IEMS5709
Advanced Topics in Information Processing: Big Data Systems and Information Processing

Overview: The Era of Big Data

Prof. Wing C. Lau
Department of Information Engineering
wclau@ie.cuhk.edu.hk

Acknowledgements

- The slides used in this chapter are adapted from the following sources:
  - "Data-Intensive Information Processing Applications," by Jimmy Lin, University of Maryland.

  - CS246 Mining Massive Data-sets, by Jure Leskovec, Stanford University.
  - Stat 260 Scalable Machine Learning of UC Berkeley, by Alex Smola, CMU.
  - 10-605 Machine Learning from Big Datasets, by William Cohen, CMU.
  - "Intro To Hadoop" in UCBerkeley i291 - Analyzing BigData with Twitter, by Bill Graham, Twitter.

What is this course about?

- Computing Infrastructure for Data-intensive Information Processing and Analytics
- Focus on the System Aspects of Big Data Processing:
  - Big Data Computing Infrastructure
  - The Big Data Processing Software Stack
  - Mainstream Parallel and Distributed Programming Models,
  - Their corresponding Platforms/ Frameworks and
  - Additional Operational/Programming Tools

  for Big Data Processing/Analytics in the Real World
- This course is NOT about the Design/ Analysis of Machine Learning or Data Mining algorithms
  - For this, you should consider to take CSCI5510 Big Data Analytics instead (or in parallel)
  - This course is complementary to CSCI5510

Course Administrivia

Course Pre-requisites

- Strong Programming and Hands-on Software Development and Operating System Management/ Configuration Skills
- No previous experience necessary in
  - MapReduce or other
  - Parallel and distributed programming

  BUT we expect you to pick them up quickly

This course is not for you…

- If you're not genuinely interested in the topic
- If you can't put in the time
- If you're not ready to do a lot of work
- If you're not open to thinking about computing in new ways
- If you can't cope with the uncertainty, unpredictability, etc. that comes with bleeding edge software

Otherwise, this will be a richly rewarding course!

Computing/ Cloud Resources

- Hadoop on your local machine
- Hadoop in a Virtual Machine on your local machine
- Sign-up for Freebie (limited-time) Trial accounts from Commercial Cloud Computing Services:
  - Amazon Web Service (AWS), Microsoft Azure, Google Compute Engine
- The IE DIC (Data-Intensive Cluster) with:
  - Real-World Datasets ;
  - Homeworks will require each student to setup various Big Data Processing systems over the cluster and use them to solve different Parallel/ Distributed Programming tasks.

Source: Wikipedia (Japanese rock garden)

Zen

- We will be using bleeding edge technologies (= immature!)
  - Bugs, undocumented features, inexplicable behavior
  - Data loss(!)
- Don't get frustrated (take a deep breath)…
  - Those W$*#T@F! moments
- Be patient…
  - We will inevitably encounter "situations" along the way
- Be flexible…
  - We will have to be creative in workarounds
- Be constructive…
  - Tell me how I can make everyone's experience better

Web-Scale, Big Data

Source: Wikipedia (Everest)

Why Should We Care about Big Data ?

- Ready-made large-data problems
  - Lots of user-generated content
  - Even more user behavior data
  - Examples: Facebook friend suggestions, Google ad placement
  - Business intelligence: gather everything in a data warehouse and run analytics to generate insight
- Utility computing
  - Provision Hadoop/MapReduce clusters on-demand in the cloud
  - Lower barrier to entry for tackling large-data problem
  - Commoditization and democratization of large-data capabilities

How Big is Big ?

- 2008: Google processes 20 PetaByte per Day (Peta=10^{15})
- Apr 2009: Facebook has 2.5 PB user data + 15 TB/day
- May 2009: eBay has 6.5 PB user data + 50 TB/day
- 2011: Yahoo! Has 180-200 PB of data
- 2012: Facebook ingests 500TB/day

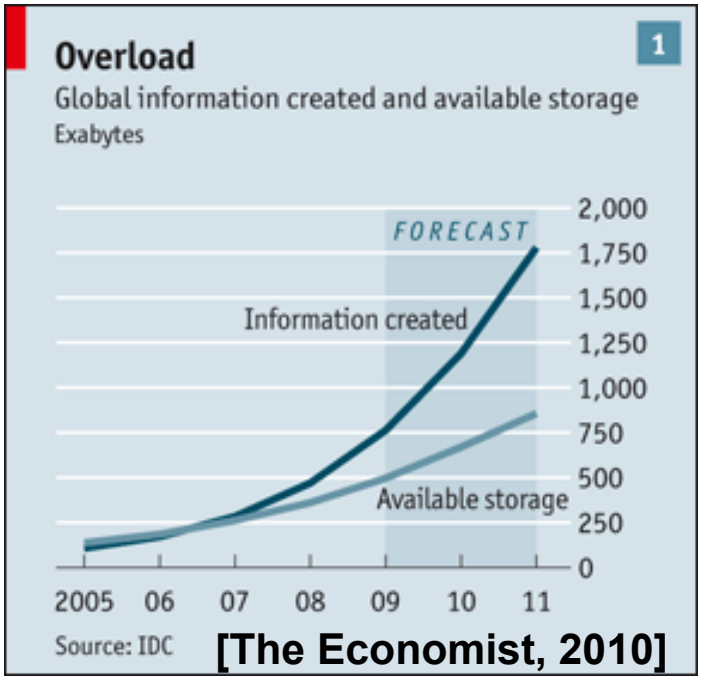
How many users and objects “recently” ?

- Flickr has >6 billion photos
- Facebook has 1.15 billion active users
- Google is serving >1.2 billion queries/day on more than 27 billion items
- >2 billion videos/day watched on YouTube

How much data?

- Modern applications use massive data:
  - Rendering 'Avatar' movie required >1 petabyte of storage
  - eBay has >6.5 petabytes of user data
  - CERN's LHC will produce about 15 petabytes of data per year
  - In 2008, Google processed 20 petabytes per day
  - German Climate computing center dimensioned for 60 petabytes of climate data
  - Someone estimated in 2013 that Google had 10 exabytes on disk and ~ 5 exabytes on tape backup
  - NSA Utah Data Center is said to have 5 zettabyte (!)
- How much is a zettabyte?
  - 1,000,000,000,000,000,000,000 bytes
  - A stack of 1TB hard disks that is 25,400 km high

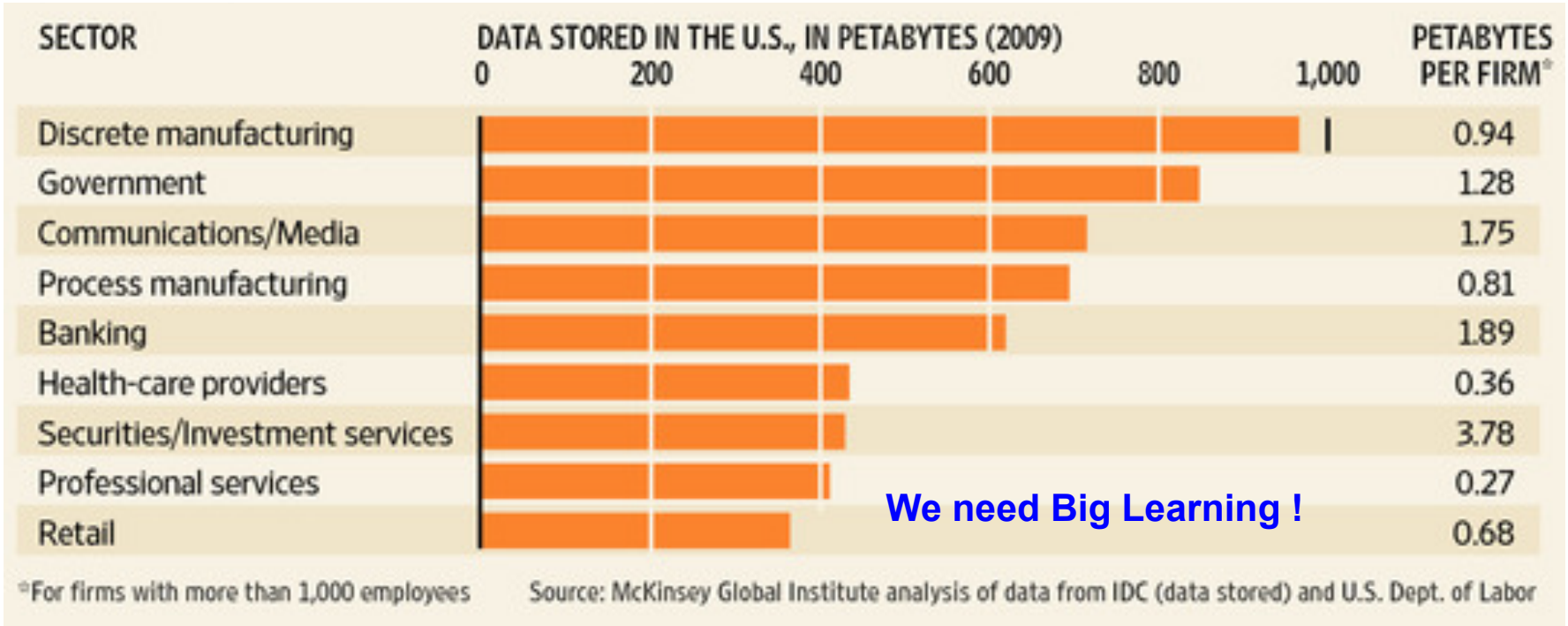
How Big is Big ?



[The Economist, 2010]

We are producing more data than we are able to store!

http://en.wikipedia.org/wiki/Zettabyte



How much computation?

- No single computer can process that much data
  - Need many computers!
- How many computers do modern services need?
  - Facebook is thought to have more than 60,000 servers
  - 1&1 Internet has over 70,000 servers
  - Akamai has 95,000 servers in 71 countries
  - Intel has ~100,000 servers in 97 data centers
  - Microsoft reportedly had at least 200,000 servers in 2008
  - Google is thought to have more than 1 million servers, is planning for 10 million (according to Jeff Dean)

Data - User generated content

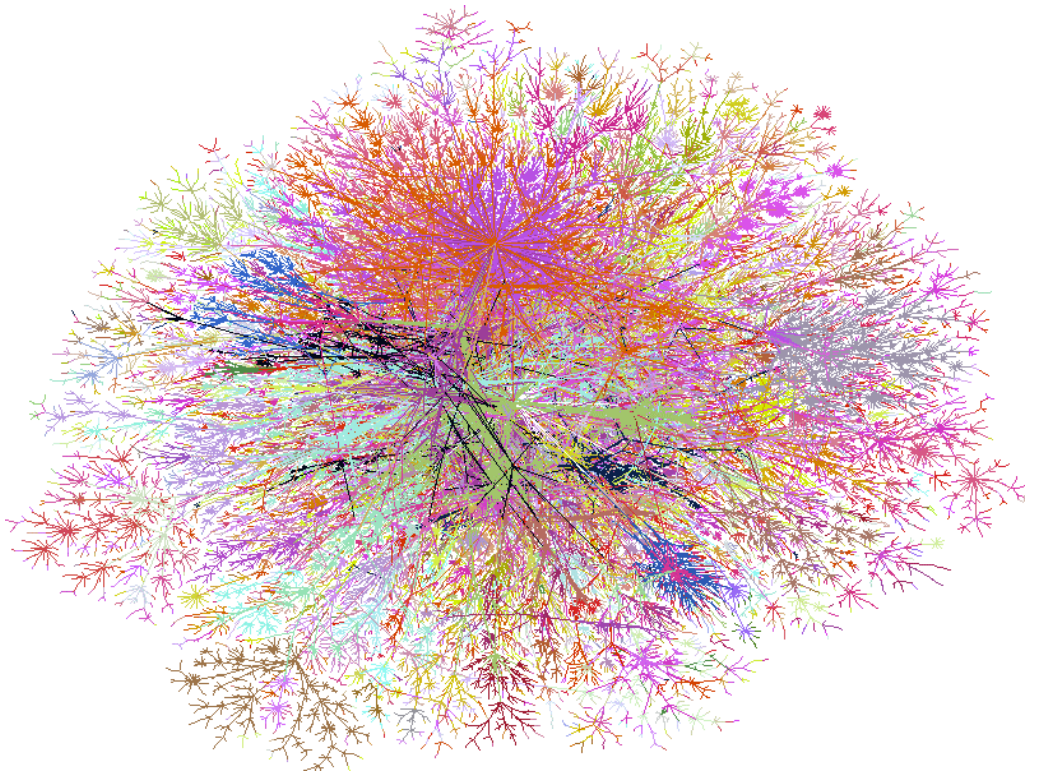
- Webpages (content, graph)
- Clicks (ad, page, social)
- Users (OpenID, FB Connect)
- e-mails (Hotmail, Y!Mail, Gmail)
- Photos, Movies (Flickr, YouTube, Vimeo ...)
- Cookies / tracking info (see Silverlight)
- Installed apps (Android market etc.)
- Location (Latitude, Loopt, Foursquared)
- User generated content (Wikipedia & co)
- Ads (display, text, DoubleClick, Yahoo)
- Comments (Disqus, Facebook)
- Reviews (Yelp, Y!Local)
- Third party features (e.g. Experian)
- Social connections (LinkedIn, Facebook)
- Purchase decisions (Netflix, Amazon)
- Instant Messages (YIM, Skype, Gtalk)
- Search terms (Google, Bing)
- Timestamp (everything)
- News articles (BBC, NYTimes, Y!News)
- Blog posts (Tumblr, Wordpress)
- Microblogs (Twitter, Jaiku, Meme)
- Link sharing (Facebook, Delicious, Buzz)
- Network traffic

crawl it

>1B images, 40h video/minute

Web-Scale Big Data

- Webpages (content, graph)
- Clicks (ad, page, social)
- Users (OpenID, FB Connect)
- e-mails (Hotmail, Y!Mail, Gmail)
- Photos, Movies (Flickr, YouTube, Vimeo ...)
- Cookies / tracking info (see Ghostery)
- Installed apps (Android market etc.)
- Location (Latitude, Loopt, Foursquared)
- User generated content (Wikipedia & co)
- Ads (display, text, DoubleClick, Yahoo)
- Comments (Disqus, Facebook)
- Reviews (Yelp, Y!Local)
- Third party features (e.g. Experian)
- Social connections (LinkedIn, Facebook)
- Purchase decisions (Netflix, Amazon)
- Instant Messages (YIM, Skype, Gtalk)
- Search terms (Google, Bing)
- Timestamp (everything)
- News articles (BBC, NYTimes, Y!News)
- Blog posts (Tumblr, Wordpress)
- Microblogs (Twitter, Jaiku, Meme)
- Link sharing (Facebook, Delicious, Buzz)
- Network traffic



>10B useful webpages

Crawling the Web for US$100k/month

- Webpages (content, graph)
- Clicks (ad, page, social)
- Users (OpenID, FB Connect)
- e-mails (Hotmail, Y!Mail, Gmail)
- Photos, Movies (Flickr, YouTube, Vimeo ...)
- Cookies / tracking info (see Ghostery)
- Installed apps (Android market etc.)
- Location (Latitude, Loopt, Foursquared)
- User generated content (Wikipedia & co)
- Ads (display, text, DoubleClick, Yahoo)
- Comments (Disqus, Facebook)
- Reviews (Yelp, Y!Local)
- Third party features (e.g. Experian)
- Social connections (LinkedIn, Facebook)
- Purchase decisions (Netflix, Amazon)
- Instant Messages (YIM, Skype, Gtalk)
- Search terms (Google, Bing)
- Timestamp (everything)
- News articles (BBC, NYTimes, Y!News)
- Blog posts (Tumblr, Wordpress)
- Microblogs (Twitter, Jaiku, Meme)
- Link sharing (Facebook, Delicious, Buzz)
- Network traffic

- **10 billion pages**
  (this is a small subset, maybe 10%)
  10k/page = 100TB
  ($10k for disks or EBS 1 month)
- **1000 machines**
  10ms/page = 1 day
  afford 1-10 MIP/page
  ($20k on EC2 for 0.68$/h)
- **10 Gbps link**
  ($10k/month via ISP or EC2)
  - **Should only need 1 day to Tx the 100TB raw data over a 10Gbps link**
  - **BUT need to wait for web-server to respond (est. latency of 300ms/page) roundtrip**
  - **?? Need 1000 servers to collect the 100TB data in parallel for 1 month ??**
    ($70k on EC2 for 0.085$/h)

Rough estimate by Alex Smola (CMU)

Data - Identity & Graph

- Webpages (content, graph)
- Clicks (ad, page, social)
- Users (OpenID, FB Connect)
- e-mails (Hotmail, Y!Mail, Gmail)
- Photos, Movies (Flickr, YouTube, Vimeo ...)
- Cookies / tracking info (see Ghostery)
- Installed apps (Android market etc.)
- Location (Latitude, Loopt, Foursquared)
- User generated content (Wikipedia & co)
- Ads (display, text, DoubleClick, Yahoo)
- Comments (Disqus, Facebook)
- Reviews (Yelp, Y!Local)
- Third party features (e.g. Experian)
- Social connections (LinkedIn, Facebook)
- Purchase decisions (Netflix, Amazon)
- Instant Messages (YIM, Skype, Gtalk)
- Search terms (Google, Bing)
- Timestamp (everything)
- News articles (BBC, NYTimes, Y!News)
- Blog posts (Tumblr, Wordpress)
- Microblogs (Twitter, Jaiku, Meme)
- Link sharing (Facebook, Delicious, Buzz)
- Network traffic

100M-1B vertices

Crawling Twitter for $10k

- Webpages (content, graph)
- Clicks (ad, page, social)
- Users (OpenID, FB Connect)
- e-mails (Hotmail, Y!Mail, Gmail)
- Photos, Movies (Flickr, YouTube, Vimeo ...)
- Cookies / tracking info (see Ghostery)
- Installed apps (Android market etc.)
- Location (Latitude, Loopt, Foursquared)
- User generated content (Wikipedia & co)
- Ads (display, text, DoubleClick, Yahoo)
- Comments (Disqus, Facebook)
- Reviews (Yelp, Y!Local)
- Third party features (e.g. Experian)
- Social connections (LinkedIn, Facebook)
- Purchase decisions (Netflix, Amazon)
- Instant Messages (YIM, Skype, Gtalk)
- Search terms (Google, Bing)
- Timestamp (everything)
- News articles (BBC, NYTimes, Y!News)
- Blog posts (Tumblr, Wordpress)
- Microblogs (Twitter, Jaiku, Meme)
- Link sharing (Facebook, Delicious, Buzz)
- Network traffic

- 307M active users (3Q2015)
- Used to be Per user 300 queries/h
- 100 edges/query
- 100 edges/account
- Need 100 machines for 2 weeks (crawl it at 10 queries/s)
  - Tweets
  - Inlinks
  - Outlinks
- Cost
  - $3k for computers on EC2
  - Similar for network & storage
  - Need 10k user keys

Recently Rate Limit lowered to 15 queries/ 15 min !!

Data - Messages

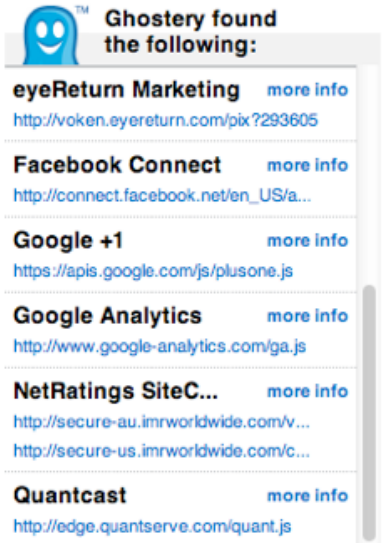
- Webpages (content, graph)
- Clicks (ad, page, social)
- Users (OpenID, FB Connect)
- e-mails (Hotmail, Y!Mail, Gmail)
- Photos, Movies (Flickr, YouTube, Vimeo ...)
- Cookies / tracking info (see Ghostery)
- Installed apps (Android market etc.)
- Location (Latitude, Loopt, Foursquared)
- User generated content (Wikipedia & co)
- Ads (display, text, DoubleClick, Yahoo)
- Comments (Disqus, Facebook)
- Reviews (Yelp, Y!Local)
- Third party features (e.g. Experian)
- Social connections (LinkedIn, Facebook)
- Purchase decisions (Netflix, Amazon)
- Instant Messages (YIM, Skype, Gtalk)
- Search terms (Google, Bing)
- Timestamp (everything)
- News articles (BBC, NYTimes, Y!News)
- Blog posts (Tumblr, Wordpress)
- Microblogs (Twitter, Jaiku, Meme)
- Link sharing (Facebook, Delicious, Buzz)
- Network traffic

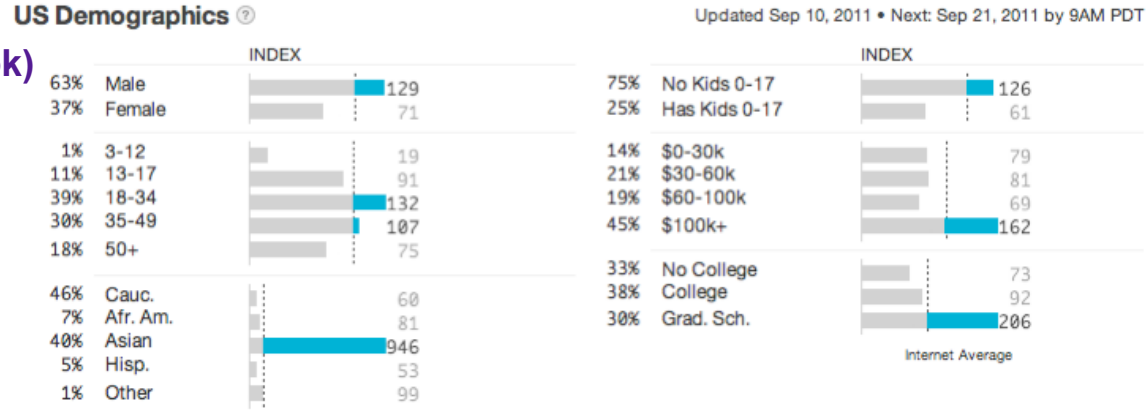
>1B texts

impossible without NDA

Data - User Tracking

- Webpages (content, graph)
- Clicks (ad, page, social)
- Users (OpenID, FB Connect)
- e-mails (Hotmail, Y!Mail, Gmail)
- Photos, Movies (Flickr, YouTube, Vimeo ...)
- Cookies / tracking info (see Ghostery)
- Installed apps (Android market etc.)
- Location (Latitude, Loopt, Foursquared)
- User generated content (Wikipedia & co)
- Ads (display, text, DoubleClick, Yahoo)
- Comments (Disqus, Facebook)
- Reviews (Yelp, Y!Local)
- Third party features (e.g. Experian)
- Social connections (LinkedIn, Facebook)
- Purchase decisions (Netflix, Amazon)
- Instant Messages (YIM, Skype, Gtalk)
- Search terms (Google, Bing)
- Timestamp (everything)
- News articles (BBC, NYTimes, Y!News)
- Blog posts (Tumblr, Wordpress)
- Microblogs (Twitter, Jaiku, Meme)
- Link sharing (Facebook, Delicious, Buzz)
- Network traffic

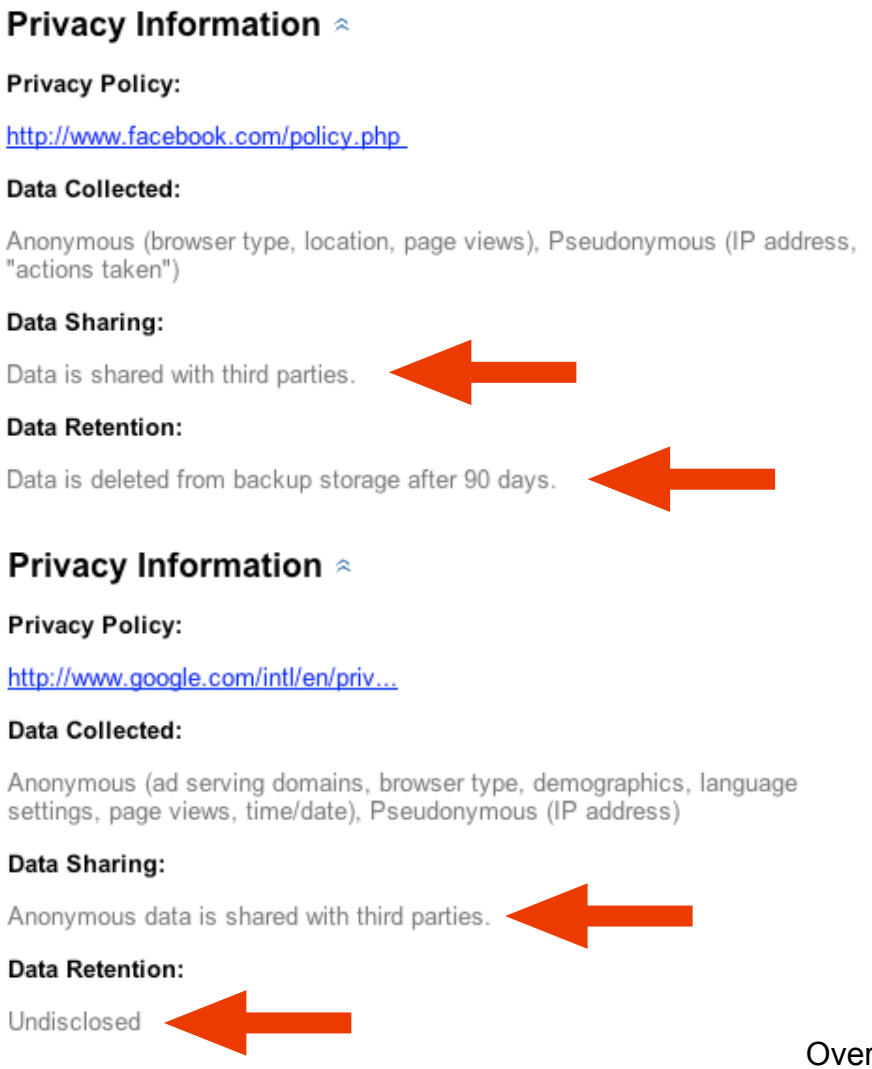




>1B 'identities'

Data - User Tracking

- Webpages (content, graph)
- Clicks (ad, page, social)
- Users (OpenID, FB Connect)
- e-mails (Hotmail, Y!Mail, Gmail)
- Photos, Movies (Flickr, YouTube, Vimeo ...)
- Cookies / tracking info (see Ghostery)
- Installed apps (Android market etc.)
- Location (Latitude, Loopt, Foursquared)
- User generated content (Wikipedia & co)
- Ads (display, text, DoubleClick, Yahoo)
- Comments (Disqus, Facebook)
- Reviews (Yelp, Y!Local)
- Third party features (e.g. Experian)
- Social connections (LinkedIn, Facebook)
- Purchase decisions (Netflix, Amazon)
- Instant Messages (YIM, Skype, Gtalk)
- Search terms (Google, Bing)
- Timestamp (everything)
- News articles (BBC, NYTimes, Y!News)
- Blog posts (Tumblr, Wordpress)
- Microblogs (Twitter, Jaiku, Meme)
- Link sharing (Facebook, Delicious, Buzz)
- Network traffic



Privacy Information

Privacy Policy:

http://www.facebook.com/policy.php

Data Collected:

Anonymous (browser type, location, page views), Pseudonymous (IP address, "actions taken")

Data Sharing:

Data is shared with third parties.

Data Retention:

Data is deleted from backup storage after 90 days.

Privacy Information

Privacy Policy:

http://www.google.com/intl/en/priv...

Data Collected:

Anonymous (ad serving domains, browser type, demographics, language settings, page views, time/date), Pseudonymous (IP address)

Data Sharing:

Anonymous data is shared with third parties.

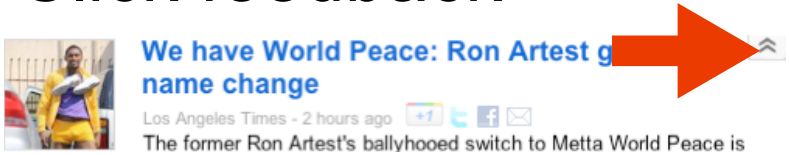
Data Retention:

Undisclosed

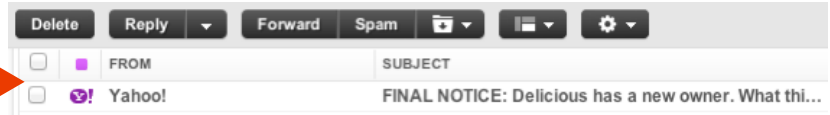
(Implicitly) Labelled Data

- Ads

- Click feedback



- Emails

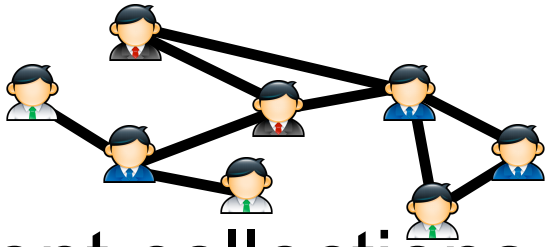


- Tags

- Editorial (Manually Labelled) data is very expensive! Do not use!

Vs. Un-Labelled Data

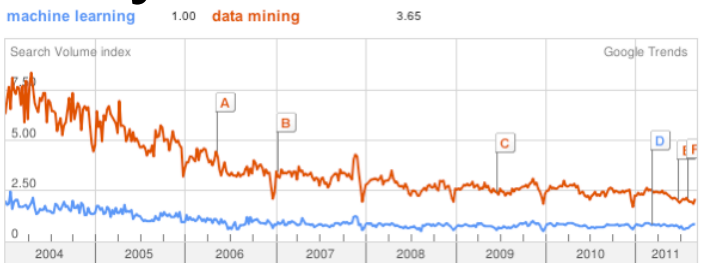
- Graphs



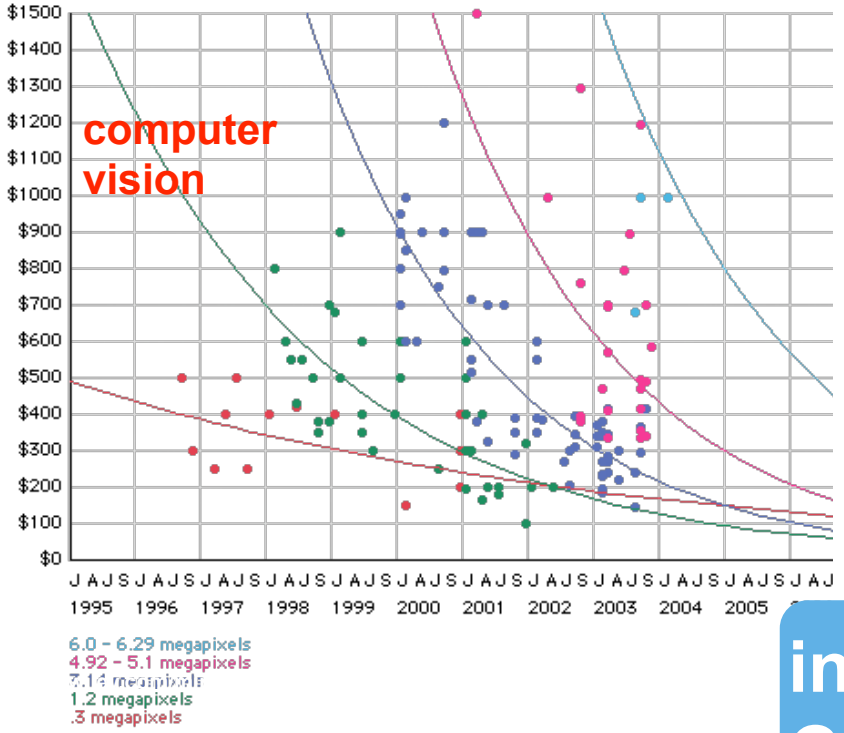
- Document collections

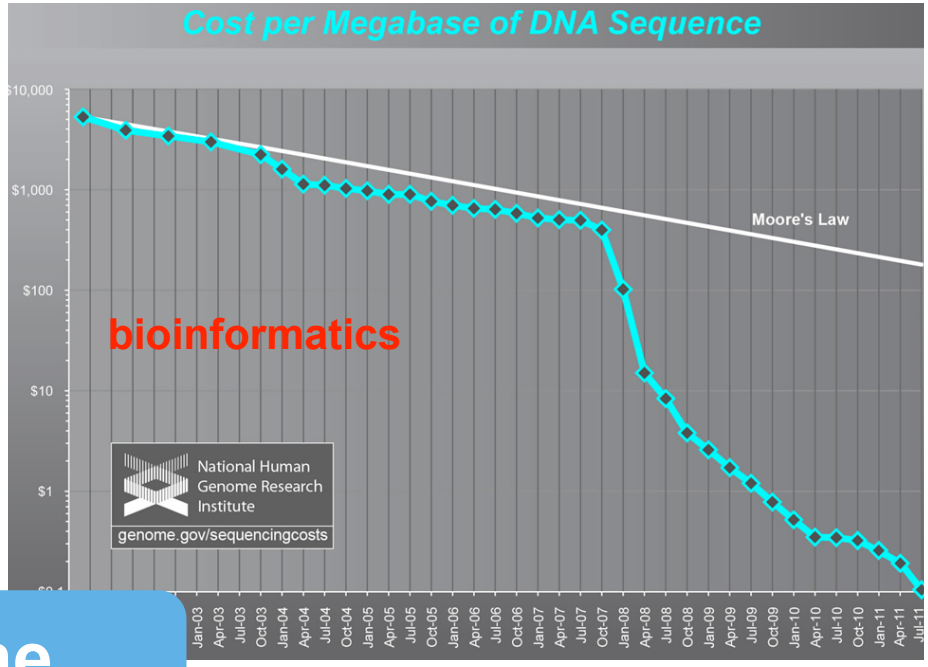
- Email/IM/Discussions

- Query stream



Many more sources of Data

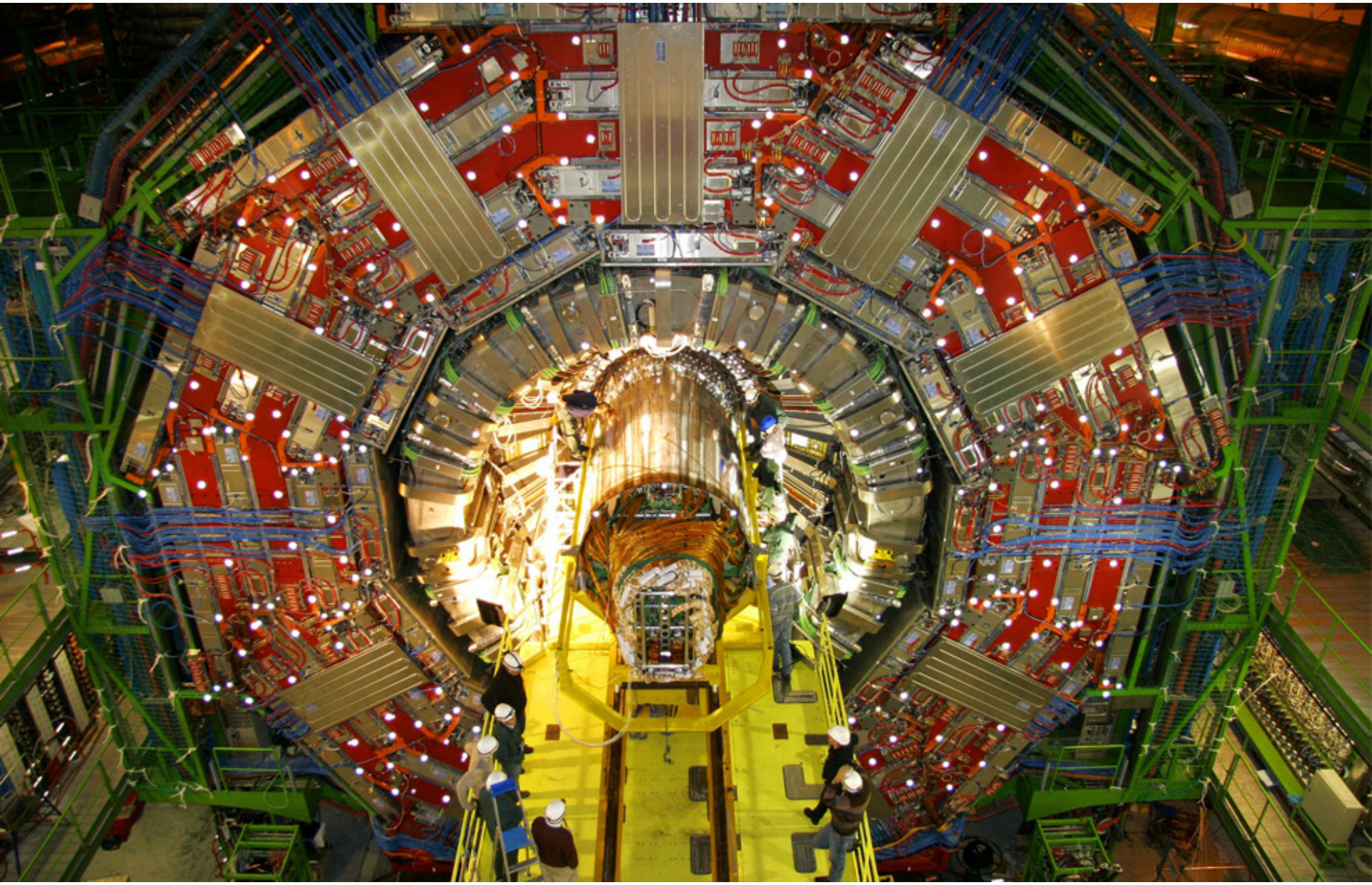




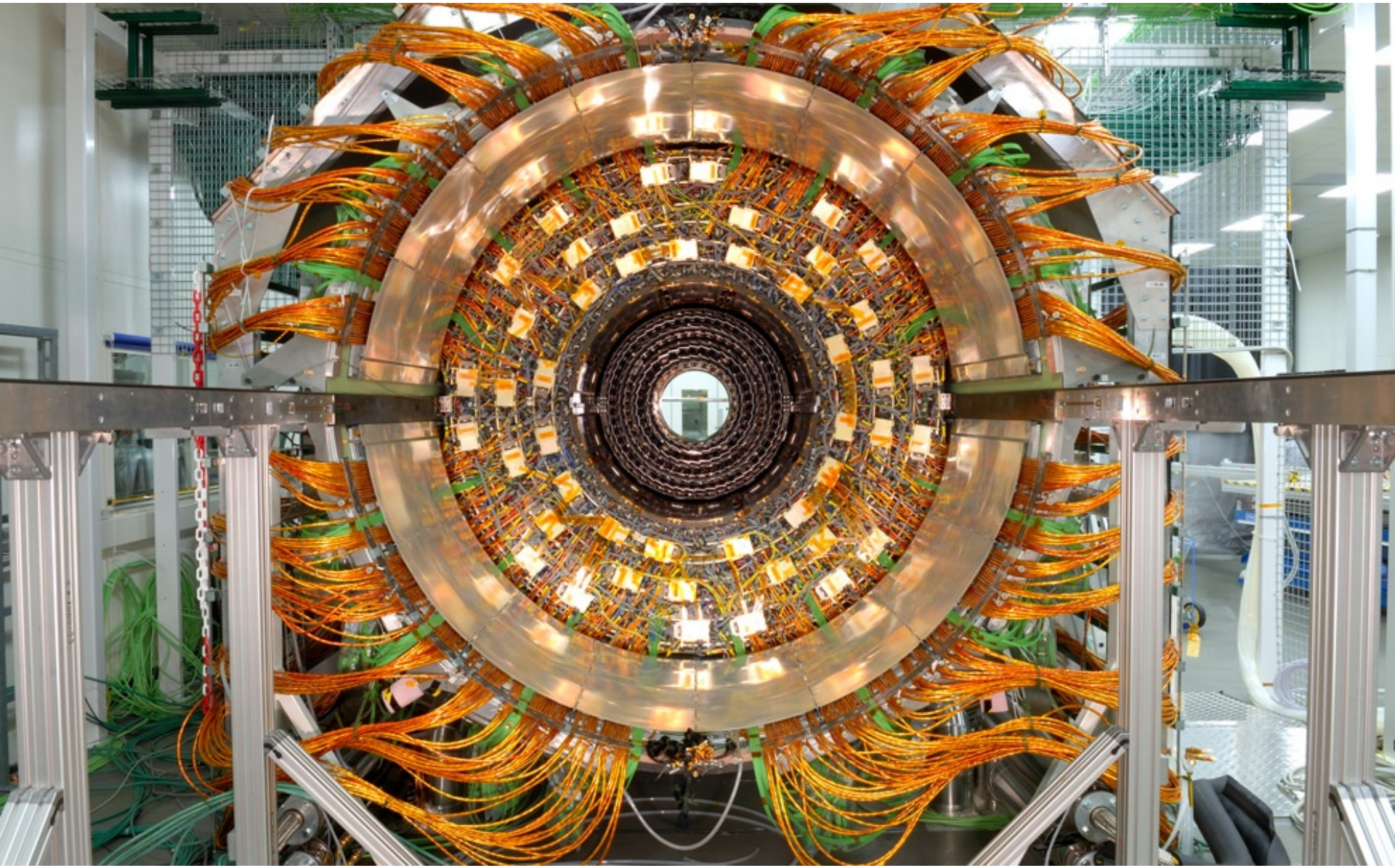
in the Cloud

Personalized Sensors

Ubiquitous Control



CERN Large Hadron Collider (LHC) will generate 15 PB/ yr (??)

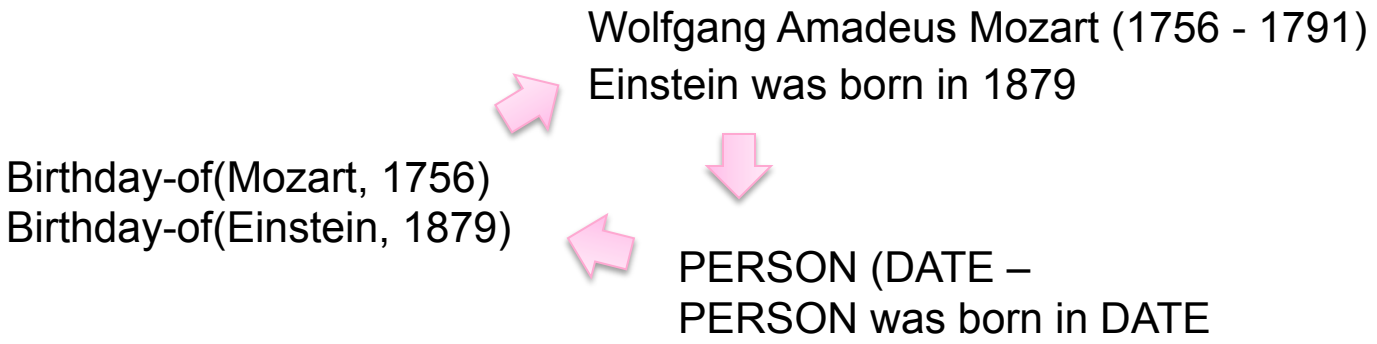


What to do with More Data ?

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

Who shot Abraham Lincoln? --> ??? shot Abraham Lincoln

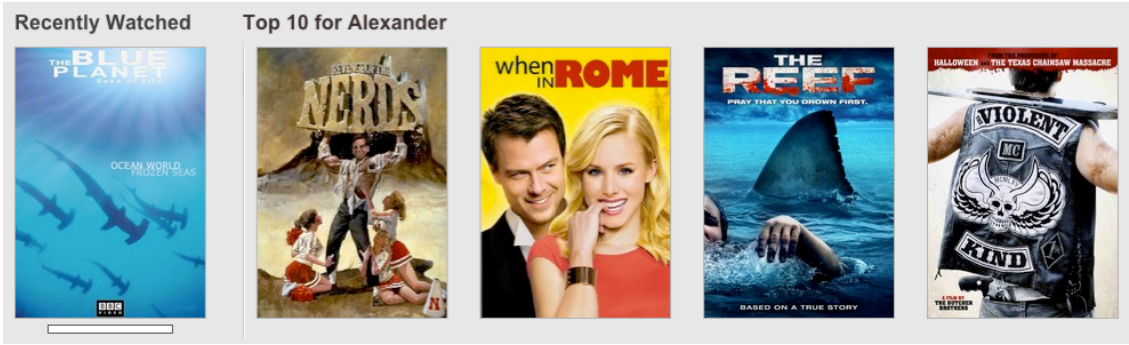
- Learning relations
  - Start with seed instances
  - Search for patterns on the Web
  - Using patterns to find more instances

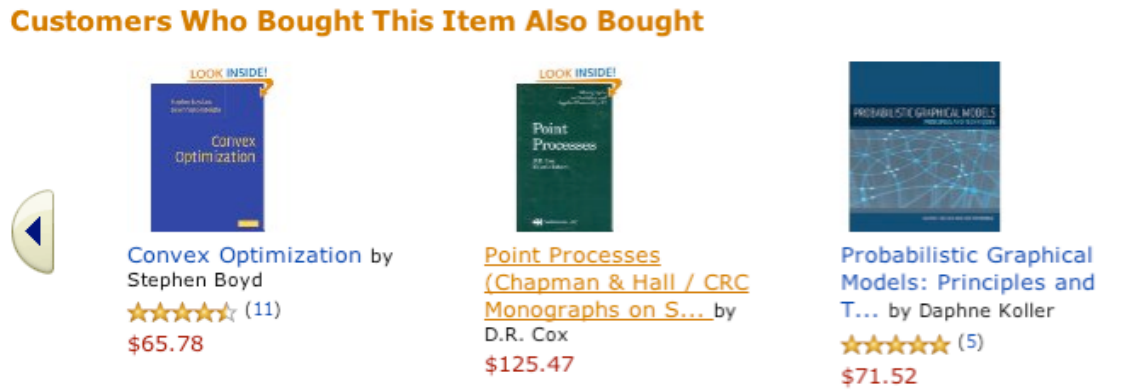


(Brill et al., TREC 2001; Lin, ACM TOIS 2007)
(Agichtein and Gravano, DL 2000; Ravichandran and Hovy, ACL 2002; …)

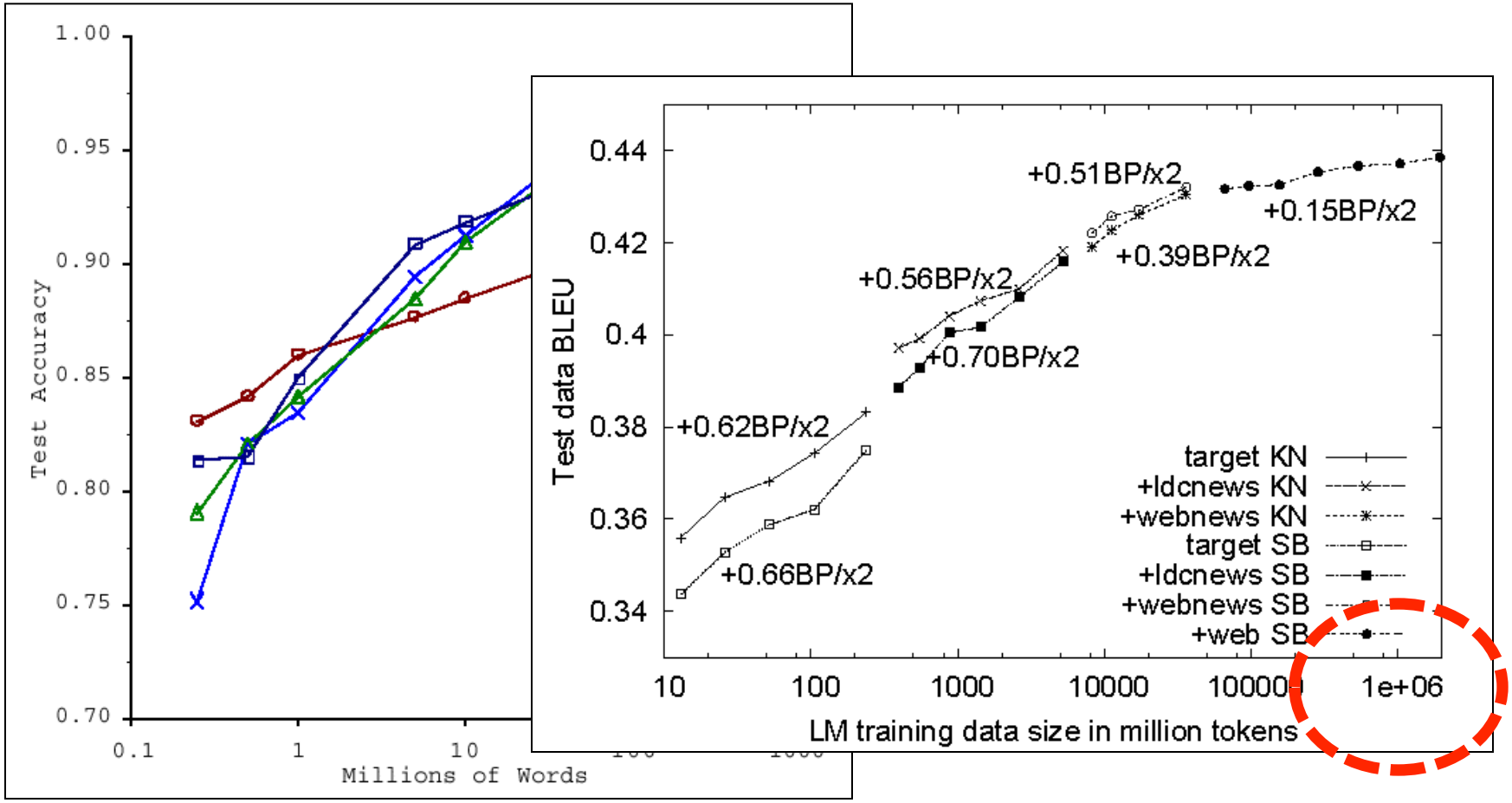
What to do with More Data ? (cont'd)
Personalization

- 100-1000M users
  - Spam filtering
  - Personalized targeting & collaborative filtering
  - News recommendation
  - Advertising





There's no Data like more Data!



How do we get here if we're not Google?

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

By 2001, we have learned that, for many tasks, there's no real *substitute* for using lots of data

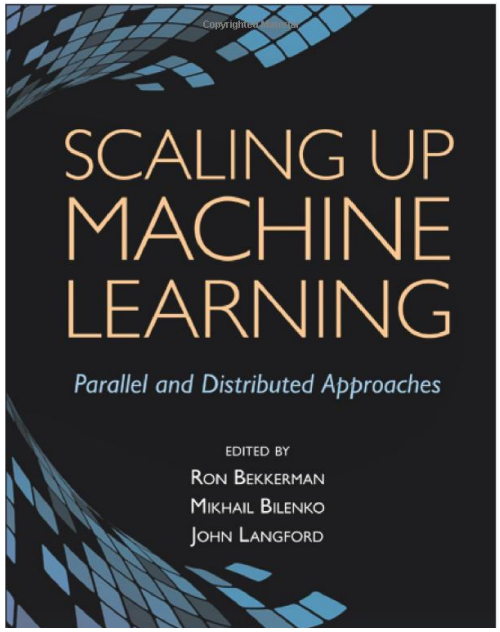
…and in 2009

Eugene Wigner's article "The Unreasonable Effectiveness of Mathematics in the Natural Sciences" examines why so much of physics can be neatly explained with simple mathematical formulas such as $f = ma$ or $e = mc^2$. Meanwhile, sciences that involve human beings rather than elementary particles have proven more resistant to elegant mathematics. Economists suffer from physics envy over their inability to neatly model human behavior. An informal, incomplete grammar of the English language runs over 1,700 pages.

Perhaps when it comes to natural language processing and related fields, we're doomed to complex theories that will never have the elegance of physics equations. But if that's so, we should stop acting as if our goal is to author extremely elegant theories, and instead embrace complexity and make use of the best ally we have: the unreasonable effectiveness of data.

Norvig, Pereira, Halevy, "The Unreasonable Effectiveness of Data", 2009

…and in 2012



Arthur Gretton, Michael Mahoney, Mehryar Mohri, Ameet Talwalkar

Gatsby Unit, UCL; Stanford; Google Research; UC Berkeley

Workshop: Low-rank Methods for Large-scale Machine Learning

7:30am - 6:30pm Saturday, December 11, 2010

Joseph Gonzalez, Sameer Singh, Graham Taylor, James Bergstra, Alice Zheng, Misha Bilenko, Yucheng Low, Yoshua Bengio, Michael Franklin, Carlos Guestrin, Andrew McCallum, Alexander Smola, Michael Jordan, Sugato Basu

Carnegie Mellon University; University of Massachusetts, Amherst; New York University; Harvard; Microsoft Research; Microsoft Research; Carnegie Mellon University; University of Montreal; UC Berkeley; Carnegie Mellon University; UMass Amherst; Yahoo! Research; University of California; Google Research

Workshop: Big Learning: Algorithms, Systems, and Tools for Learning at Scale

Location: Montebajo: Theater

SMLA Workshop 2010

29 June - 01 July, 2010, Bradford, UK

International Workshop on
Scalable Machine Learning and Applications (SMLA-10)
In conjunction with CIT 2010

What to do with More Data? (cont'd)

- User Behavior Analysis
- AB Test Analysis
- Ad Targetting
- Trending Topics
- User and Topic Modeling
- Recommendations (Collaborative Filtering)
- Predictions
- Novel Detection and More …

- Following Theory, Experiment and Simulation,

Big Data has become the 4th-Paradigm of Science

s/knowledge/data/g

Knowledge Discovery via **Scalable** Information Analytics, e.g.

Scalable Data Mining, Statistical Modeling, Machine Learning

What is Data Mining?

- Discovery of **patterns and models that are:**
  - **Valid:** hold on new data with some certainty
  - **Useful:** should be possible to act on the item
  - **Unexpected:** non-obvious to the system
  - **Understandable:** humans should be able to interpret the pattern

Data Mining Tasks

- **Descriptive Methods:** Find human-interpretable patterns that describe the data, e.g.
  - Clustering
  - Dimensionality Reduction
  - Association Rule Discovery
  - Sequential Pattern Discovery
- **Predictive Methods:** Use some variables to predict unknown or future values of other variables, e.g.
  - Classification
  - Regression
  - Novelty Detection

Then, what is Machine Learning ?

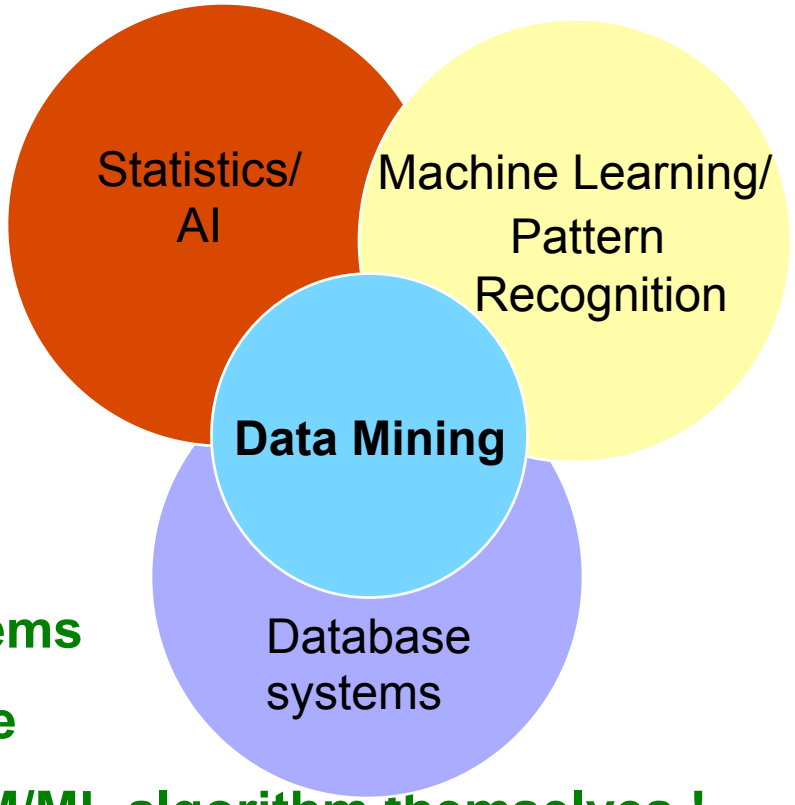
- Arthur Samuel (1959). Machine Learning: Field of study that gives computers the ability to learn without being explicitly programmed.
- Tom Mitchell (1998) Well-posed Learning Problem: A computer program is said to *learn* from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E.

Then, what is Machine Learning (cont'd) ?

- Given "a few" examples (labelled data for training), make a machine learn how to:
  - **Predict** on NEW Samples or
  - **Discover** Patterns in Data
- Major Learning Paradigms:
  - **Supervised Learning**
    - Regression (to predict a continuous output, like curve fitting)
    - Classification (to predict a class or category)
    - Ranking (to predict rank ordering)
  - **Unsupervised Learning**
    - Clustering
    - Density Estimation
    - Dimensionality Reduction
  - **There is also Semi-supervised Learning**
    - Large amount of unlabelled data + small amount of labelled ones

Different Cultures of: Data Mining, Statistics, Machine Learning

- **Data mining overlaps with:**
  - **Databases:** Large-scale data, simple queries
  - **Machine learning:** Traditionally with Small data, Complex models
  - **Statistics:** Traditionally focus on using as little data as possible to construct Predictive Models for inference
- **Different cultures:**
  - To a DB person, data mining is an extreme form of **analytic processing** – queries that examine large amounts of data
    - Result is the query answer
  - To a statistics/ML person, data-mining is the **inference of models**
    - Result is the parameters of the model

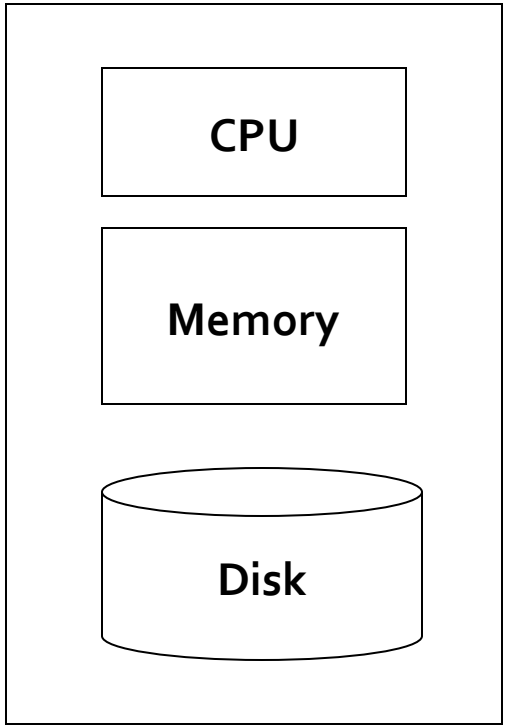


- **In this class, we will focus on the Systems and Programming Models which enable Web-scale DM and ML BUT NOT the DM/ML algorithm themselves !**

How do we scale up processing for Big Data ?

Or: How to run Algorithms on MANY REAL and FAULTY boxes ?

Single Node Architecture

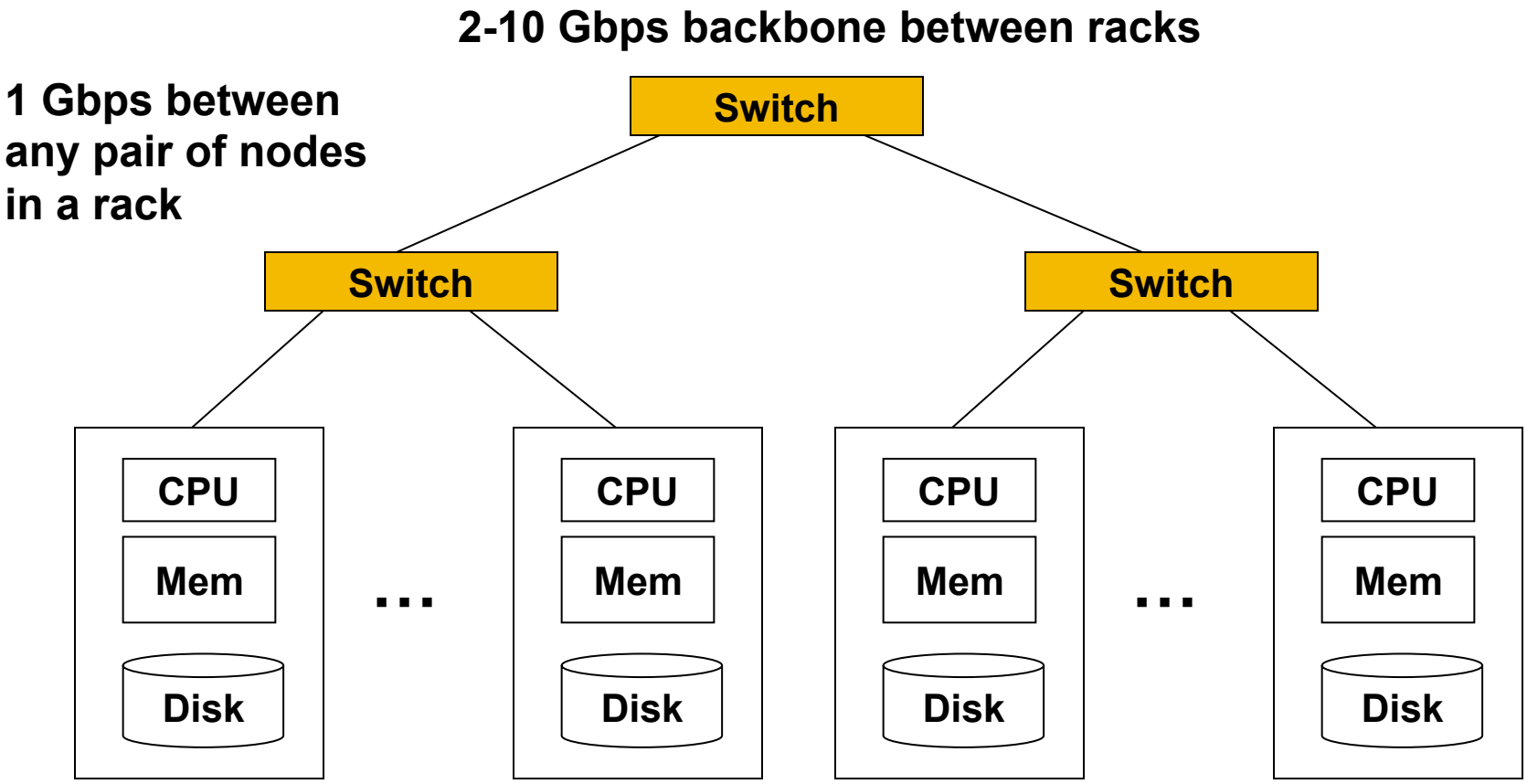


"Classical"
Machine Learning, Statistics,
Data Mining

Motivation: Google Example

- 20+ billion web pages x 20KB = 400+ TB
- 1 computer reads 30-35 MB/sec from disk
  - ~4 months to read the web
- ~1,000 hard drives to store the web
- Takes even more to **do** something useful with the data!
- **Today, a standard architecture for such problems is emerging:**
  - Cluster of commodity Linux nodes
  - Commodity network (ethernet) to connect them

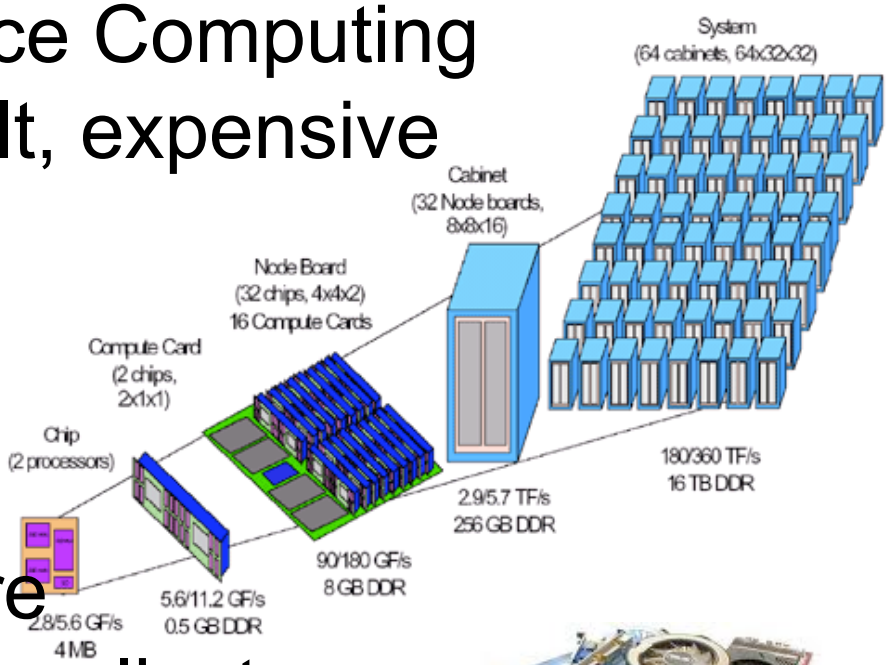
Cluster Architecture



In 2011, it was guestimated that Google had 1M machines, http://bit.ly/Shh0RO

Using Commodity Hardware

- 80-90's: High Performance Computing
  Very reliable, custom built, expensive



- Now: Consumer hardware
  Cheap, efficient, easy to replicate,
  BUT not very reliable,
- MUST deal with it!

Why commodity machines?

	HP INTEGRITY SUPERDOME-ITANIUM2	HP PROLIANT ML350 G5
Processor	64 sockets, 128 cores (dual-threaded), 1.6 GHz Itanium2, 12 MB last-level cache	1 socket, quad-core, 2.66 GHz X5355 CPU, 8 MB last-level cache
Memory	2,048 GB	24 GB
Disk storage	320,974 GB, 7,056 drives	3,961 GB, 105 drives
TPC-C price/performance	$2.93/tpmC	$0.73/tpmC
price/performance (server HW only)	$1.28/transactions per minute	$0.10/transactions per minute
Price/performance (server HW only) (no discounts)	$2.39/transactions per minute	$0.12/transactions per minute

Source: Barroso and Urs Hölzle (2009); performance figures from late 2007

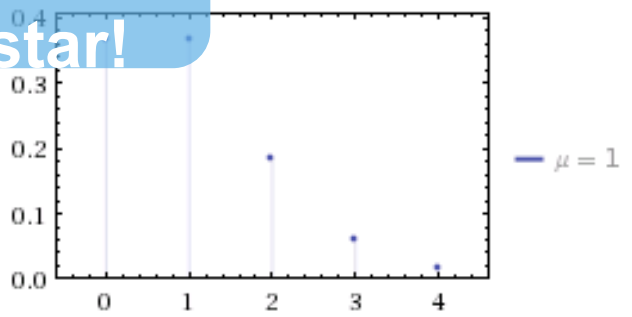
Fault Tolerance

- Performance goal
  - 1 failure per year
  - for a 1000-machine Cluster
- Poisson approximation

$$\Pr(n) = \frac{1}{n!} e^{-\mu} \mu^n$$

  - Assume failure rate μ per machine
  - Poisson rates of independent random variables are additive, so we can combine

not IBM Deskstar!

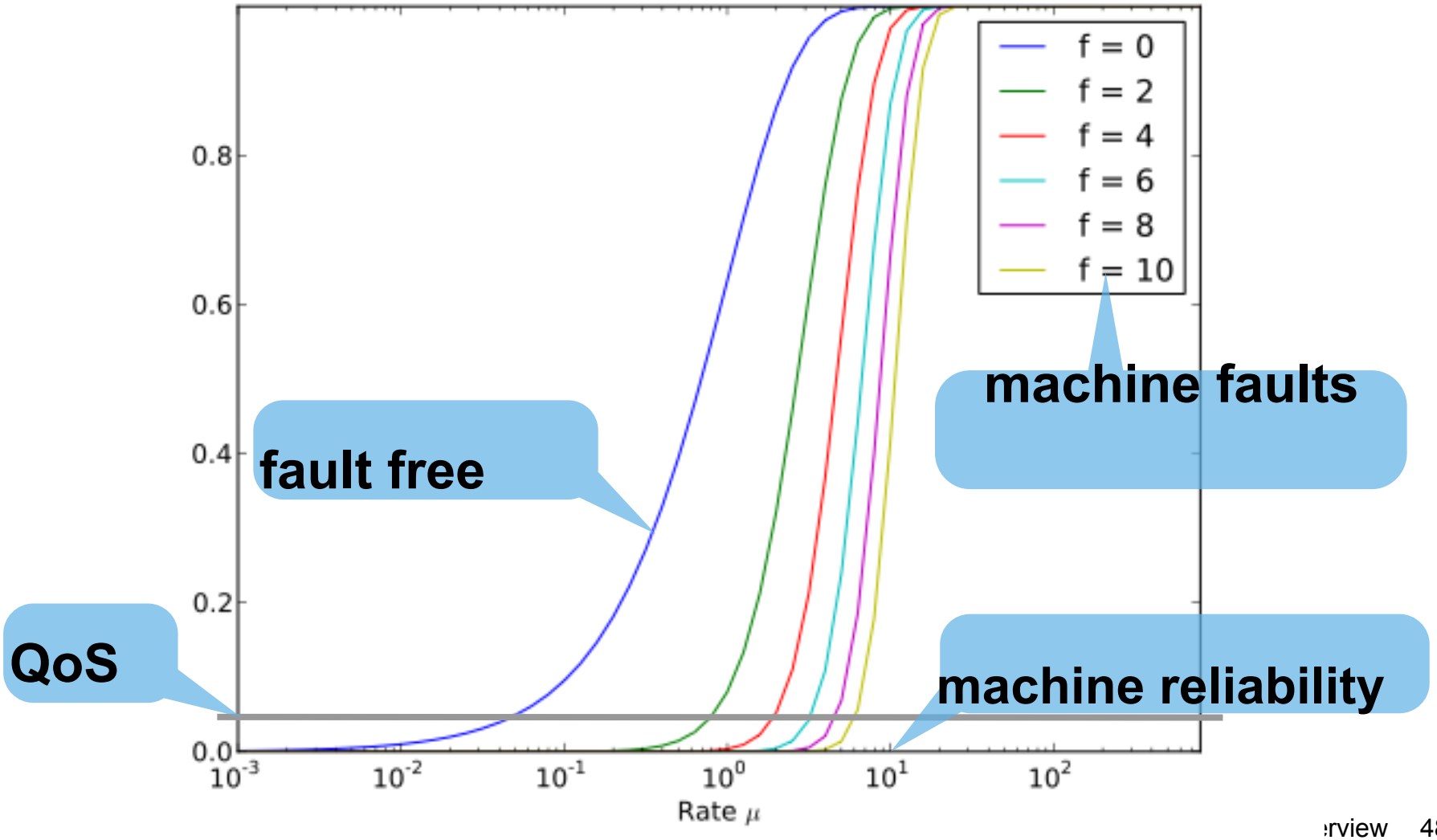


=> With Fault Intolerant Engineering
We need a rate of 1 failure per 1000 years per machine

- Fault tolerance
  Assume we can tolerate k faults among m machines in t time

$$\Pr(f > k) = 1 - \sum_{n=0}^{k} \frac{1}{n!} e^{-\lambda t} (\lambda t)^n$$

Fault tolerance



Hardware

The Joys of Real Hardware

Typical first year for a new cluster:

~0.5 overheating (power down most machines in <5 mins, ~1-2 days to recover)
~1 PDU failure (~500-1000 machines suddenly disappear, ~6 hours to come back)
~1 rack-move (plenty of warning, ~500-1000 machines powered down, ~6 hours)
~1 network rewiring (rolling ~5% of machines down over 2-day span)
~20 rack failures (40-80 machines instantly disappear, 1-6 hours to get back)
~5 racks go wonky (40-80 machines see 50% packetloss)
~8 network maintenances (4 might cause ~30-minute random connectivity losses)
~12 router reloads (takes out DNS and external vips for a couple minutes)
~3 router failures (have to immediately pull traffic for an hour)
~dozens of minor 30-second blips for dns
~1000 individual machine failures
~thousands of hard drive failures

slow disks, bad memory, misconfigured machines, flaky machines, etc.

Slide from talk of Jeff Dean:
http://research.google.com/people/jeff/stanford-295-**talk**.pdf

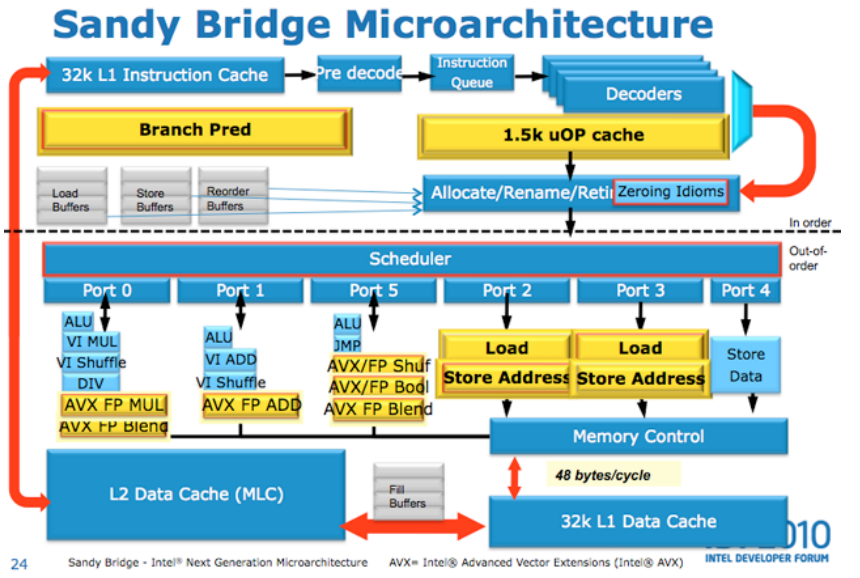
Google

“Facts” about Jeff Dean

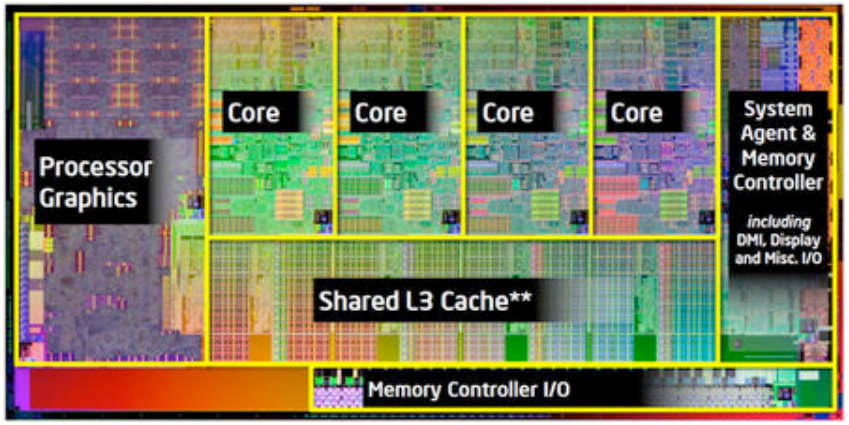
- Compilers don’t warn Jeff Dean. Jeff Dean warns compilers.
- Jeff Dean builds his code before committing it, but only to check for compiler and linker bugs.
- Jeff Dean writes directly in binary. He then writes the source code as a documentation for other developers.
- Jeff Dean once shifted a bit so hard, it ended up on another computer.
- When Jeff Dean has an ergonomic evaluation, it is for the protection of his keyboard.
- gcc -O4 emails your code to Jeff Dean for a rewrite.
- When he heard that Jeff Dean's autobiography would be exclusive to the platform, Richard Stallman bought a Kindle.
- Jeff Dean puts his pants on one leg at a time, but if he had more legs, you’d realize the algorithm is actually only O(log n)

CPU

- Multiple cores (4-8)
- Multiple sockets (1-4) per board
- 2-4 GHz clock
- 10-100W power
- Several cache levels (hierarchical, 8-16MB total)
- Vector processing units (SSE4, AVX)



- Perform several operations at once
- Use this for fast linear algebra (4-8 multiply adds in one operation)
- Memory interface 20-40GB/s
- Internal bandwidth >100GB/s
- 100+ GFlops for matrix matrix multiply
- Integrated low end GPU



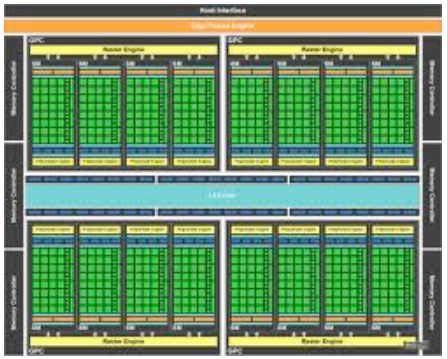
RAM

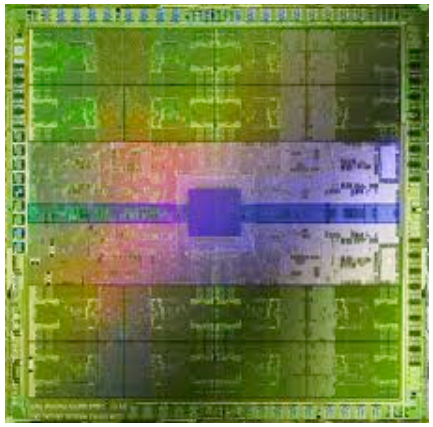
- 2-4 channels (32 bit wide)
- 1GHz speed
- High latency (10ns for DDR3)
- High burst data rate (>10 GB/s)

- Avoid random access in code if possible.
- Memory align variables
- Know your platform (FBDIMM vs. DDR)
  (code may run faster on old MacBookPro than a Xeon)

GPU

- Up to 512 cores / 200W
- Cores have hierarchical structure tricky to synchronize threads (interrupts, semaphores, etc.)
- 1-3GB memory (Tesla 6GB)
- 1 TFlop (single precision)
- Memory bandwidth > 100GB/s
- 4GB/s PCI bus bottleneck





Storage

- Harddisks
  - 3TB of storage (30GB/$)
  - 100 MB/s bandwidth (sequential)
  - 5 ms seek (200 IOPS)
  - cheap
- SSD
  - 100-500 MB storage (1GB/$)
  - 300 MB/s bandwidth (sequential)
  - 50,000 IOPS / 1 ms seek (queueing)
  - Random writes often faster than reads
  - reliable (but limited lifetime - NAND)

Numbers (Jeff Dean says) Everyone Should Know

L1 cache reference	0.5 ns
Branch mispredict	5 ns
L2 cache reference	7 ns
Mutex lock/unlock	100 ns
Main memory reference	100 ns
Compress 1K bytes with Zippy	10,000 ns
Send 2K bytes over 1 Gbps network	20,000 ns
Read 1 MB sequentially from memory	250,000 ns
Round trip within same datacenter	500,000 ns
Disk seek	10,000,000 ns
Read 1 MB sequentially from network	10,000,000 ns
Read 1 MB sequentially from disk	30,000,000 ns
Send packet CA->Netherlands->CA	150,000,000 ns

~= 10x

~= 15x

40x diff

~= 100,000x slower than main mem access

A typical disk

What do we count?

- Compilers don't warn Jeff Dean. Jeff Dean warns compilers.
- ….
- Memory access/instructions are *qualitatively different* from disk access
- Seeks are *qualitatively different* from sequential reads on disk
- Cache, disk fetches, etc work best when you stream through data *sequentially*
- Best case for data processing: stream through the data *once* in *sequential order,* as it's found on disk.

Seeks vs. Scans

- Consider a 1 TB database with 100 byte records
  - We want to update 1 percent of the records
- Scenario 1: random access
  - Each update takes ~30 ms (seek, read, write)
  - 10^8 updates = ~35 days
- Scenario 2: rewrite all records
  - Assume 100 MB/s throughput
  - Time = 5.6 hours(!)
- Lesson: avoid random seeks!

Source: Ted Dunning, on Hadoop mailing list

Other lessons -?

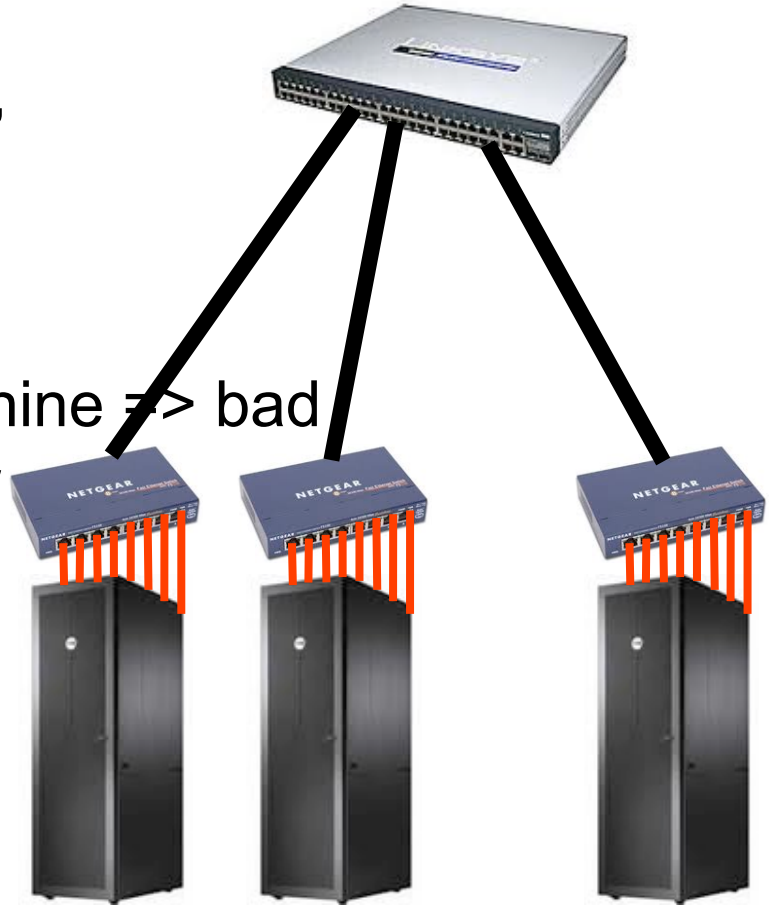
Encoding Your Data

- CPUs are fast, memory/bandwidth are precious, ergo…
  - Variable-length encodings
  - Compression
  - Compact in-memory representations
- Compression very important aspect of many systems
  - inverted index posting list formats
  - storage systems for persistent data

Switches & Colos

- In theory perfect point to point bandwidth (e.g. 1Gb Ethernet)
- Big switches are expensive
  crossbar bandwidth linear in #ports,

  BUT price superlinear
- Real switches have finite buffers
  - many connections to a single machine => bad
  - buffer overflow / dropped packets / collision avoidance
- Hierarchical structure
  - more bandwidth within rack
  - lower latency within rack
  - lots of latency between Colos
- Hadoop gives you machines where the data is (not necessarily on same rack!)



Communication Cost ?

- Nodes need to talk to each other!
  - SMP (Symmetric Multi-Processor machine): latencies ~100 ns
  - LAN: latencies ~100 us
- Scaling “up” vs. scaling “out”
  - Smaller cluster of SMP machines vs. larger cluster of commodity machines
  - E.g., 8 128-core machines vs. 128 8-core machines
  - Note: no single SMP machine is big enough
- Let’s model communication overhead…

Source: analysis on this and subsequent slides from Barroso and Urs Hölzle, “The Datacenter as a Computer,” 2009.

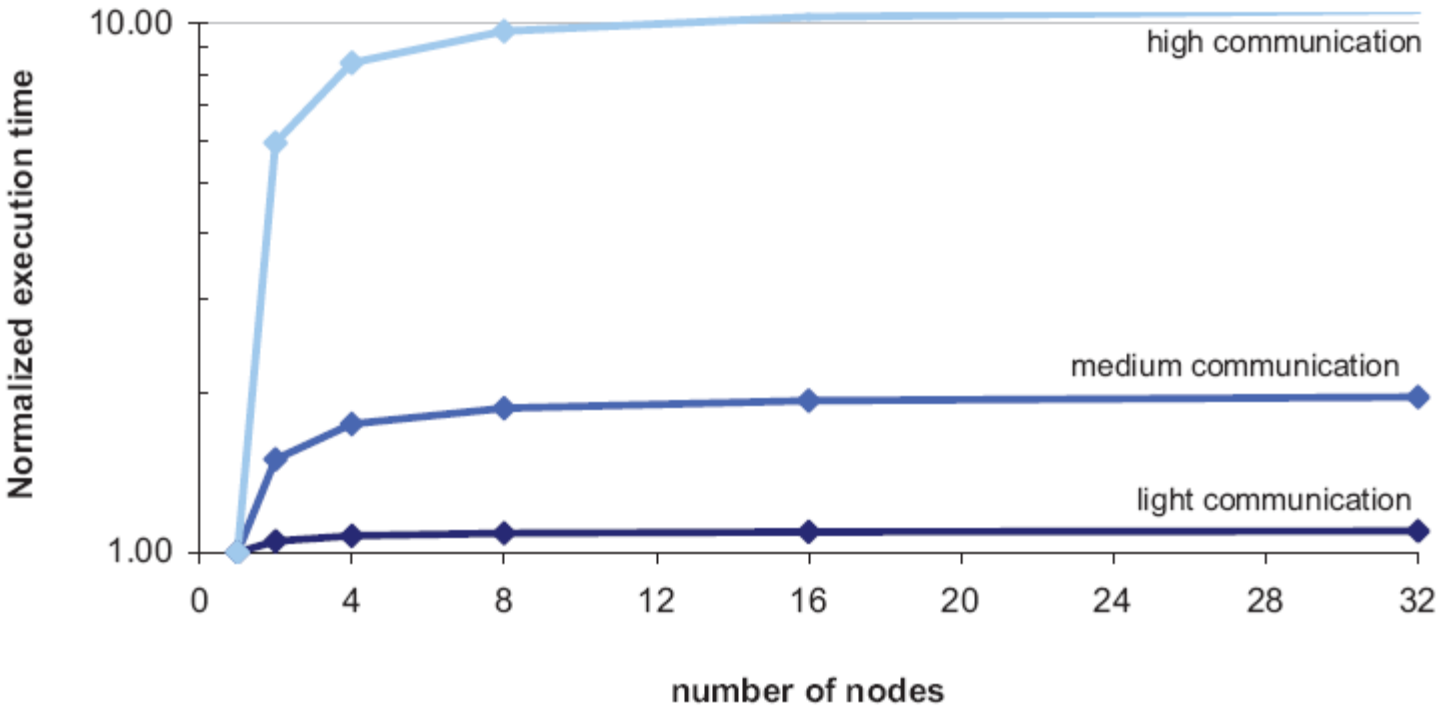
Modeling Communication Costs

- Simple execution cost model:
  - Fraction of local access inversely proportional to size of cluster
  - *n* nodes (each node is a shared-memory SMP domain, ignoring cores for now)
  - Total cost = cost of computation + cost to access global data

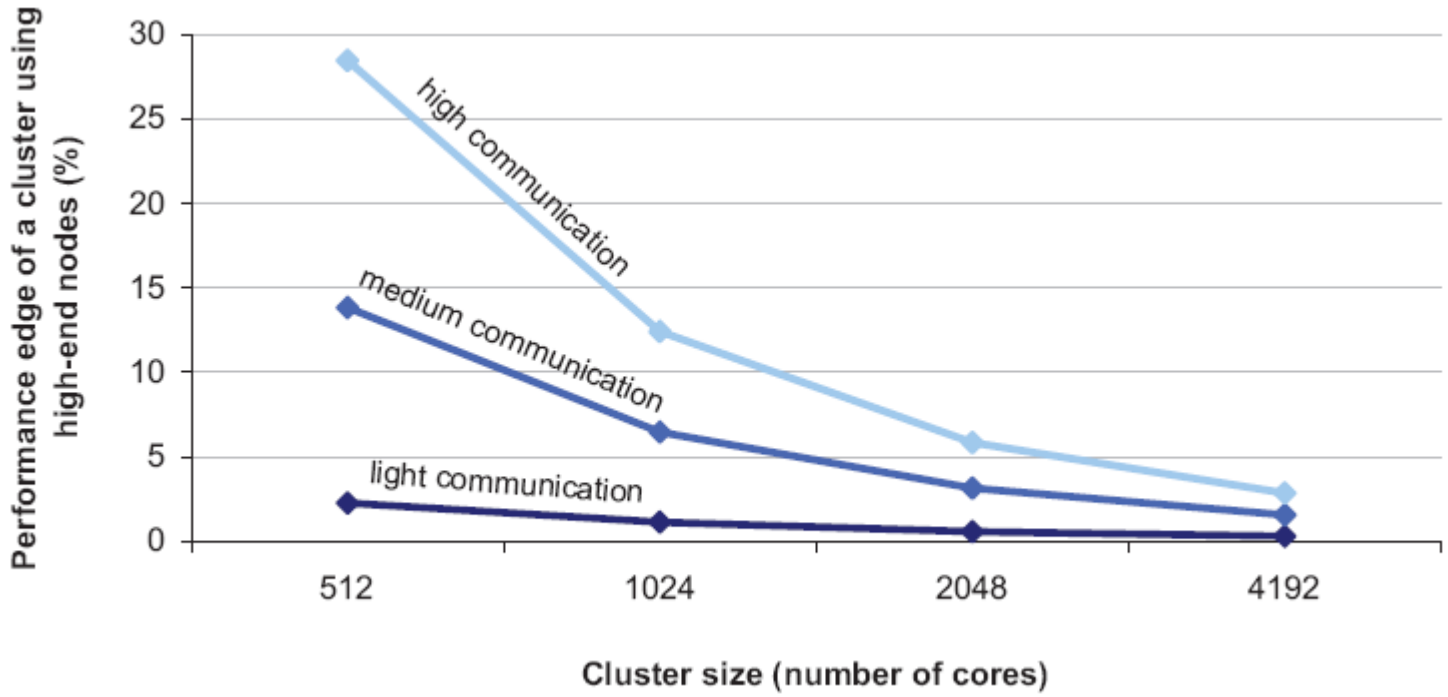
$$= 1 \text{ ms} + f \times [100 \text{ ns} / n + 100 \text{ us} \times (1 - 1/n)]$$

    - Light communication: $f = 1$
    - Medium communication: $f = 10$
    - Heavy communication: $f = 100$
- What are the costs in parallelization?

Cost of Parallelization



Advantages of scaling “up”



So why not?

Data Intensive Computing

- Data collection too large to transmit economically over Internet --- Petabyte data collections
- Computation produces small data output containing a high density of information
- Implemented in “Clouds”
- Easy to write programs, fast turn around.
- MapReduce.
  - Map(k1, v1) -> list (k2, v2)
  - Reduce(k2,list(v2)) -> list(v3)
- Hadoop (Yarn), PIG, HDFS, Hbase
- Sawzall, Google File System, BigTable

The datacenter *is* the computer

“Big Ideas”

- Scale “out”, not “up”
  - Limits of SMP and large shared-memory machines
- Move processing to the data
  - Cluster have limited bandwidth
- Process data sequentially, avoid random access
  - Seeks are expensive, disk throughput is reasonable
- Seamless scalability
  - From the mythical man-month to the tradable machine-hour

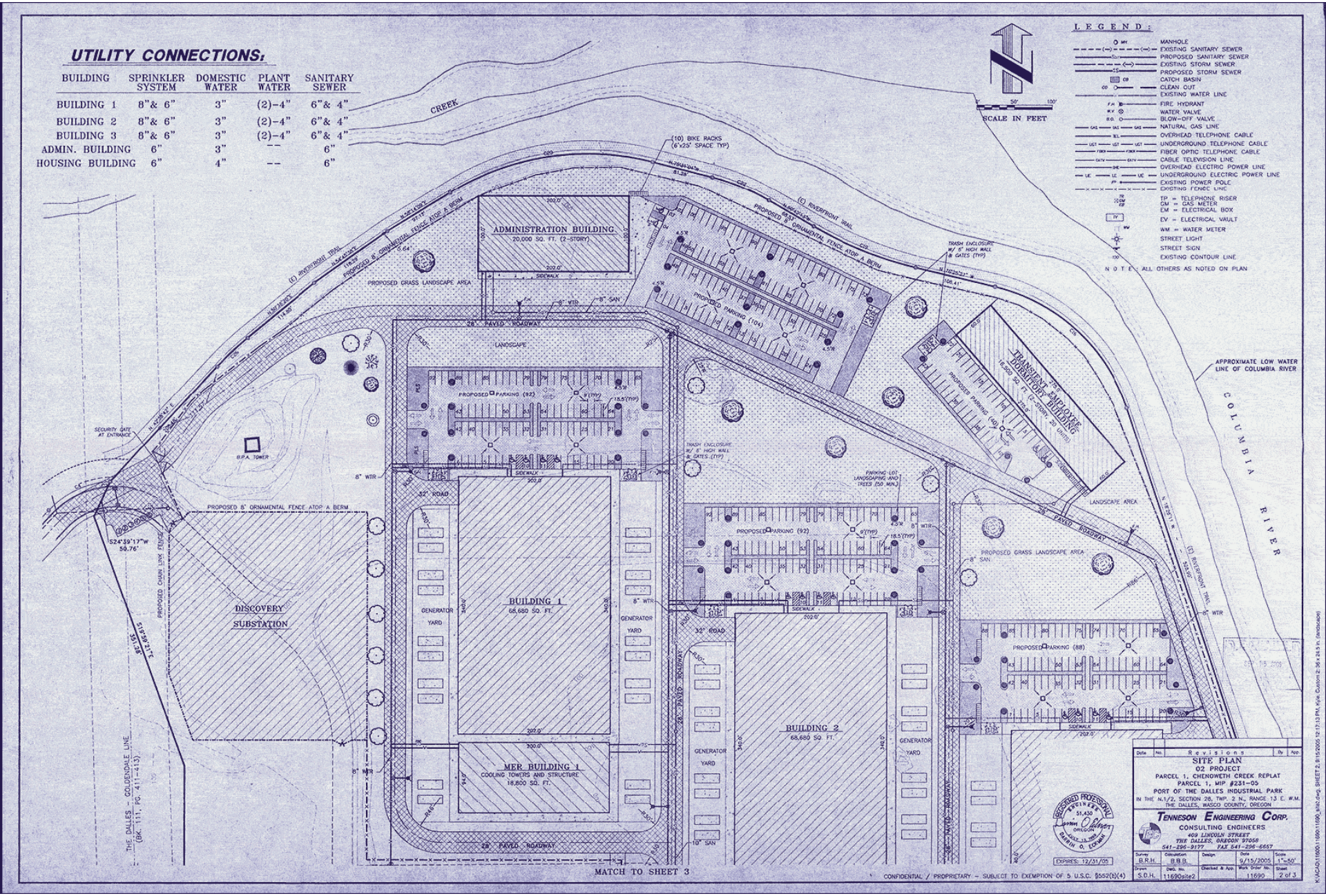
“Big Ideas”

- Scale “out”, not “up”
  - Limits of SMP and large shared-memory machines
- Move processing to the data
  - Cluster have limited bandwidth
- Process data sequentially, avoid random access
  - Seeks are expensive, disk throughput is reasonable
- Seamless scalability
  - From the mythical man-month to the tradable machine-hour

Source: Wikipedia (The Dalles, Oregon)

Source: NY Times (6/14/2006)

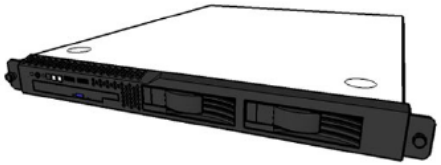
Source: www.robinmajumdar.com

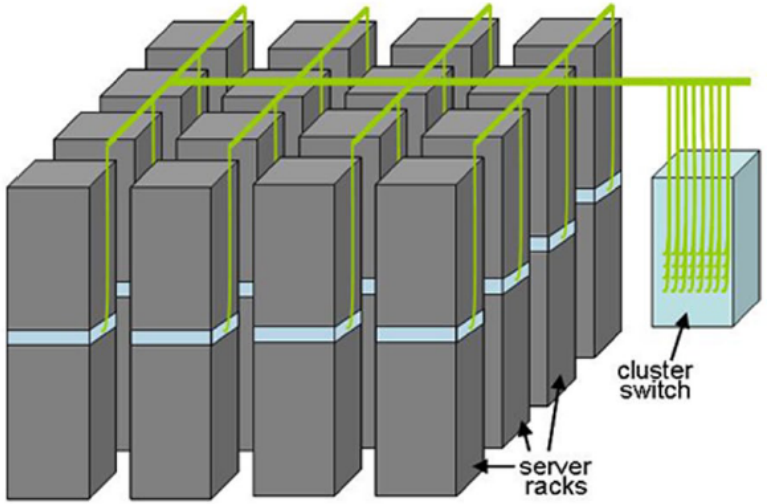


Source: Harper's (Feb, 2008)

Source: Bonneville Power Administration

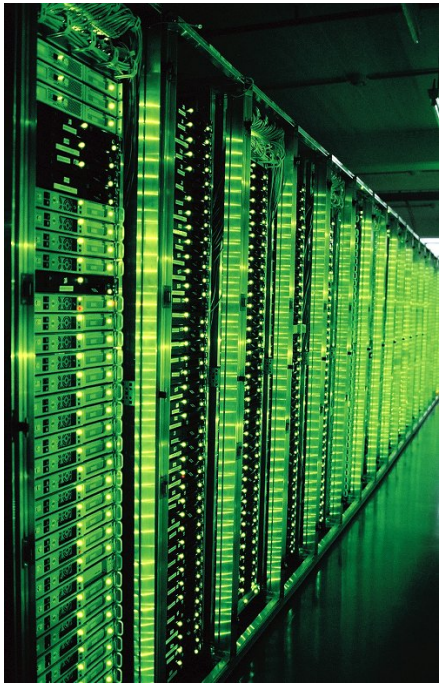
Building Blocks





Source: Barroso and Urs Hölzle (2009)

What's in a data center?



- Hundreds or thousands of racks

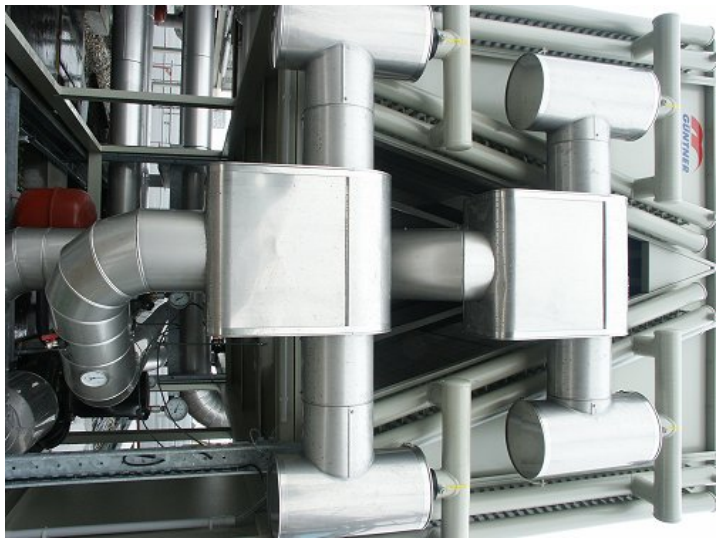
What's in a data center?

- Massive networking

What's in a data center?

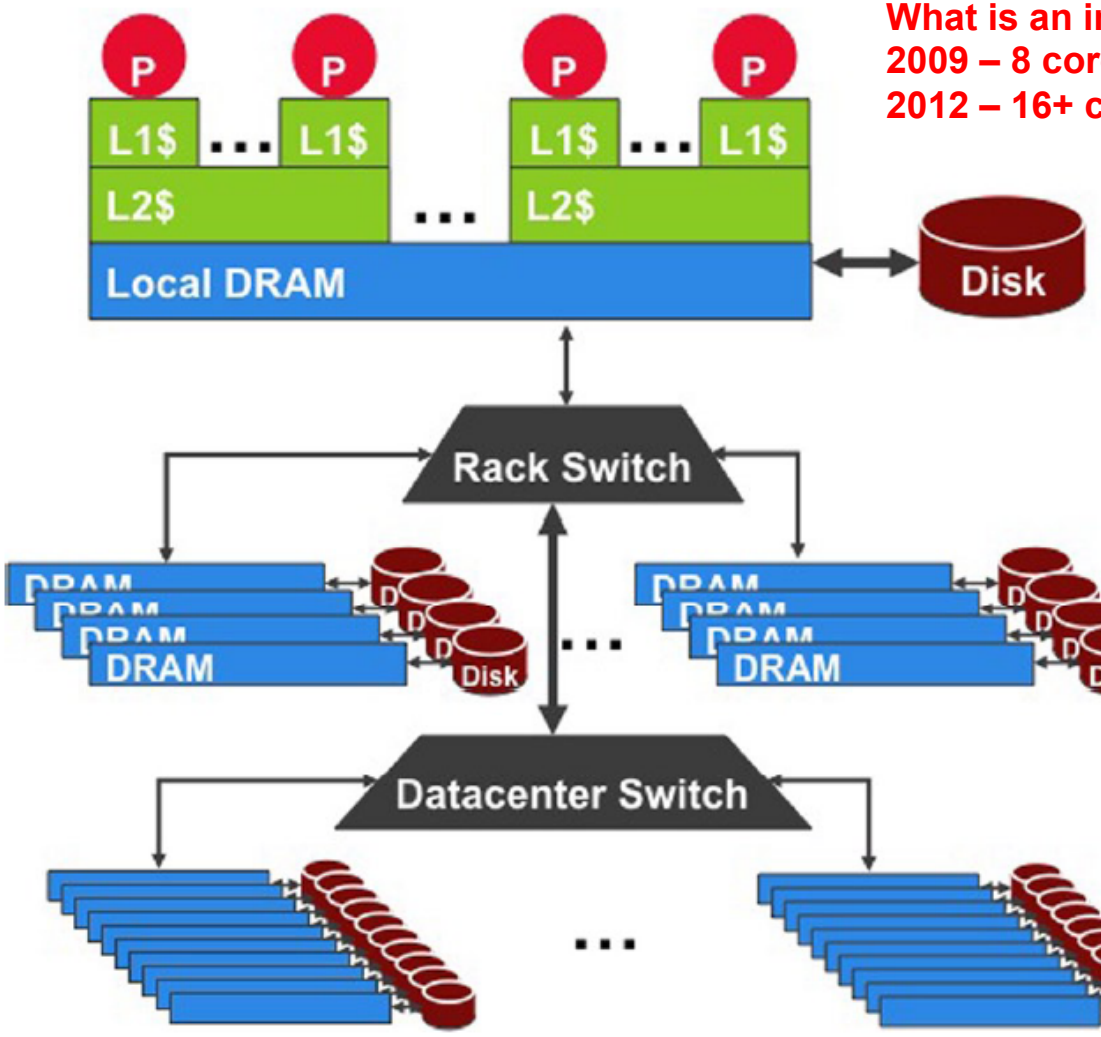
- Emergency power supplies

What's in a data center?



- Massive cooling

Storage Hierarchy



What is an industrial "Commodity Machine" ?
2009 – 8 cores, 16GB RAM, 4x1TB Disks
2012 – 16+ cores, 48-96GB RAM, 12x(2~3)TB Disks

One server
DRAM: 16GB, 100ns, 20GB/s
Disk: 2TB, 10ms, 200MB/s

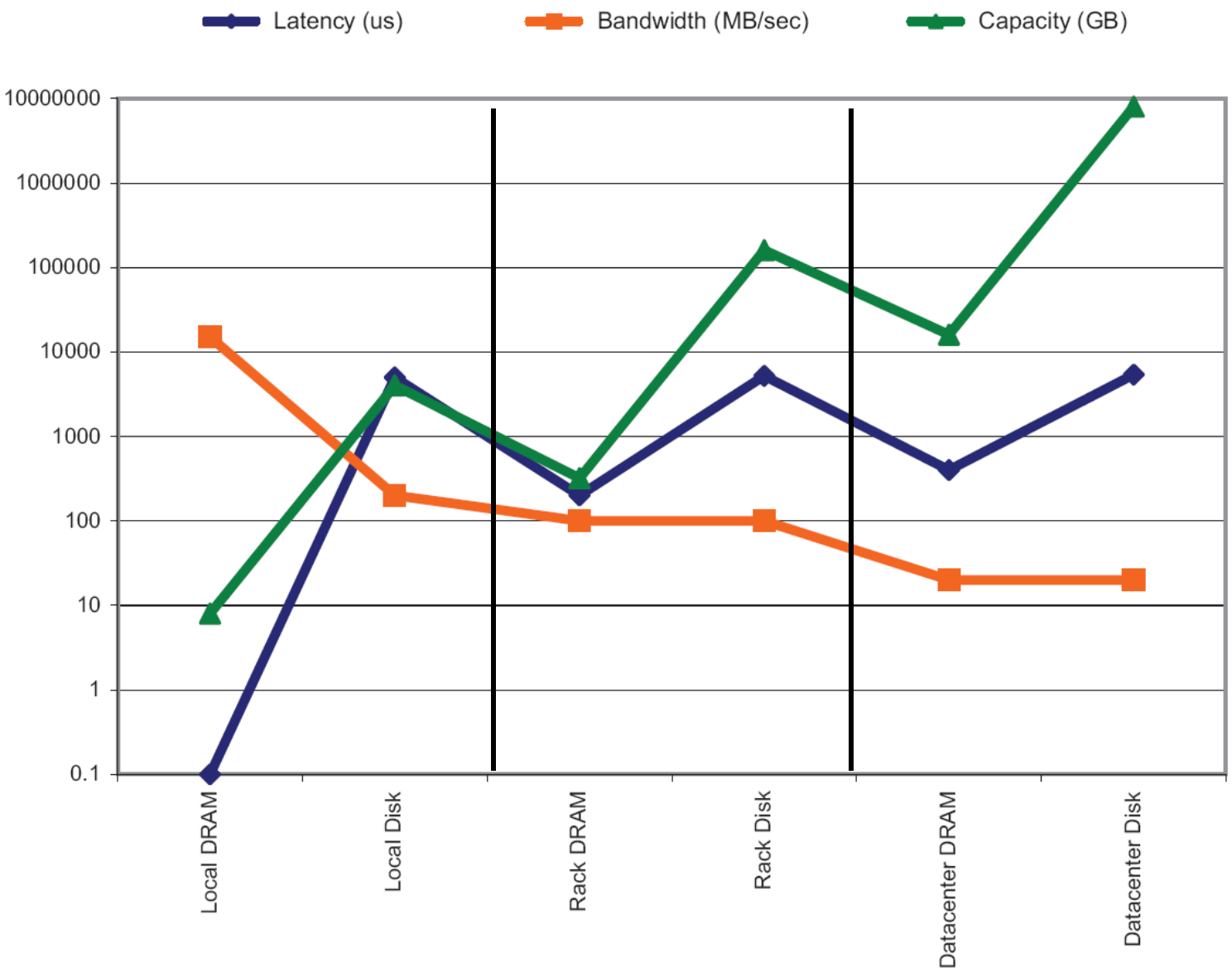
Local rack (80 servers)
DRAM: 1TB, 300us, 100MB/s
Disk: 160TB, 11ms, 100MB/s

Cluster (30 racks)
DRAM: 30TB, 500us, 10MB/s
Disk: 4.80PB, 12ms, 10MB/s

The sense of scale…

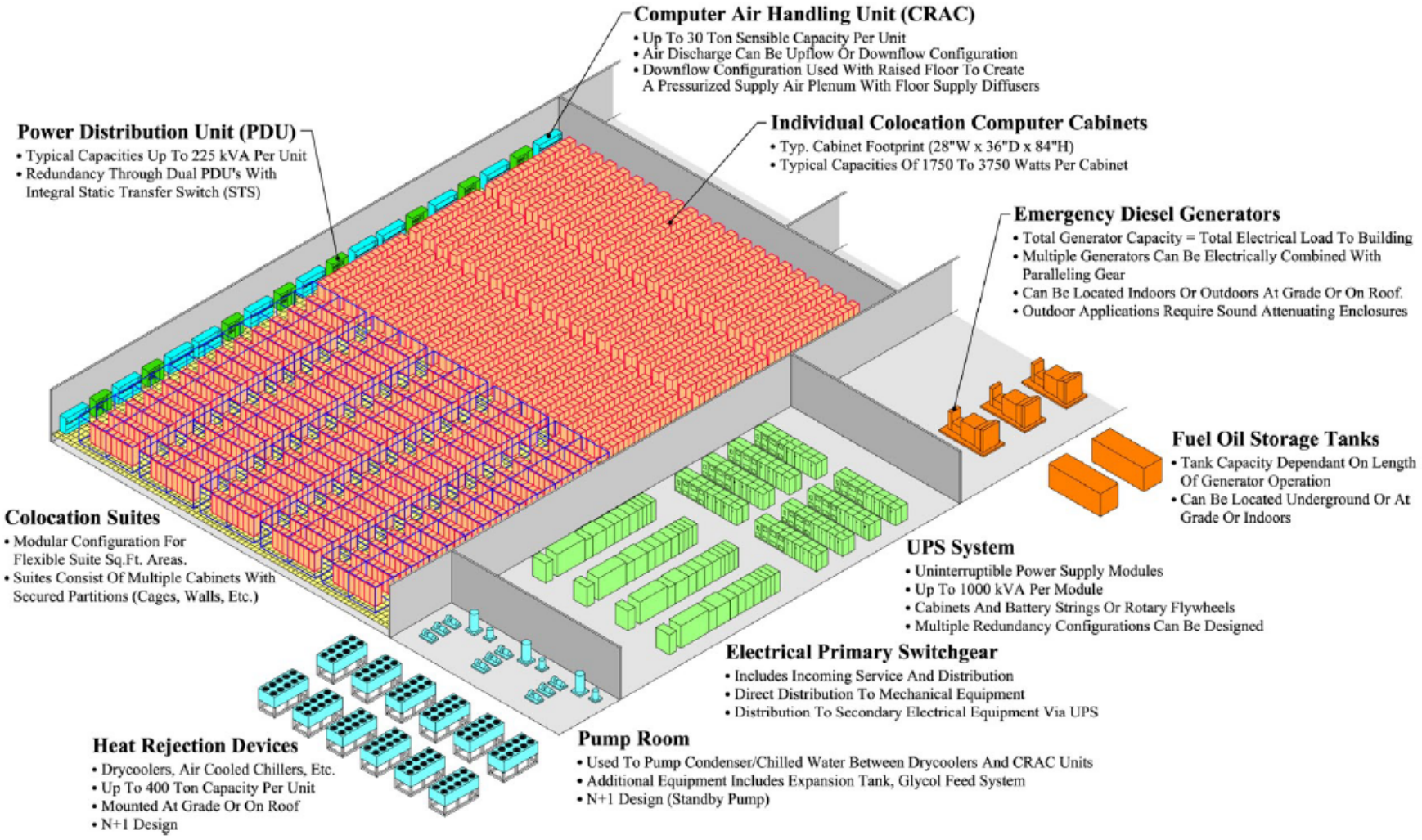
Source: Barroso and Urs Hölzle (2009)

Storage Hierarchy



Source: Barroso and Urs Hölzle (2009)

Anatomy of a Datacenter



Source: Barroso and Urs Hölzle (2009)

Energy matters!

Company	Servers	Electricity	Cost
eBay	16K	~0.6*10^5 MWh	~$3.7M/yr
Akamai	40K	~1.7*10^5 MWh	~$10M/yr
Rackspace	50K	~2*10^5 MWh	~$12M/yr
Microsoft	>200K	>6*10^5 MWh	>$36M/yr
Google	>500K	>6.3*10^5 MWh	>$38M/yr
USA (2006)	10.9M	610*10^5 MWh	$4.5B/yr

- Data centers consume a lot of energy
  - Makes sense to build them near sources of cheap electricity
  - Example: Price per KWh is 3.6ct in Idaho (near hydroelectric power), 10ct in California (long distance transmission), 18ct in Hawaii (must ship fuel)
  - Most of this is converted into heat  Cooling is a big issue!

Scaling up





- What if even a data center is not big enough?
  - Build additional data centers
  - Where? How many?

Global distribution

- Data centers are often globally distributed
  - Example above: Google data center locations (inferred)
- Why?
  - Need to be close to users (physics!)
  - Cheaper resources
  - Protection against failures

Trend: Modular data center

- Need more capacity? Just deploy another container!

Justifying the “Big Ideas”

- Scale “out”, not “up”
  - Limits of SMP and large shared-memory machines
- Move processing to the data
  - Cluster have limited bandwidth
- Process data sequentially, avoid random access
  - Seeks are expensive, disk throughput is reasonable
- Seamless scalability
  - From the mythical man-month to the tradable machine-hour

Recap

- Web-Scale Data – their sources and uses
- What is this Course about ?
- How to scale up hardware for Web-scale Data processing ?
- Computing Infrastructure for Web-scale Data Processing