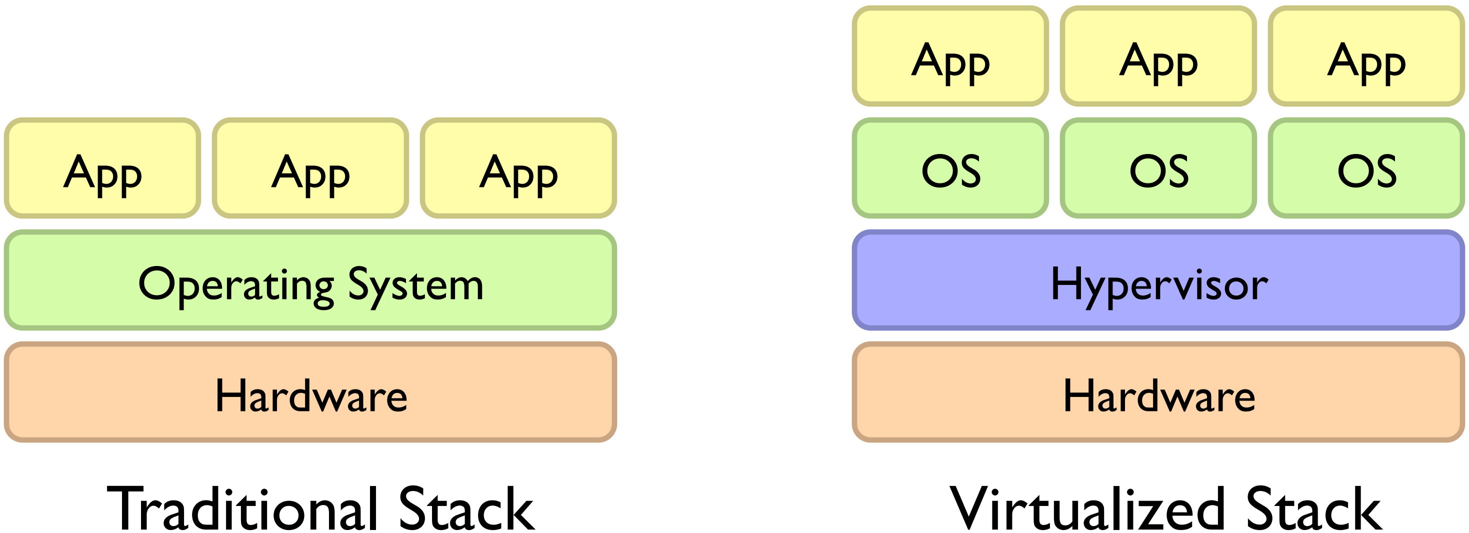


Source: Wikipedia (Electricity meter)

Utility Computing

- What?
  - Computing resources as a metered service (“pay as you go”)
  - Ability to dynamically provision virtual machines
- Why?
  - Cost: capital vs. operating expenses
  - Scalability: “infinite” capacity
  - Elasticity: scale up or down on demand
- Does it make sense?
  - Benefits to cloud users
  - Business case for cloud providers

Enabling Technology: Virtualization



Everything as a Service

- Utility computing = Infrastructure as a Service (IaaS)
  - Why buy machines when you can rent them?
  - Examples: Amazon's EC2, Rackspace
- Platform as a Service (PaaS)
  - Give me nice API and take care of the maintenance, upgrades, …
  - Example: Google App Engine
- Software as a Service (SaaS)
  - Just run it for me!
  - Example: Gmail, Salesforce

Different Types of Clouds

- Public clouds
- Private clouds
- Hybrid clouds


Our World: Large Data
Source: Wikipedia (Hard disk drive)

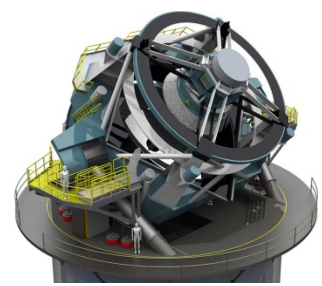
Google processes 20 PB a day (2008)

Wayback Machine:
10 PB web archive (10/2012)

eBay
10 PB data in Hadoop/Teradata
75B DB calls/day (6/2012)

LHC: 15 PB a year

100+ PB user data
+500 TB/day (8/2012)



640K ought to be enough for anybody.

LSST: 6-10 PB a year (~2015)

How much data?

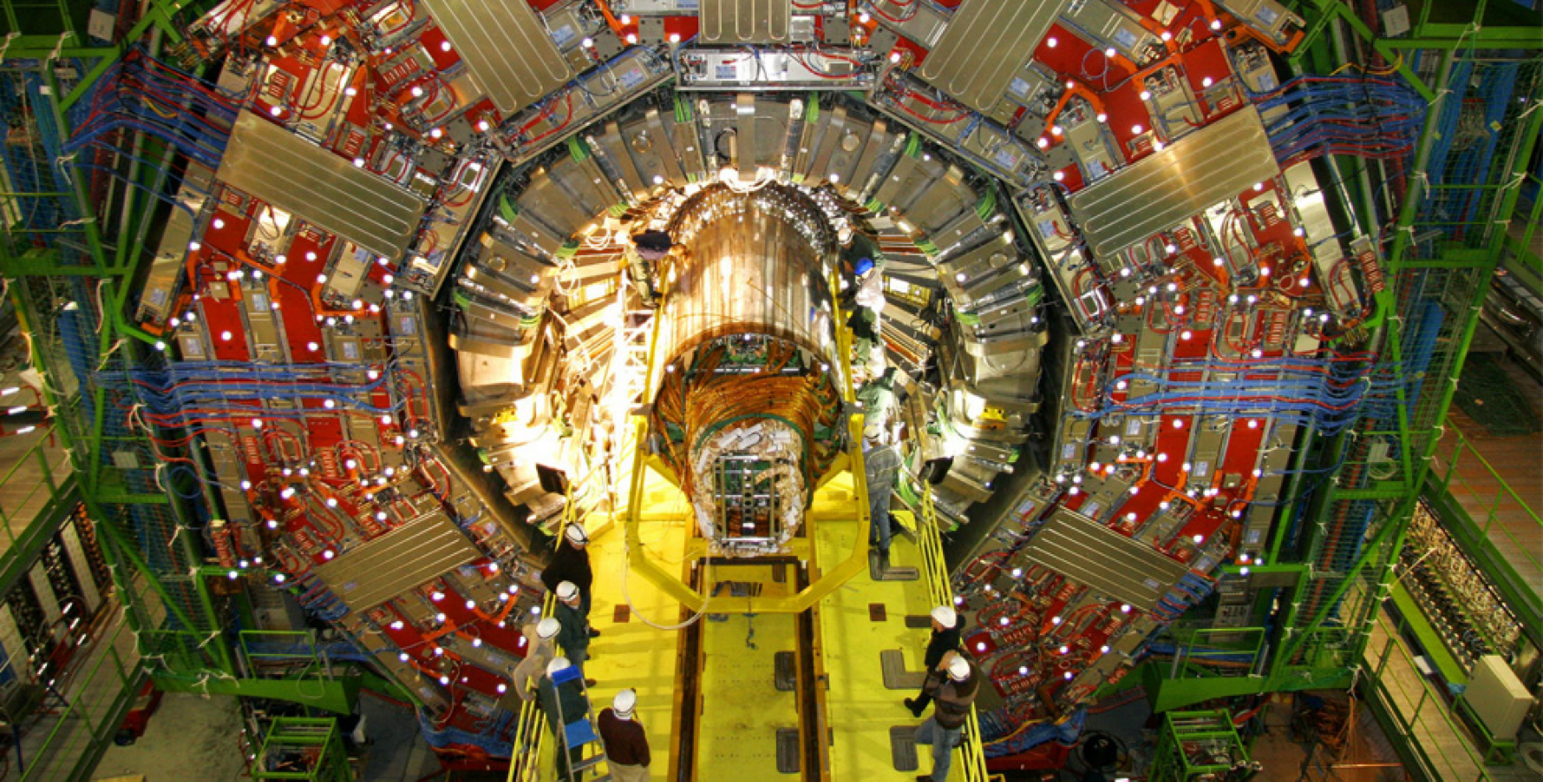
Why large data?

Science

Engineering

Commerce

Source: Wikipedia (Everest)

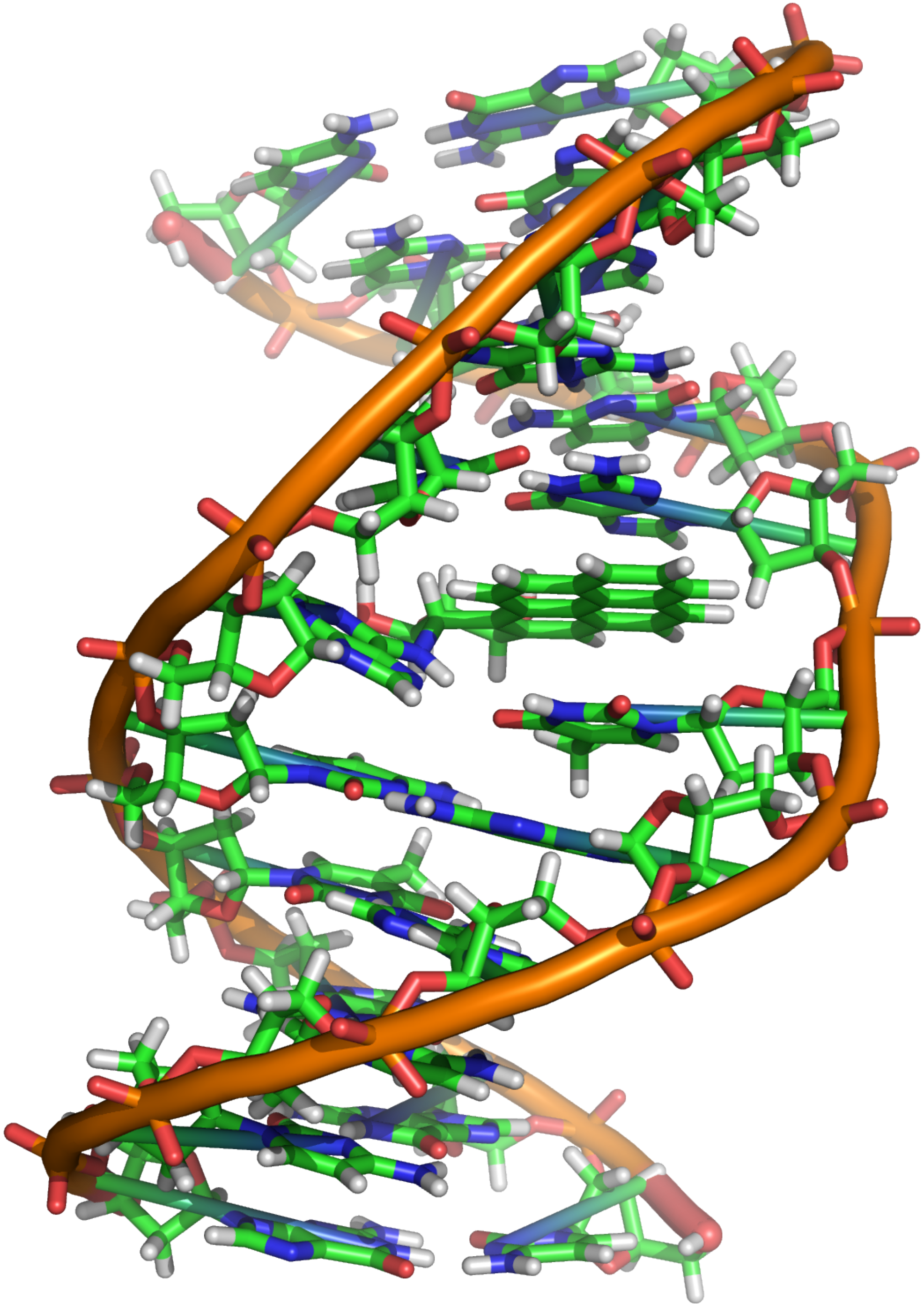


Science

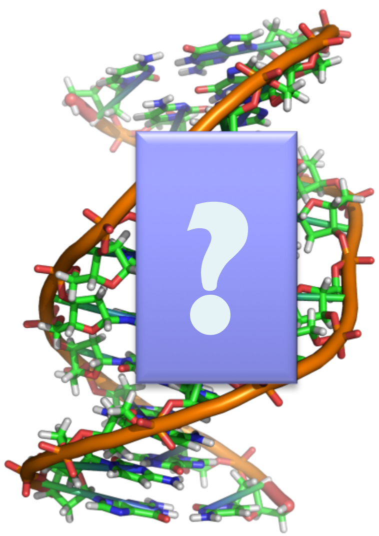
- Emergence of the 4th Paradigm
- Data-intensive e-Science

Maximilien Brice, © CERN

Maximilien Brice, © CERN



Source: Wikipedia (DNA)



Subject genome

Sequencer

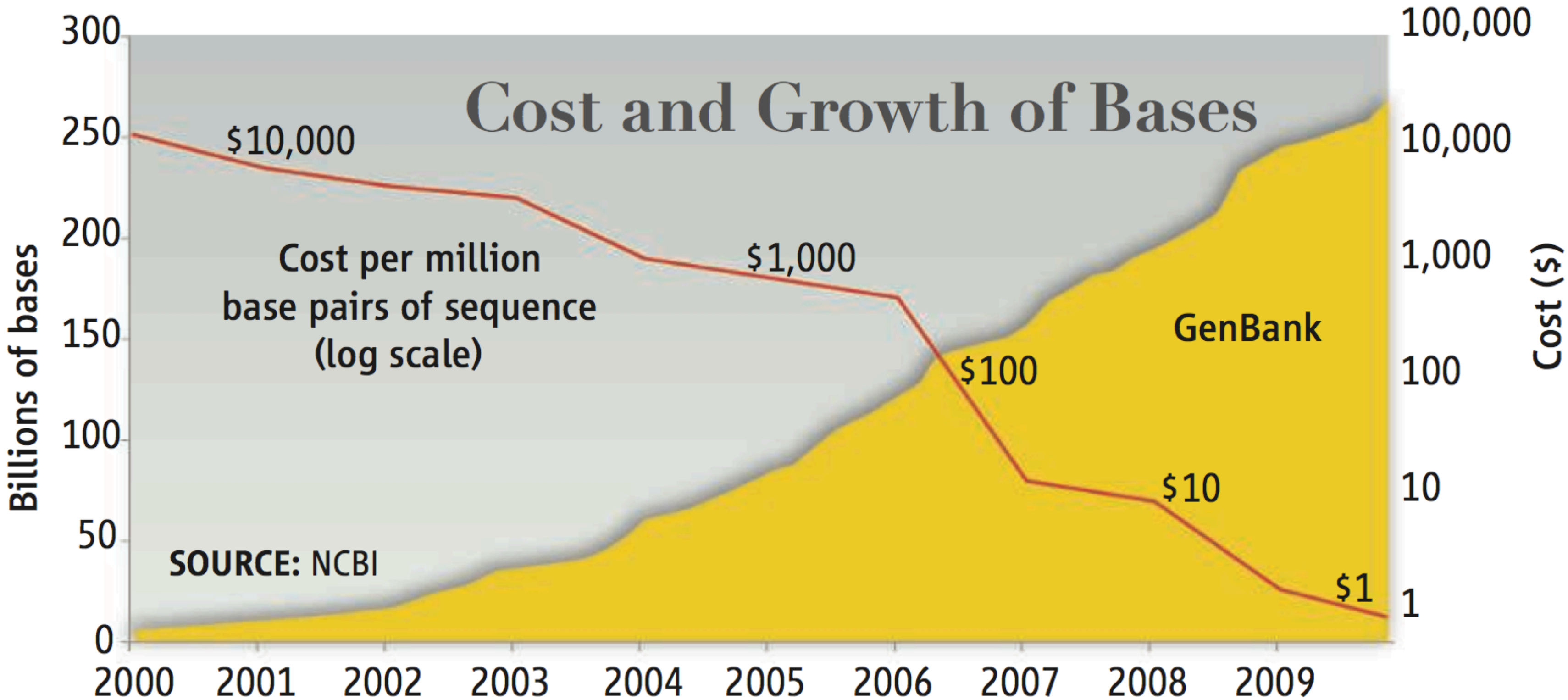
GATGCTTACTATGCGGGCCCC
CGGTCTAATGCTTACTATGC
GCTTACTATGCGGGCCCCTT
AATGCTTACTATGCGGGCCCCTT
TAATGCTTACTATGC
AATGCTTAGCTATGCGGGC
AATGCTTACTATGCGGGCCCCTT
AATGCTTACTATGCGGGCCCCTT
CGGTCTAGATGCTTACTATGC
AATGCTTACTATGCGGGCCCCTT
CGGTCTAATGCTTAGCTATGC
ATGCTTACTATGCGGGCCCCTT

Reads

Human genome: 3 gbp
A few billion short reads
(~100 GB compressed data)

DNA Data Tsunami

Current world-wide sequencing capacity exceeds 13 Pbp/year and is growing at 5x per year!



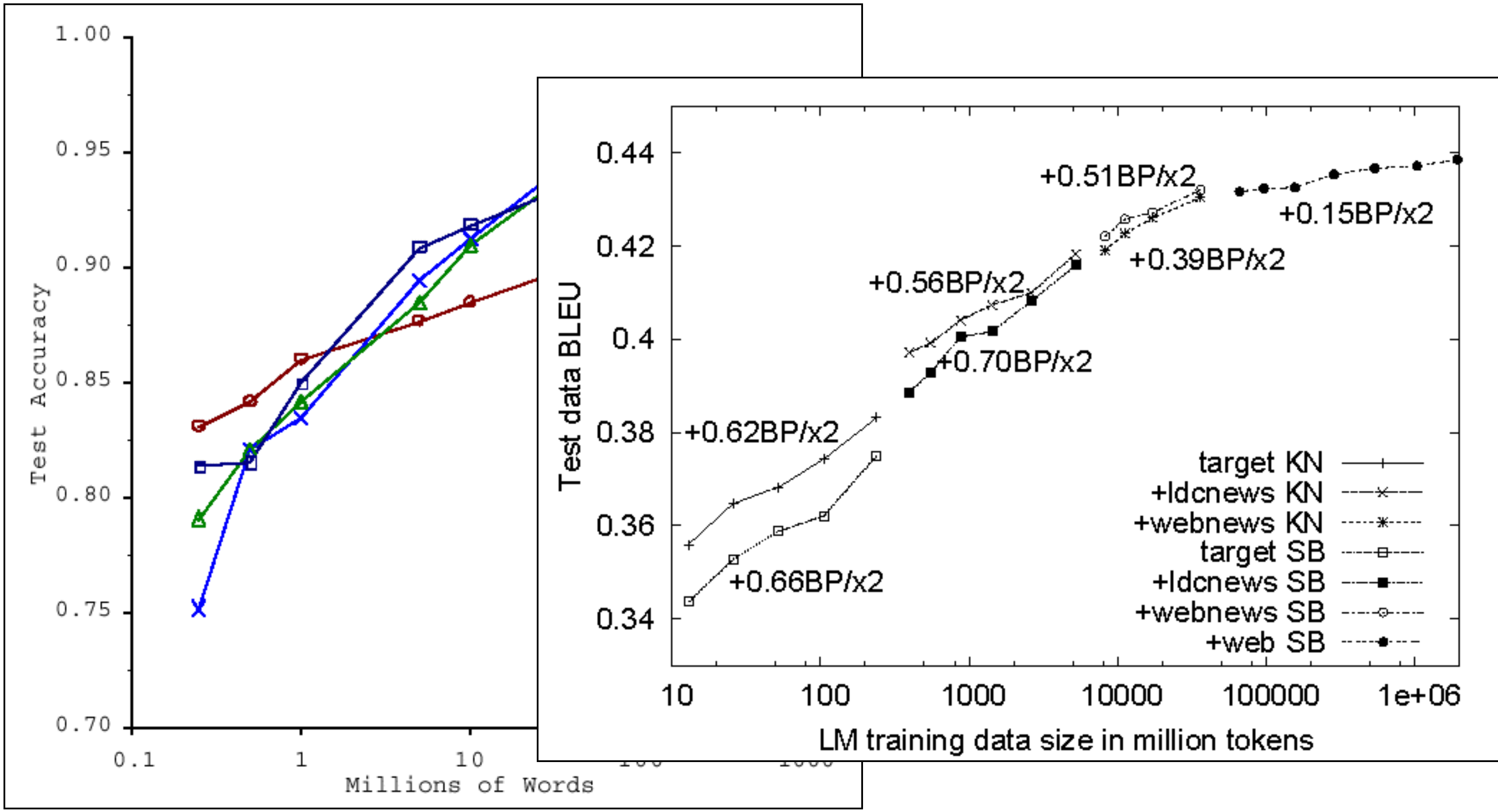
"Will Computers Crash Genomics?"
Elizabeth Pennisi (2011) *Science*. 331(6018): 666-668.

Engineering

- The unreasonable effectiveness of data
- Count and normalize!

Source: Wikipedia (Three Gorges Dam)

No data like more data!



(Banko and Brill, ACL 2001)
(Brants et al., EMNLP 2007)

What to do with more data?

- Answering factoid questions
  - Pattern matching on the Web
  - Works amazingly well

    Who shot Abraham Lincoln? → X shot Abraham Lincoln

- Learning relations
  - Start with seed instances
  - Search for patterns on the Web
  - Using patterns to find more instances

Wolfgang Amadeus Mozart (1756 - 1791)
Einstein was born in 1879

Birthday-of(Mozart, 1756)
Birthday-of(Einstein, 1879)

PERSON (DATE –
PERSON was born in DATE

(Brill et al., TREC 2001; Lin, ACM TOIS 2007)
(Agichtein and Gravano, DL 2000; Ravichandran and Hovy, ACL 2002; …)

Commerce

- Know thy customers
- Data → Insights → Competitive advantages

Source: Wikiedia (Shinjuku, Tokyo)

Business Intelligence

- Premise: more data leads to better business decisions
  - Periodic reporting as well as ad hoc queries
  - Rise of the data scientist
  - Listen to your customers, not the HiPPO
- Examples:
  - Slicing-and-dicing activity by different dimensions to better understand the marketplace
  - Analyzing log data to improve front-end experience
  - Analyzing log data to better optimize ad placement
  - Analyzing purchasing trends for better supply-chain management
  - Mining for correlations between otherwise unrelated activities

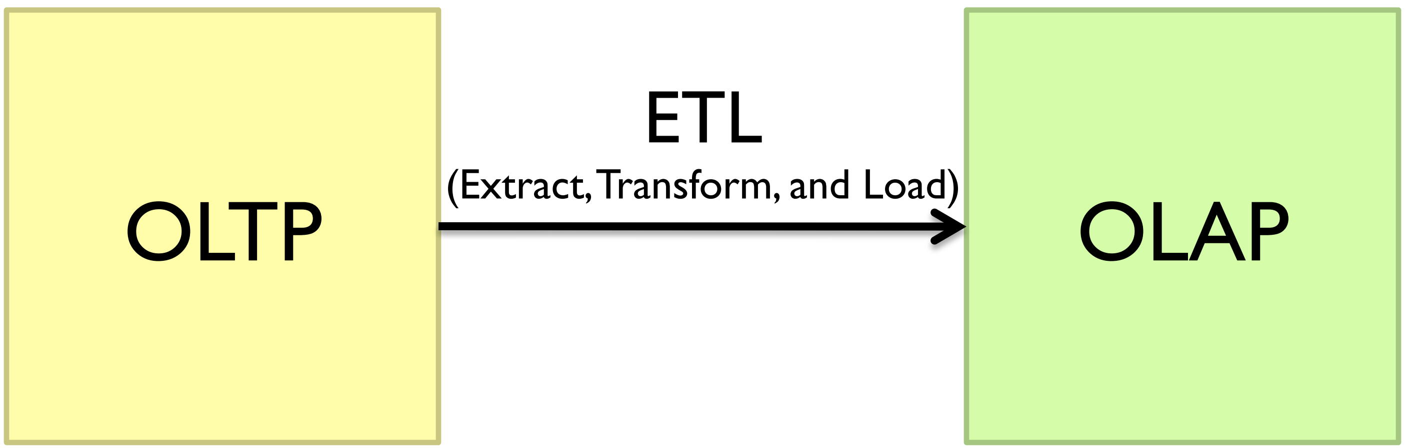
Database Workloads

- OLTP (online transaction processing)
  - Typical applications: e-commerce, banking, airline reservations
  - User facing: real-time, low latency, highly-concurrent
  - Tasks: relatively small set of "standard" transactional queries
  - Data access pattern: random reads, updates, writes (involving relatively small amounts of data)
- OLAP (online analytical processing)
  - Typical applications: business intelligence, data mining
  - Back-end processing: batch workloads, less concurrency
  - Tasks: complex analytical queries, often ad hoc
  - Data access pattern: table scans (involving large amounts of data)

One Database or Two?

- Downsides of co-existing OLTP and OLAP workloads
  - Poor memory management
  - Conflicting data access patterns
  - Variable latency
- Solution: separate databases
  - User-facing OLTP database for high-volume transactions
  - Data warehouse for OLAP workloads
  - How do we connect the two?

OLTP/OLAP Architecture



OLTP/OLAP Integration

- OLTP database for user-facing transactions
  - Retain records of all activity
  - Periodic ETL (e.g., nightly)
- Extract-Transform-Load (ETL)
  - Extract records from source
  - Transform: clean data, check integrity, aggregate, etc.
  - Load into OLAP database
- OLAP database for data warehousing
  - Business intelligence: reporting, ad hoc queries, data mining, etc.
  - Feedback to improve OLTP services

Challenge of Big Data

- Volume
- Cost
- ETL Latency

Source: Wikipedia (Clouds)

Cloud Computing Meets Big Data

- Rise of social media and user-generated content
  - Cloud services exacerbates big data problems
- Utility computing democratizes big data capabilities
  - Efficient dynamic allocation of large-scale computing resources

What *really* is the cloud?

Source: Wikipedia (Clouds)

Source: Wikipedia (The Dalles, Oregon)

Source: Google

Source: Bonneville Power Administration

Source: Google

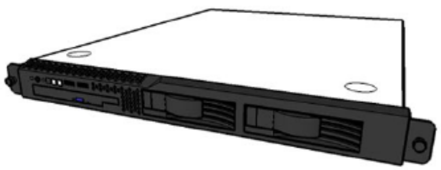
Source: Google

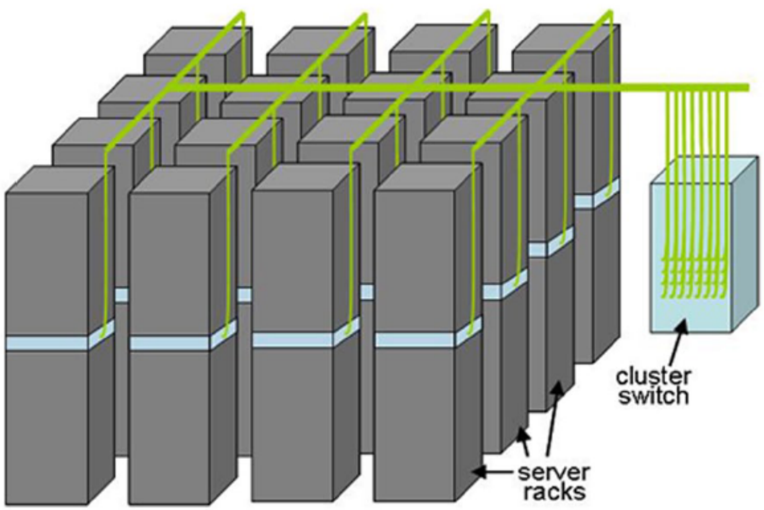
Source: Facebook

Source: Facebook


Source: Facebook

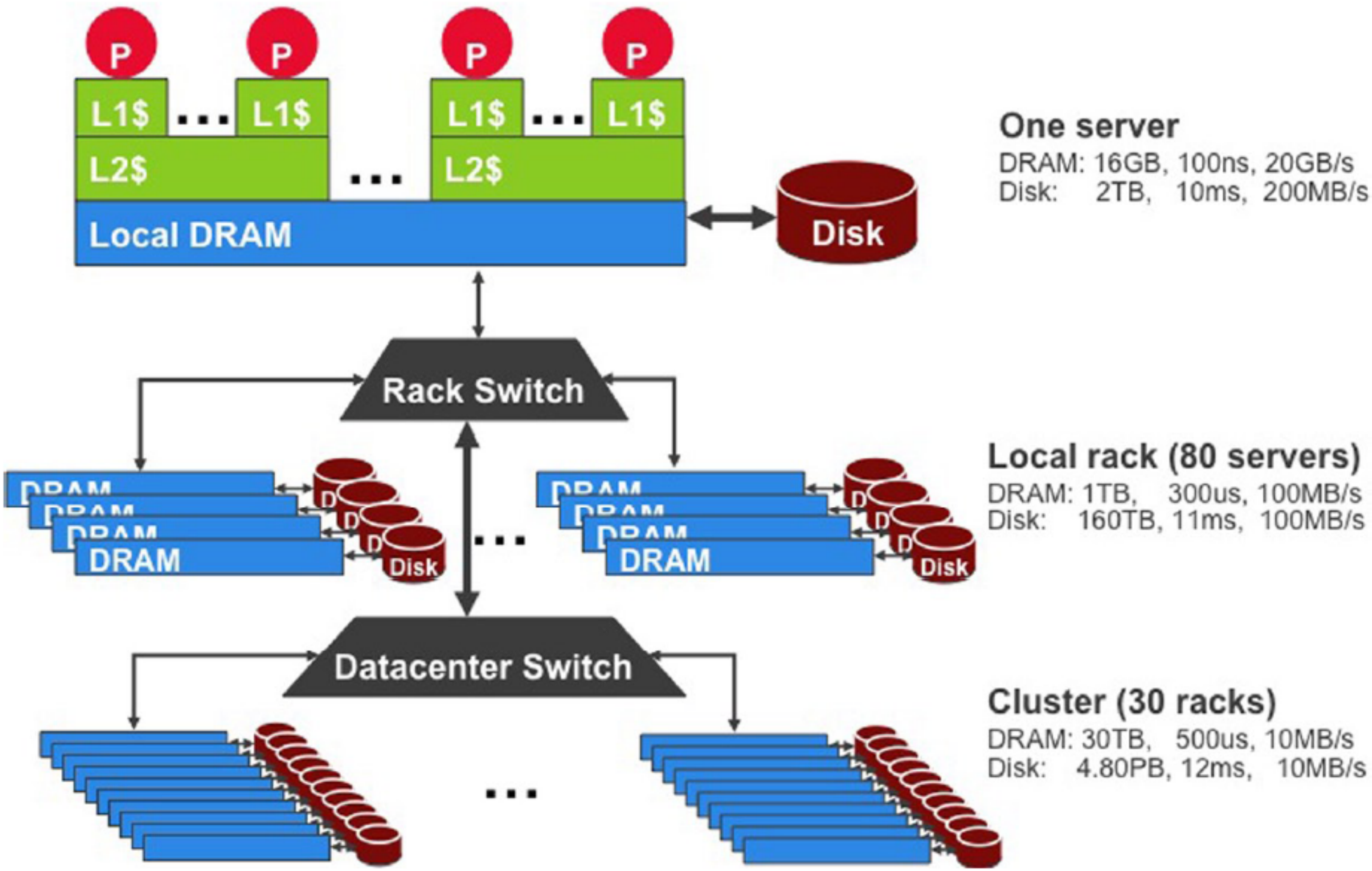
Building Blocks





Source: Barroso and Urs Hölzle (2009)

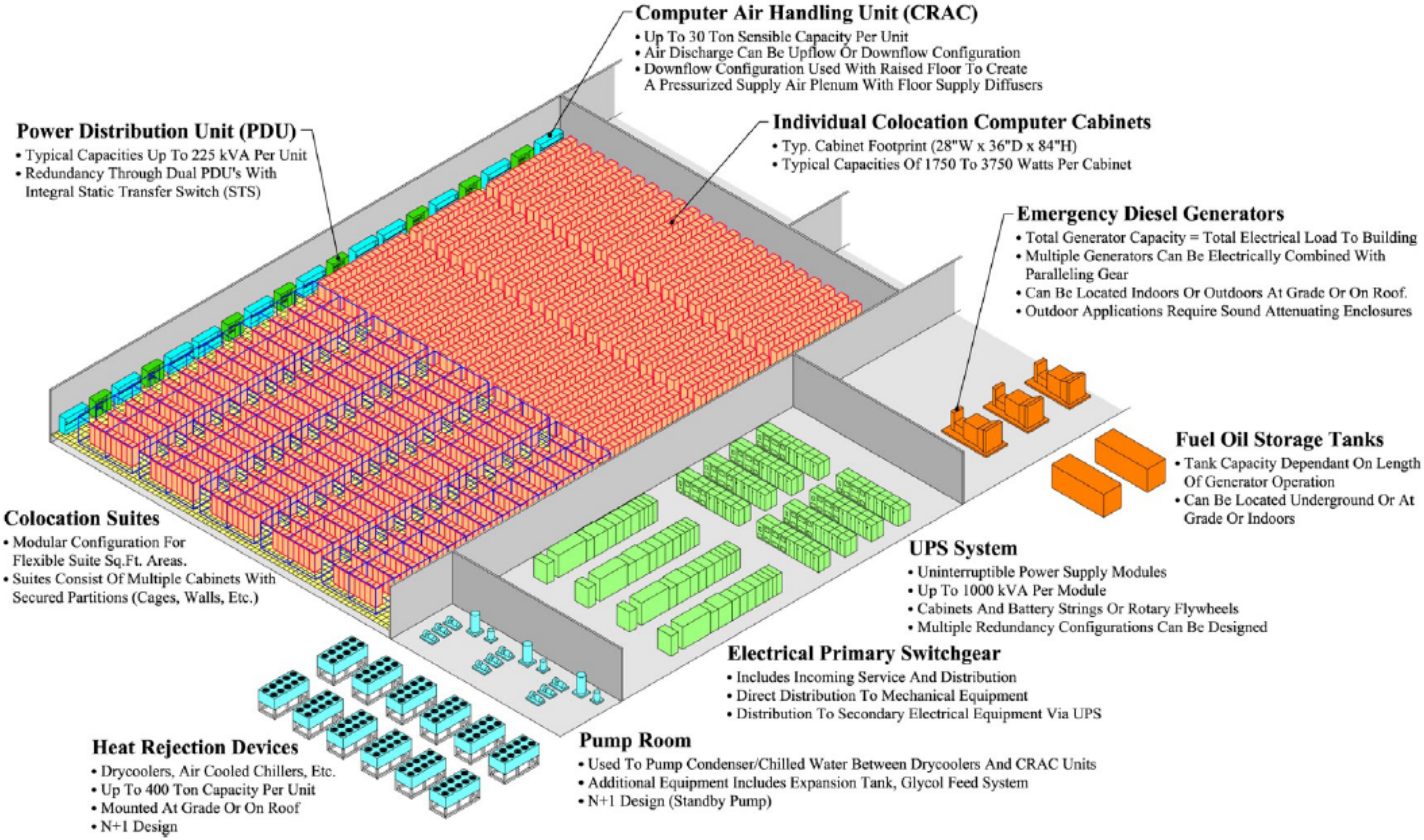
Storage Hierarchy



Funny story about sense of scale…

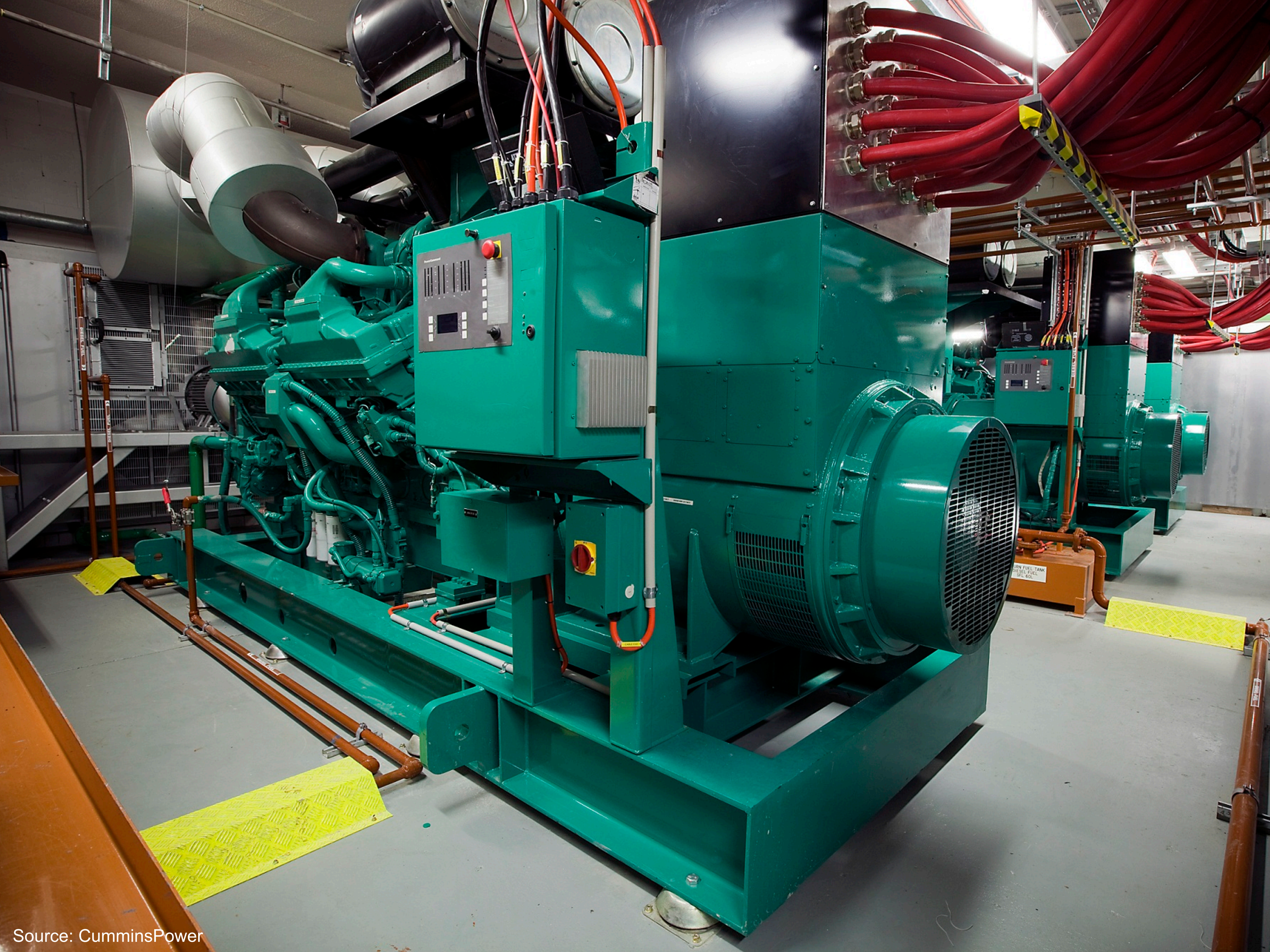
Source: Barroso and Urs Hölzle (2009)

Anatomy of a Datacenter



Source: Barroso and Urs Hölzle (2009)

Source: Google

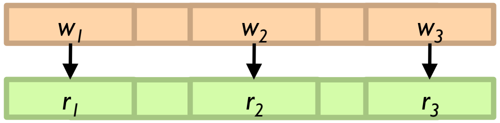

Source: CumminsPower


Source: Google

Source: Wikipedia (Everest)

Divide et impera

- Chop problem into smaller parts
- Combine partial results



Source: Wikiedia (Forest)

Synchronization Challenges

- How to split large chunks up into smaller ones
- How to integrate results from each chunk
- How to distribute shared data
- How to update shared data
- How to coordinate access to shared resources
- How to schedule different processing chunks
- How to cope of machine failure

Source: Ricardo Guimarães Herrmann

Typical Large-Data Problem

- Iterate over a large number of records
- Extract something of interest from each
- Shuffle and sort intermediate results
- Aggregate intermediate results
- Generate final output

Map

Reduce

(Dean and Ghemawat, OSDI 2004)

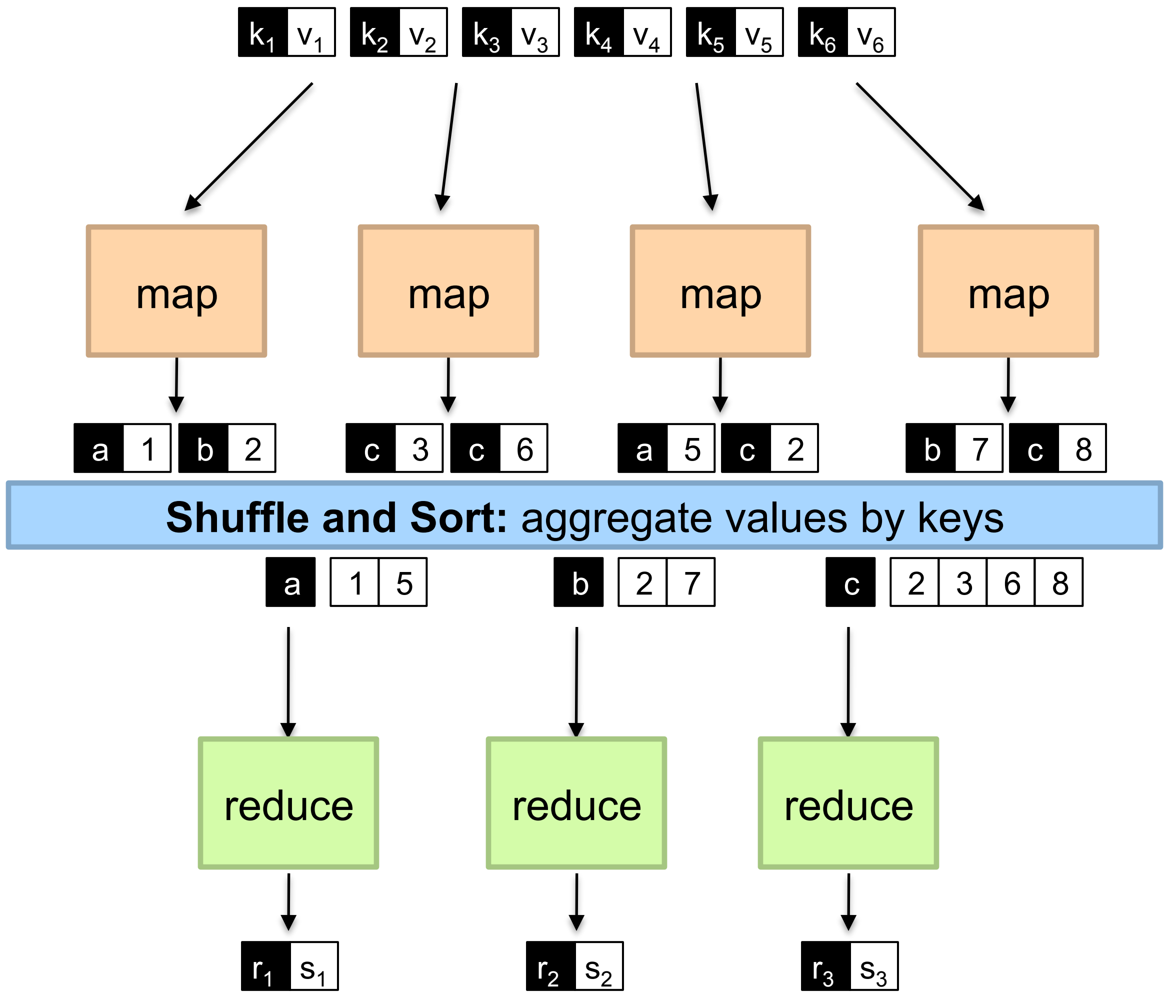
MapReduce

- Programmers specify two functions:

  map (k, v) → <k', v'>*

  reduce (k', v') → <k', v'>*

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



k1 v1
k2 v2
k3 v3
k4 v4
k5 v5
k6 v6
map
map
map
map
a 1 b 2
c 3 c 6
a 5 c 2
b 7 c 8
Shuffle and Sort: aggregate values by keys
a 1 5
b 2 7
c 2 3 6 8
reduce
reduce
reduce
r1 s1
r2 s2
r3 s3

MapReduce

- Programmers specify two functions:

  map (k, v) → <k', v'>*

  reduce (k', v') → <k', v'>*

  - All values with the same key are sent to the same reducer

- The execution framework handles everything else…

What's "everything else"?

MapReduce “Runtime”

- Handles scheduling
  - Assigns workers to map and reduce tasks
- Handles “data distribution”
  - Moves processes to data
- Handles synchronization
  - Gathers, sorts, and shuffles intermediate data
- Handles errors and faults
  - Detects worker failures and restarts

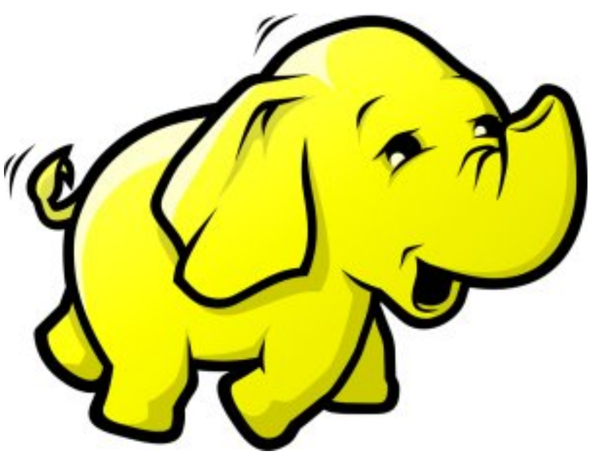
MapReduce Word Count

```
Map(String docid, String text):
    for each word w in text:
        Emit(w, 1);

Reduce(String term, Iterator<Int> values):
    int sum = 0;
    for each v in values:
        sum += v;
        Emit(term, value);
```

MapReduce Implementations

- Google has a proprietary implementation
- Hadoop is an open-source implementation in Java
    - Originally developed by Yahoo, now an Apache project
    - Center of a rapidly expanding software ecosystem



Now you know…

- Cloud computing
- Big data
- Relationship between the two
- Challenges with big data processing
- MapReduce/Hadoop


Questions?
Source: Wikipedia (Clouds)