UNIVERSITY OF WATERLOO

Data-Intensive Distributed Computing

CS 431/631 451/651 (Fall 2019)

Part 1: MapReduce Algorithm Design (1/4)

Ali Abedi

These slides are available at https://www.student.cs.uwaterloo.ca/~cs451/

Agenda for Today

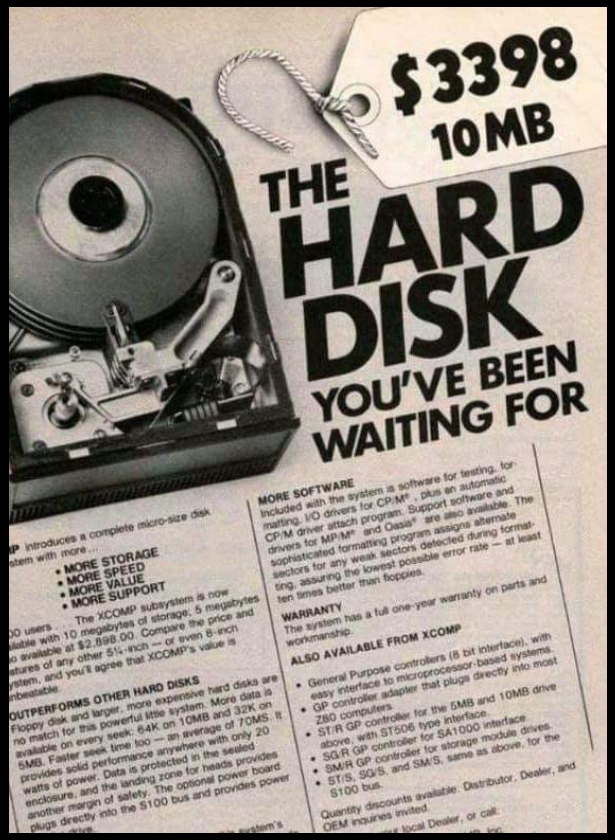
Who am I?
What is big data?
Why big data?
What is this course about?
Administrivia

Who am I?

PhD from Waterloo (2017)
Systems and Networking Research Group

Source: Wikipedia (Hard disk drive)

1950s

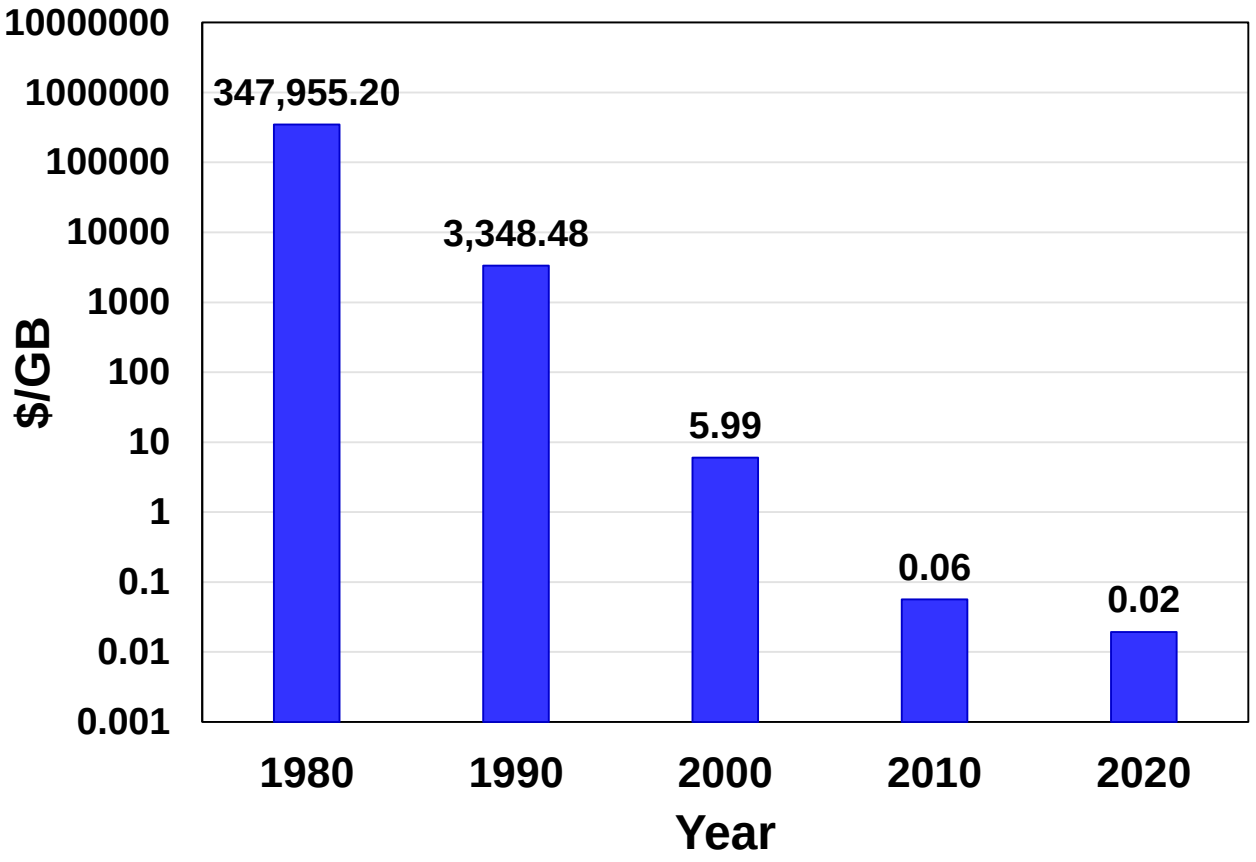


1980s

Today

Storage evolution over time

Storage cost over time



Google

Processes 20 PB a day (2008)
Crawls 20B web pages a day (2012)
Search index is 100+ PB (5/2014)
Bigtable serves 2+ EB, 600M QPS (5/2014)

INTERNET ARCHIVE

400B pages,
10+ PB (2/2014)

YAHOO!

19 Hadoop clusters: 600 PB, 40k servers (9/2015)

JPMorganChase

150 PB on 50k+ servers running 15k apps (6/2011)

ebay

Hadoop: 10K nodes, 150K cores, 150 PB (4/2014)

300 PB data in Hive + 600 TB/day (4/2014)

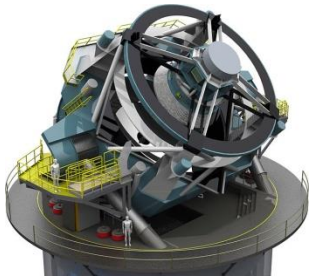
facebook

amazon web services™

S3: 2T objects, 1.1M request/second (4/2013)

LHC: ~15 PB a year

CERN



LSST: 6-10 PB a year (~2020)

640K ought to be enough for anybody.

SKA: 0.3 – 1.5 EB per year (~2020)

How much data?

A DAY IN DATA

The exponential growth of data is undisputed, but the numbers behind this explosion – fuelled by internet of things and the use of connected devices – are hard to comprehend, particularly when looked at in the context of one day

500m

tweets are sent every day

Twitter

4PB

of data created by Facebook, including

350m photos

100m hours of video watch time

Facebook Research

294bn

billion emails are sent

Radicati Group

320bn emails to be sent each day by 2021

306bn emails to be sent each day by 2020

3.9bn

people use emails

4TB

of data produced by a connected car

Intel

65bn

messages sent over WhatsApp and two billion minutes of voice and video calls made

Facebook

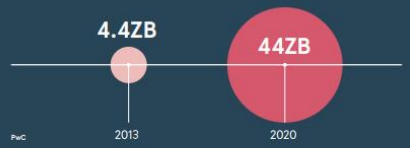
DEMYSTIFIYING DATA UNITS

From the more familiar 'bit' or 'megabyte', larger units of measurement are more frequently being used to explain the masses of data

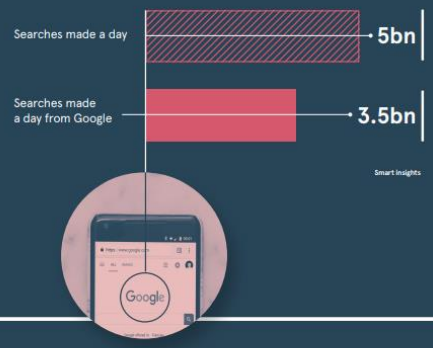
Unit		Value	Size
b	bit	0 or 1	1/8 of a byte
B	byte	8 bits	1 byte
KB	kilobyte	1,000 bytes	1,000 bytes
MB	megabyte	$1{,}000^2$ bytes	1,000,000 bytes
GB	gigabyte	$1{,}000^3$ bytes	1,000,000,000 bytes
TB	terabyte	$1{,}000^4$ bytes	1,000,000,000,000 bytes
PB	petabyte	$1{,}000^5$ bytes	1,000,000,000,000,000 bytes
EB	exabyte	$1{,}000^6$ bytes	1,000,000,000,000,000,000 bytes
ZB	zettabyte	$1{,}000^7$ bytes	1,000,000,000,000,000,000,000 bytes
YB	yottabyte	$1{,}000^8$ bytes	1,000,000,000,000,000,000,000,000 bytes

*A lowercase "b" is used as an abbreviation for bits, while an uppercase "B" represents bytes.

ACCUMULATED DIGITAL UNIVERSE OF DATA



PwC



Smart Insights

463EB

of data will be created every day by 2025

IDC

95m

photos and videos are shared on Instagram

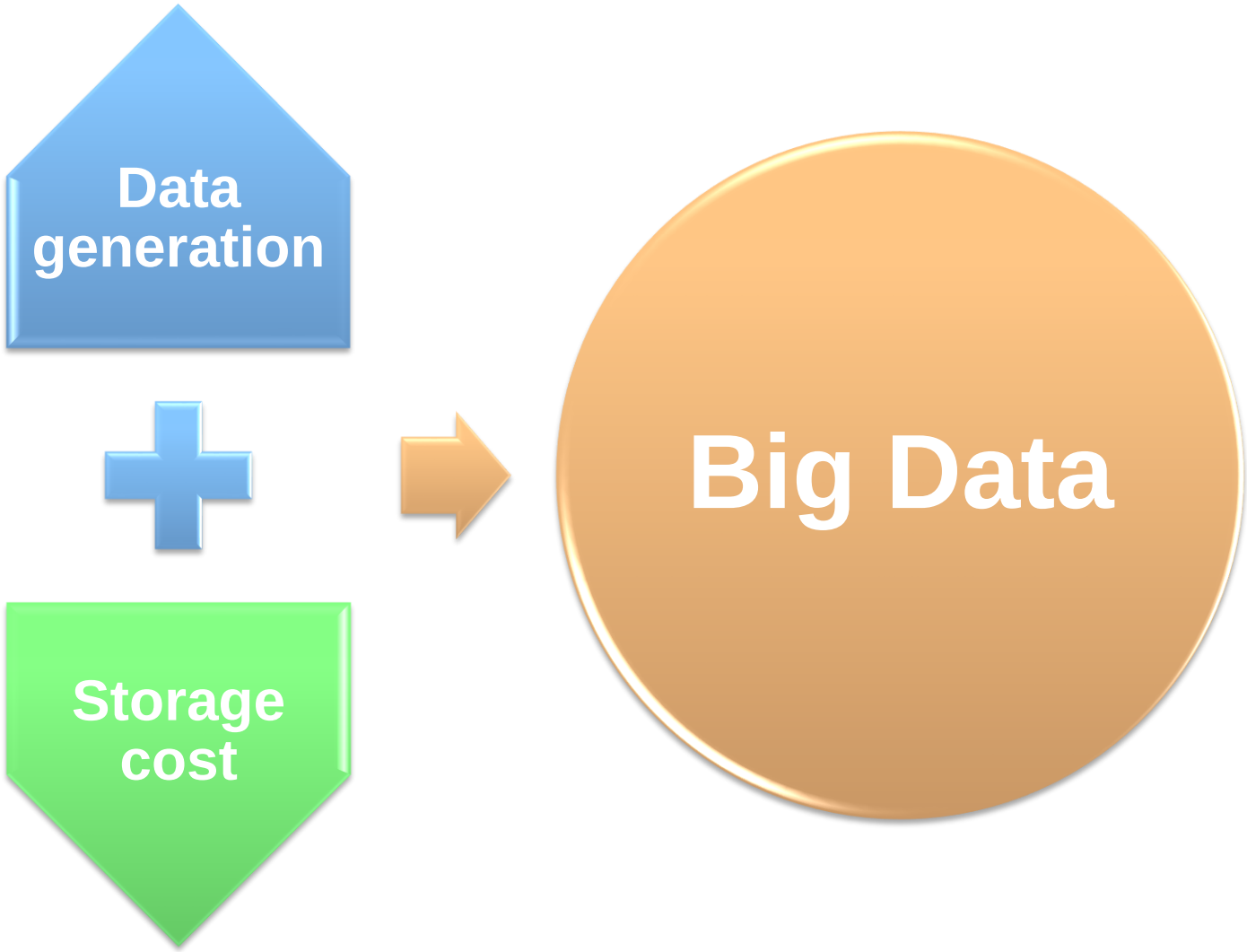
Instagram Business

28PB

to be generated from wearable devices by 2020

Statista

RACONTEUR


Data generation
Storage cost
Big Data

Why big data?

Science
Business
Society

Source: Wikipedia (Everest)

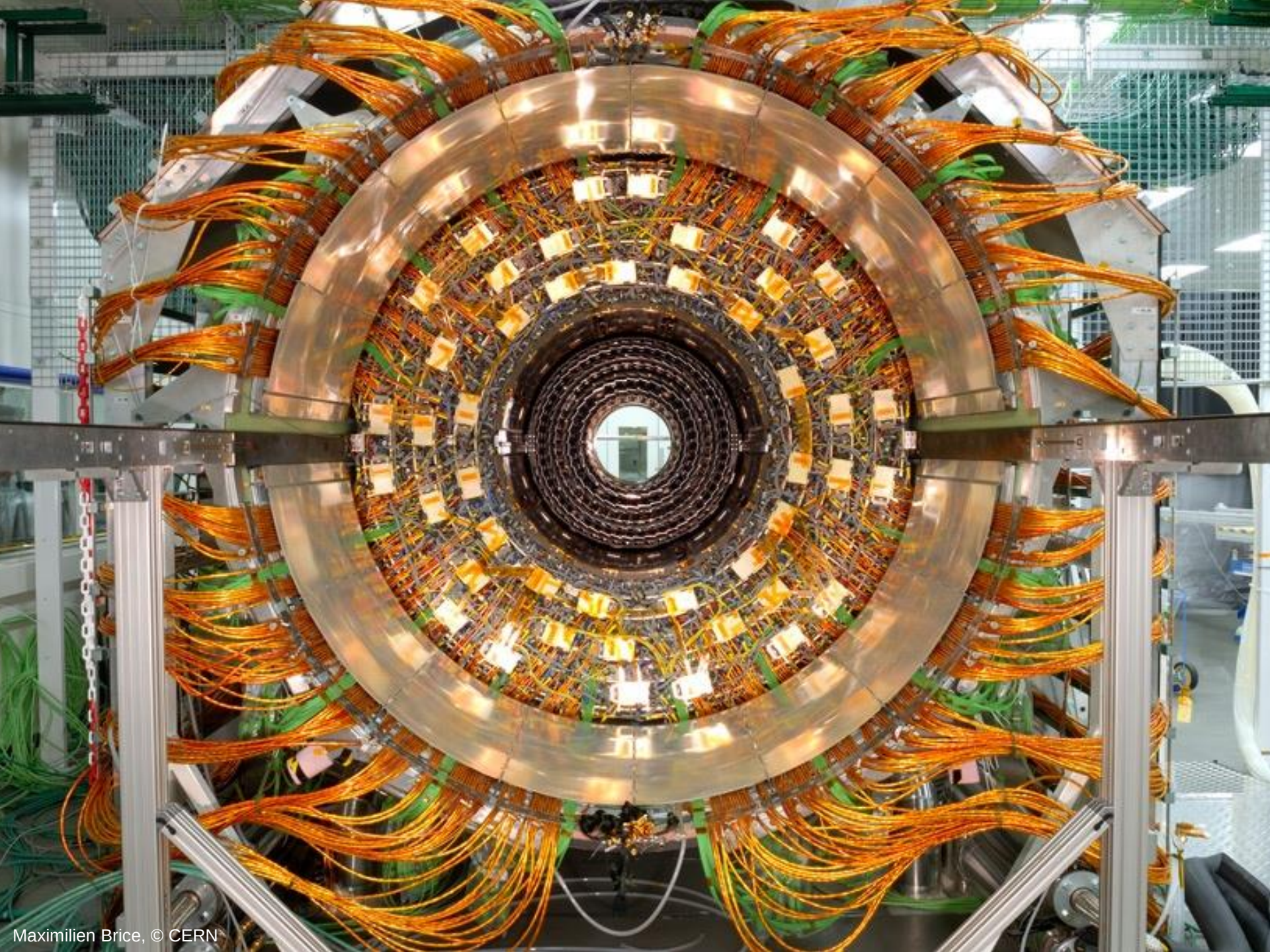
Science

Emergence of the 4th Paradigm

Data-intensive e-Science

Maximilien Brice, © CERN

Maximilien Brice, © CERN



Maximilien Brice, © CERN

Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC

The ATLAS Collaboration

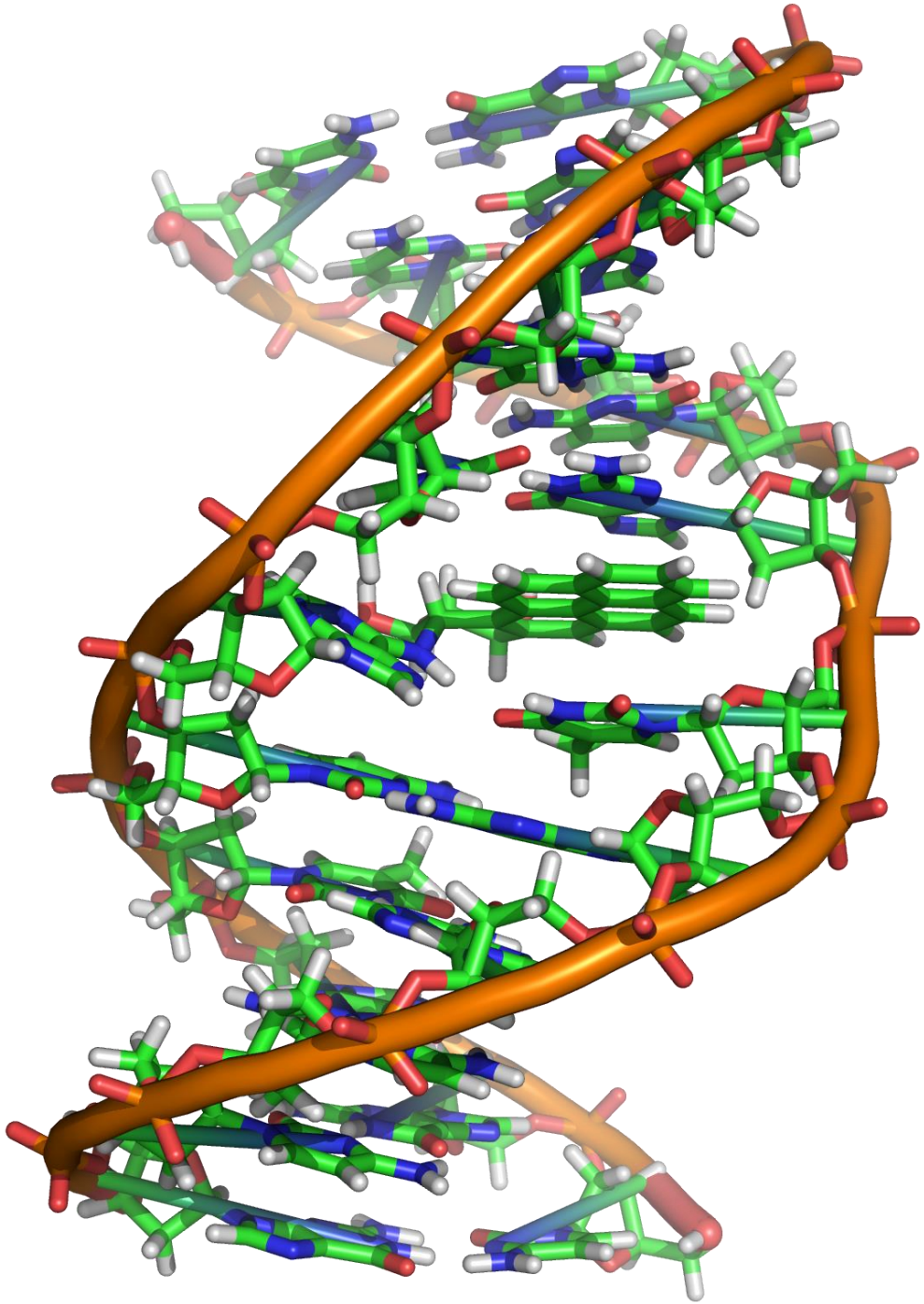
(Submitted on 31 Jul 2012 (v1), last revised 31 Aug 2012 (this version, v2))

A search for the Standard Model Higgs boson in proton-proton collisions with the ATLAS detector at the LHC is presented. The datasets used correspond to integrated luminosities of approximately 4.8 fb^-1 collected at sqrt(s) = 7 TeV in 2011 and 5.8 fb^-1 at sqrt(s) = 8 TeV in 2012. Individual searches in the channels H->ZZ^(*)->llll, H->gamma gamma and H->WW->e nu mu nu in the 8 TeV data are combined with previously published results of searches for H->ZZ^(*), WW^(*), bbbar and tau^+tau^- in the 7 TeV data and results from improved analyses of the H->ZZ^(*)->llll and H->gamma gamma channels in the 7 TeV data. Clear evidence for the production of a neutral boson with a measured mass of 126.0 +/- 0.4(stat) +/- 0.4(sys) GeV is presented. This observation, which has a significance of 5.9 standard deviations, corresponding to a background fluctuation probability of 1.7x10^-9, is compatible with the production and decay of the Standard Model Higgs boson.

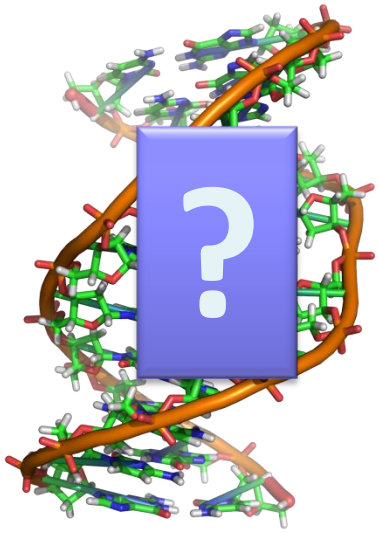
Comments:	24 pages plus author list (38 pages total), 12 figures, 7 tables, revised author list, matches version to appear in Physics Letters B
Subjects:	**High Energy Physics - Experiment (hep-ex)**
Journal reference:	Phys.Lett. B716 (2012) 1-29
DOI:	10.1016/j.physletb.2012.08.020
Report number:	CERN-PH-EP-2012-218
Cite as:	**arXiv:1207.7214 [hep-ex]**
	(or **arXiv:1207.7214v2 [hep-ex]** for this version)

Submission history

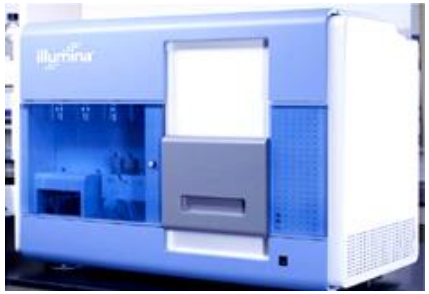
From: Atlas Publications [view email]
[v1] Tue, 31 Jul 2012 11:59:59 GMT (334kb)
[v2] Fri, 31 Aug 2012 19:29:54 GMT (334kb)



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)

Business

Data-driven decisions

Data-driven products

Source: Wikiedia (Shinjuku, Tokyo)

Business Intelligence

An organization should retain data that result from carrying out its mission and exploit those data to generate insights that benefit the organization, for example, market analysis, strategic planning, decision making, etc.

Duh!?

This is not a new idea!

In the 1990s, Wal-Mart found that customers tended to buy diapers and beer together. So they put them next to each other and increased sales of both.*

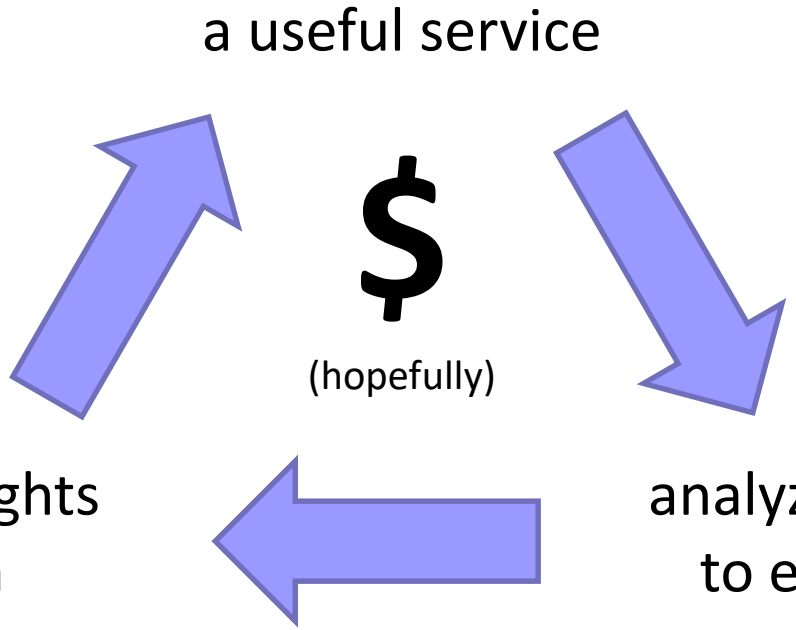
So what's changed?

More compute and storage

Ability to gather behavioral data

* BTW, this is completely apocryphal. (But it makes a nice story.)

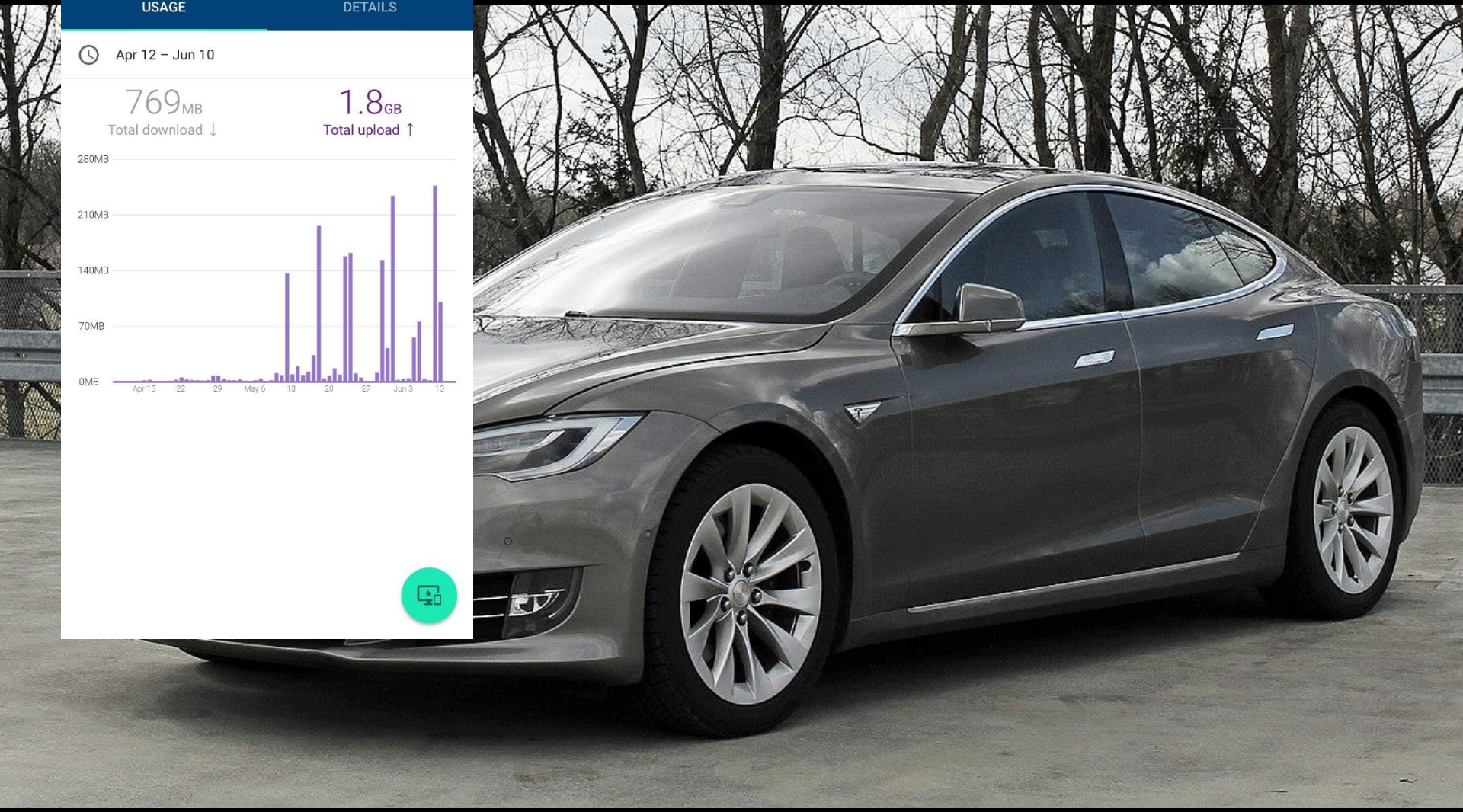
Virtuous Product Cycle



Google. Facebook. Twitter. Amazon. Uber.

data products **data science**

Source: https://images.lookhuman.com/render/standard/8002245806006052/pillow14in-whi-z1-t-netflixing.png



Source: https://www.reddit.com/r/teslamotors/comments/6gsc6v/i_think_the_neural_net_mining_is_just_starting/ (June 2017)

Translate

Turn off instant translation

Chinese | English | Spanish | Detect language

English | Chinese (Traditional) | Chinese (Simplified) | Translate

How does Google translate English into Chinese?

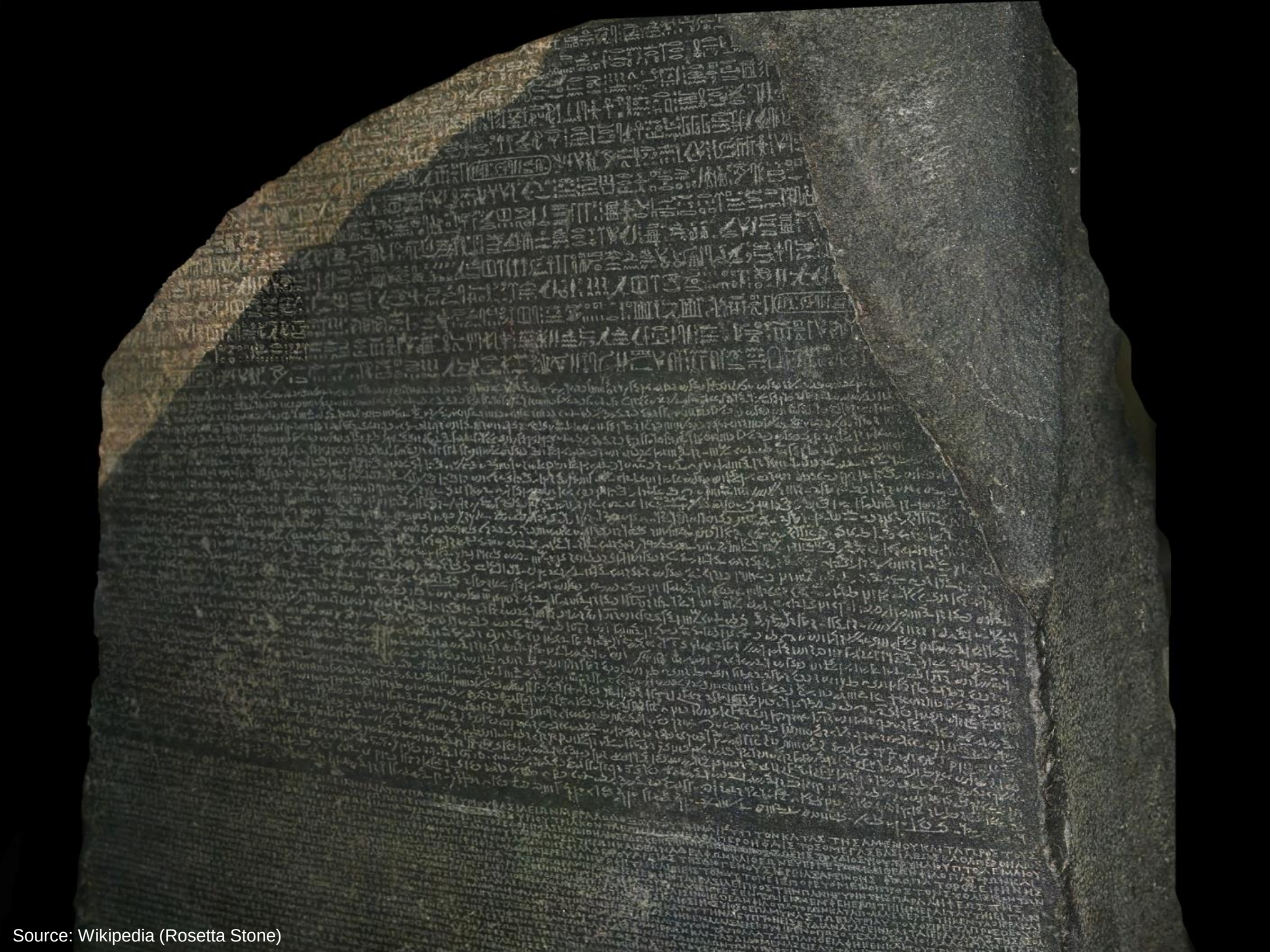
47/5000

Google如何将英语翻译成中文?

Suggest an edit

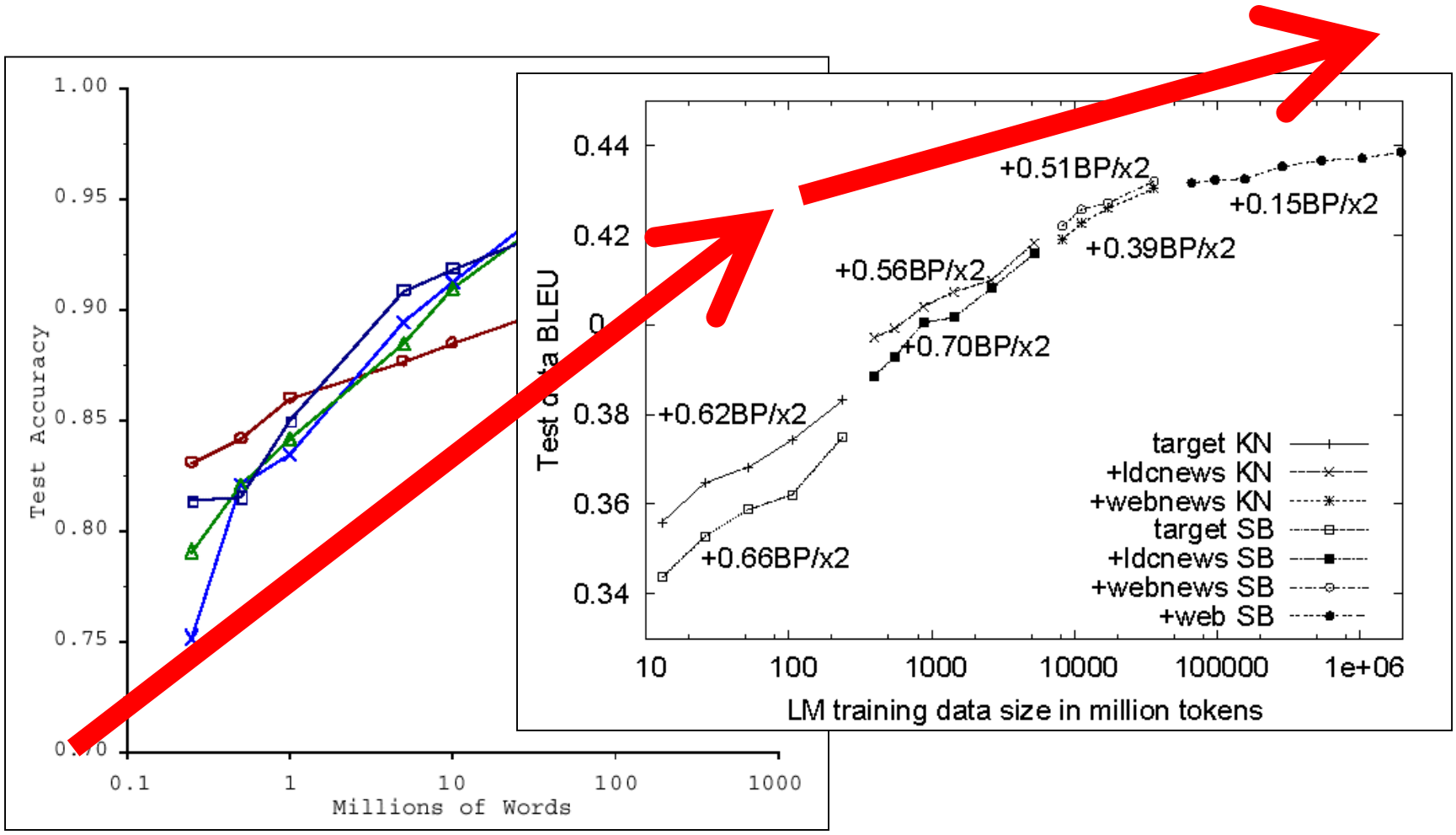
Google rúhé jiāng yīngyǔ fānyì chéng zhōngwén?

Google Translate for Business: Translator Toolkit Website Translator



Source: Wikipedia (Rosetta Stone)

No data like more data!



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

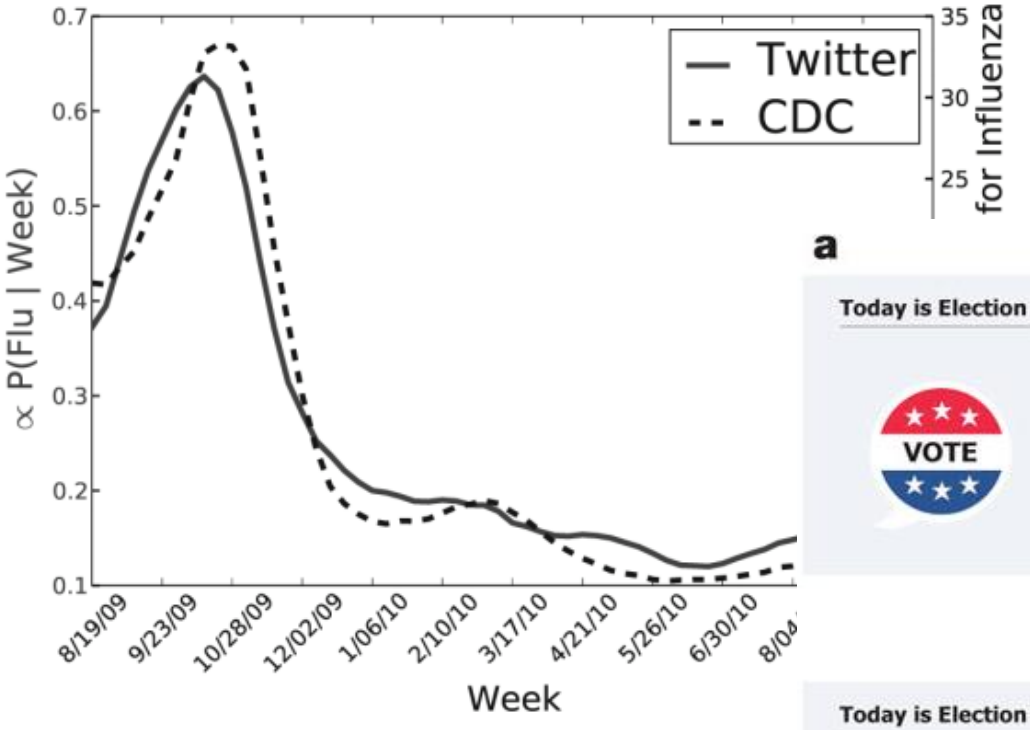
Society

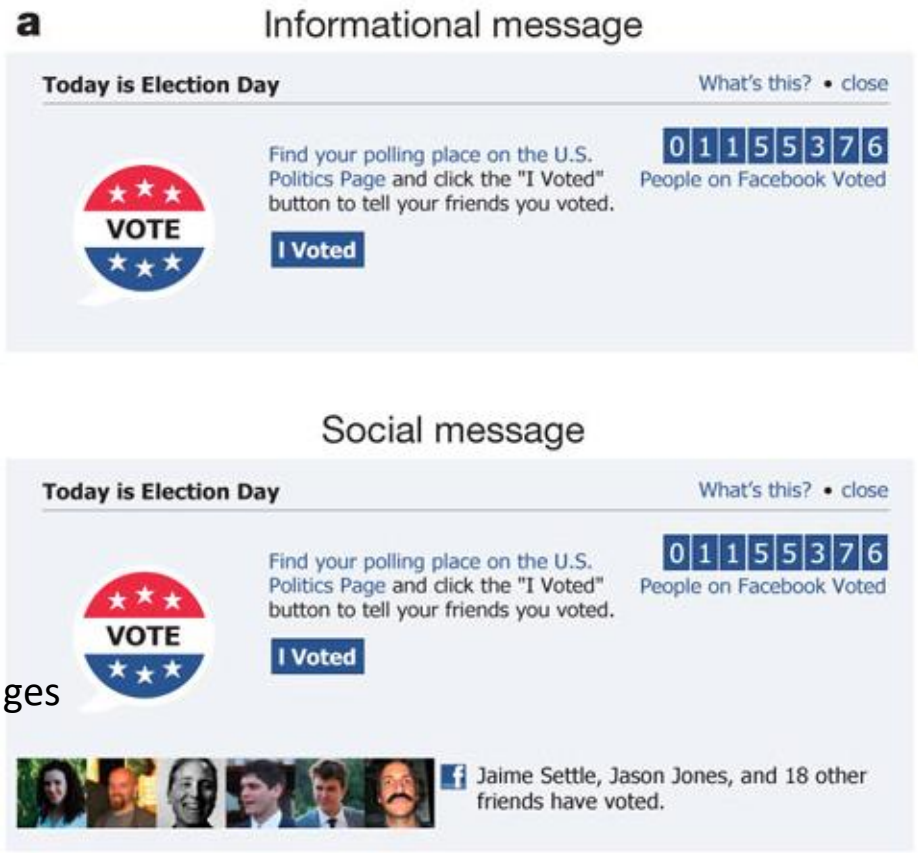
Humans as social sensors

Computational social science

Source: Guardian

Predicting *X* with Twitter





2010 US Midterm Elections:
60m users shown "I Voted" Messages

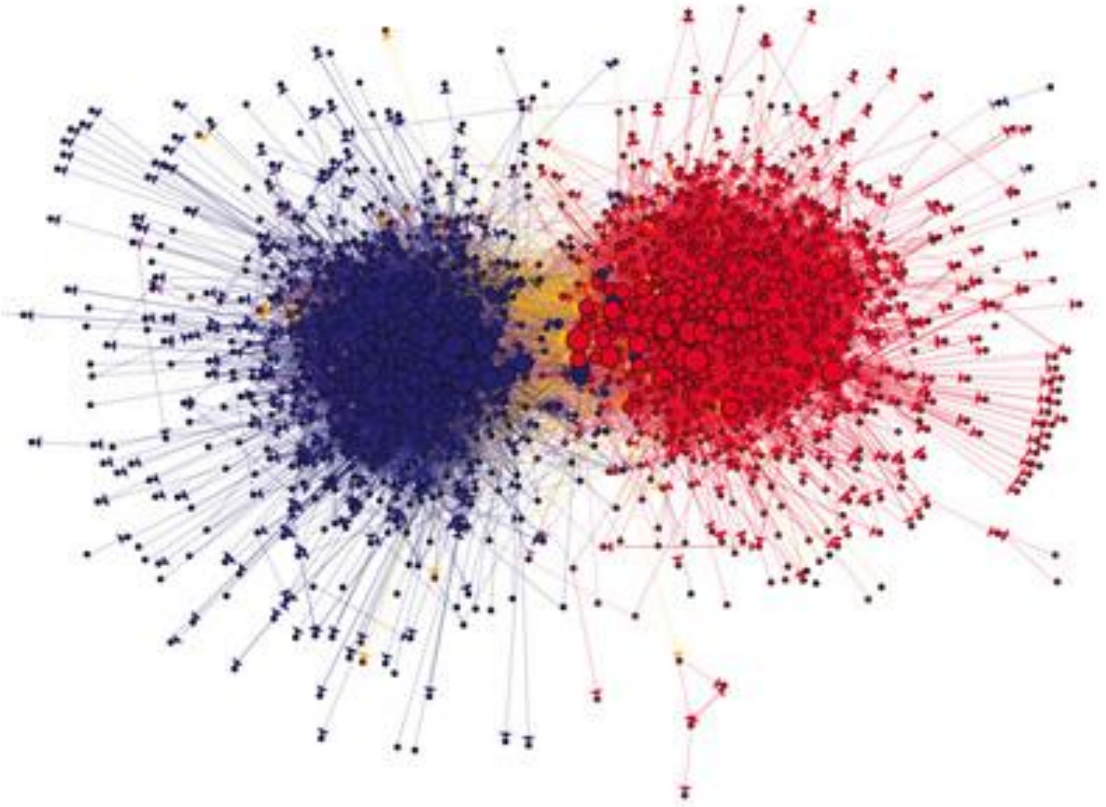
Summary: increased turnout by 60k directly and 280k indirectly

Woah! You should feel unsettled about this!

Political Mobilization on Facebook

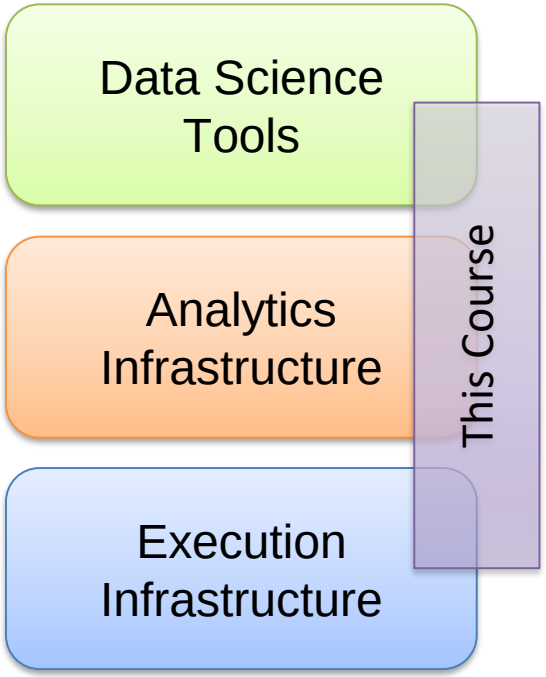
(Paul and Dredze, ICWSM 2011; Bond et al., Nature 2011)

The Political Blogosphere and the 2004 U.S. Election



Source: Popular Internet Meme

What is this course about?

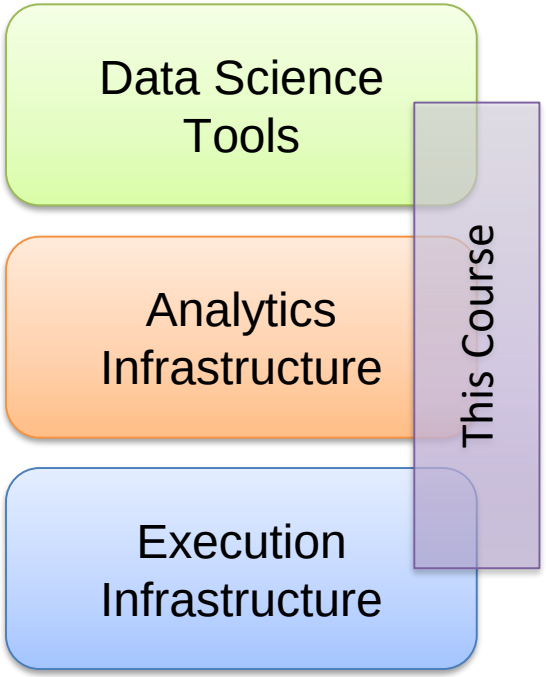


"big data stack"

Buzzwords

data science, data analytics, business intelligence, data warehouses and data lakes

MapReduce, Spark, Flink, Pig, Dryad, Hive, Dryad, noSQL, Pregel, Giraph, Storm/Heron



"big data stack"

Text: frequency estimation, language models, inverted indexes

Graphs: graph traversals, random walks (PageRank)

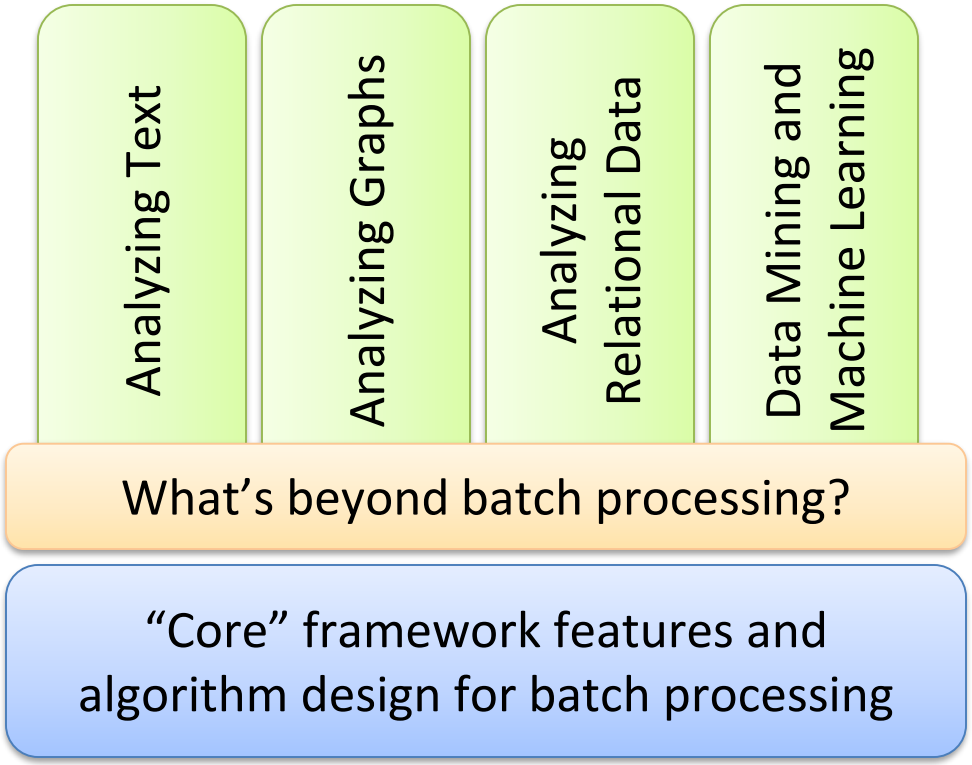
Relational data: SQL, joins, column stores

Data mining: hashing, clustering (*k*-means), classification, recommendations

Streams: probabilistic data structures (Bloom filters, CMS, HLL counters)

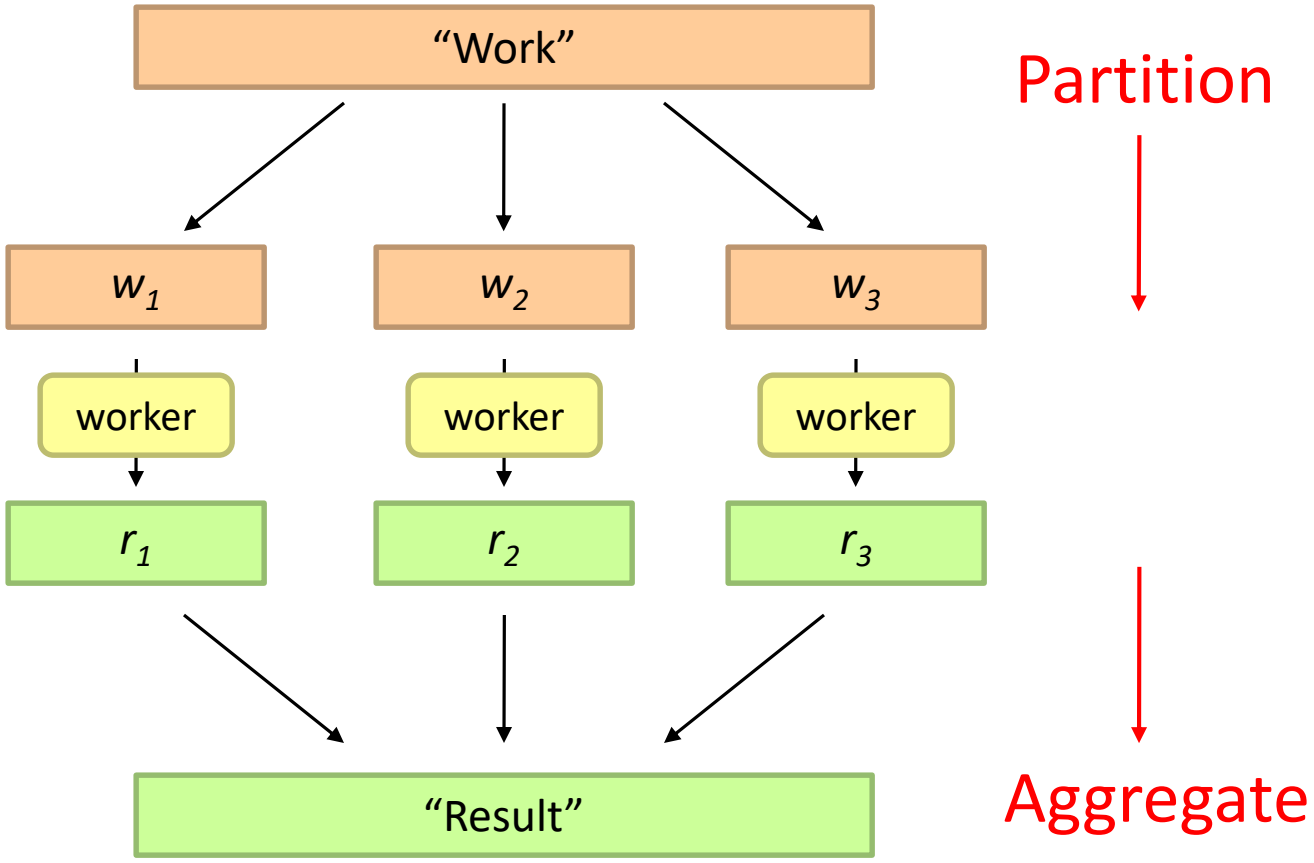
This course focuses on algorithm design and "thinking at scale"

Structure of the Course



Source: Google

Divide and Conquer



Parallelization Challenges

How do we assign work units to workers?
What if we have more work units than workers?
What if workers need to communicate partial results?
What if workers need to access shared resources?
How do we know when a worker has finished? (Or is simply waiting?)
What if workers die?

Difficult because:

We don't know the order in which workers run…
We don't know when workers interrupt each other…
We don't know when workers need to communicate partial results…
We don't know the order in which workers access shared resources…

What's the common theme of all of these challenges?

Common Theme?

Parallelization challenges arise from:

Need to communicate partial results
Need to access shared resources

(In other words, sharing state)

How do we tackle these challenges?

“Current” Tools

Basic primitives

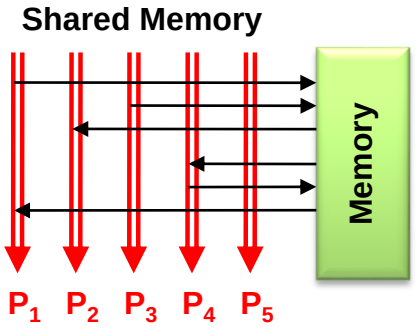
Semaphores (lock, unlock)
Conditional variables (wait, notify, broadcast)
Barriers

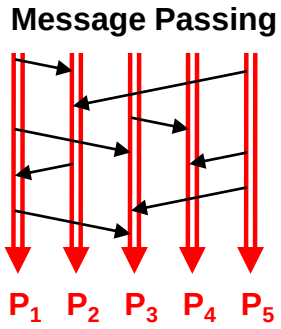
Awareness of Common Problems

Deadlock, livelock, race conditions...
Dining philosophers, sleeping barbers, cigarette smokers...

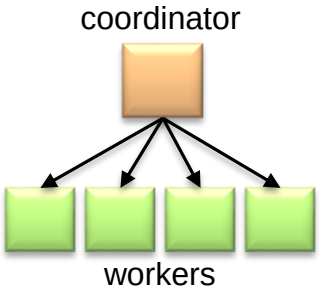
“Current” Tools

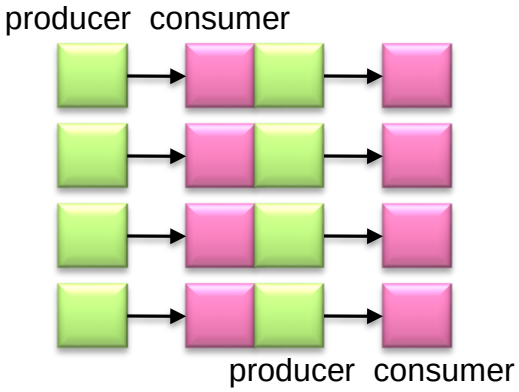
Programming Models

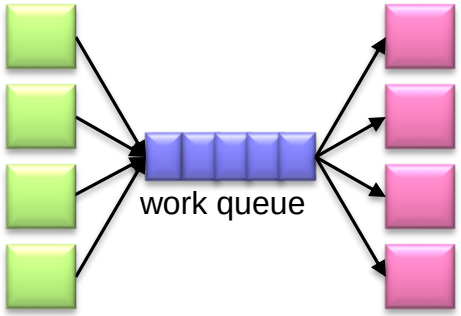




Design Patterns







When Theory Meets Practices

Concurrency is already difficult to reason about…

Now throw in:

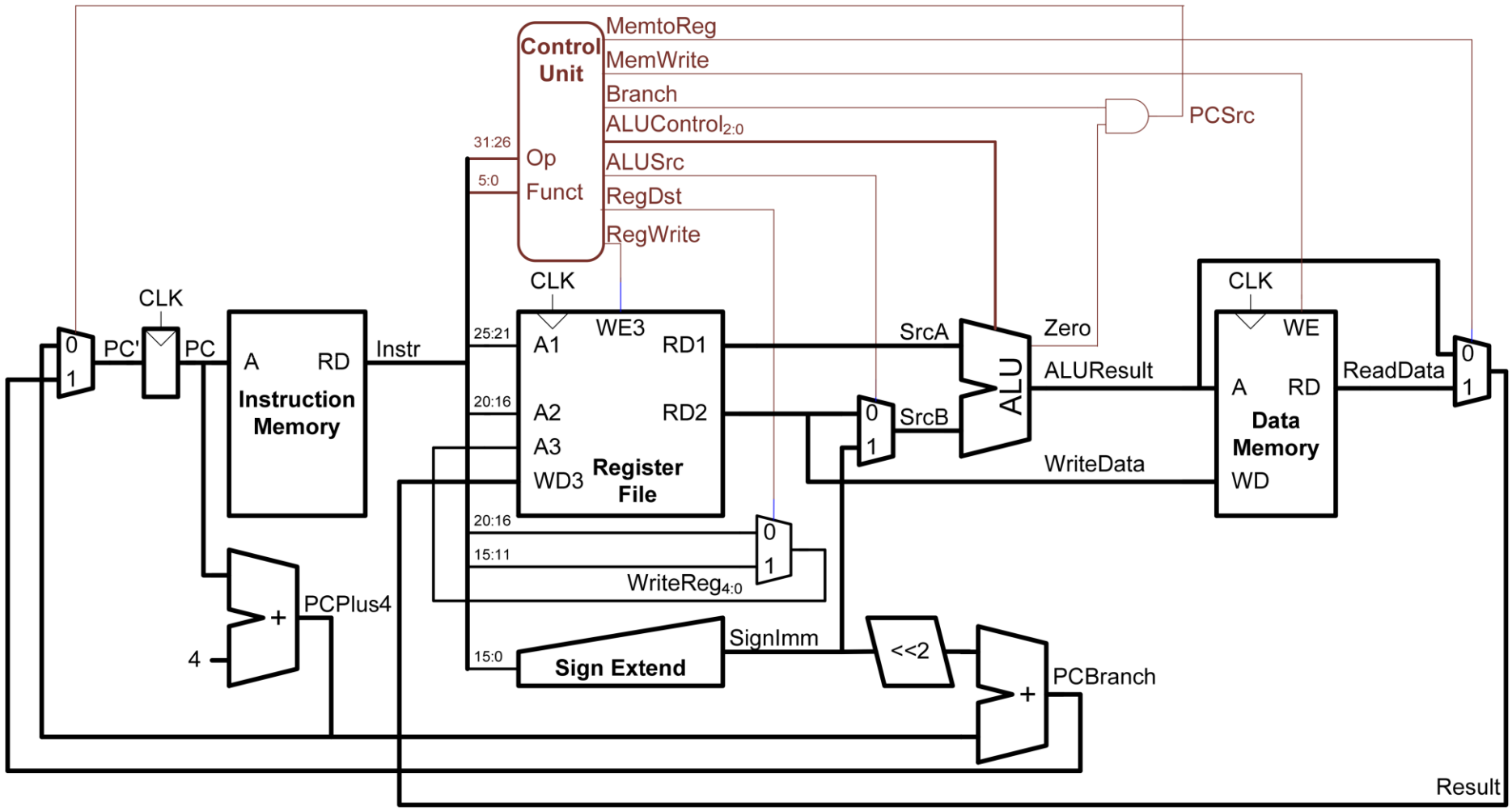
The scale of clusters and (multiple) datacenters
The presence of hardware failures and software bugs
The presence of multiple interacting services

The reality:

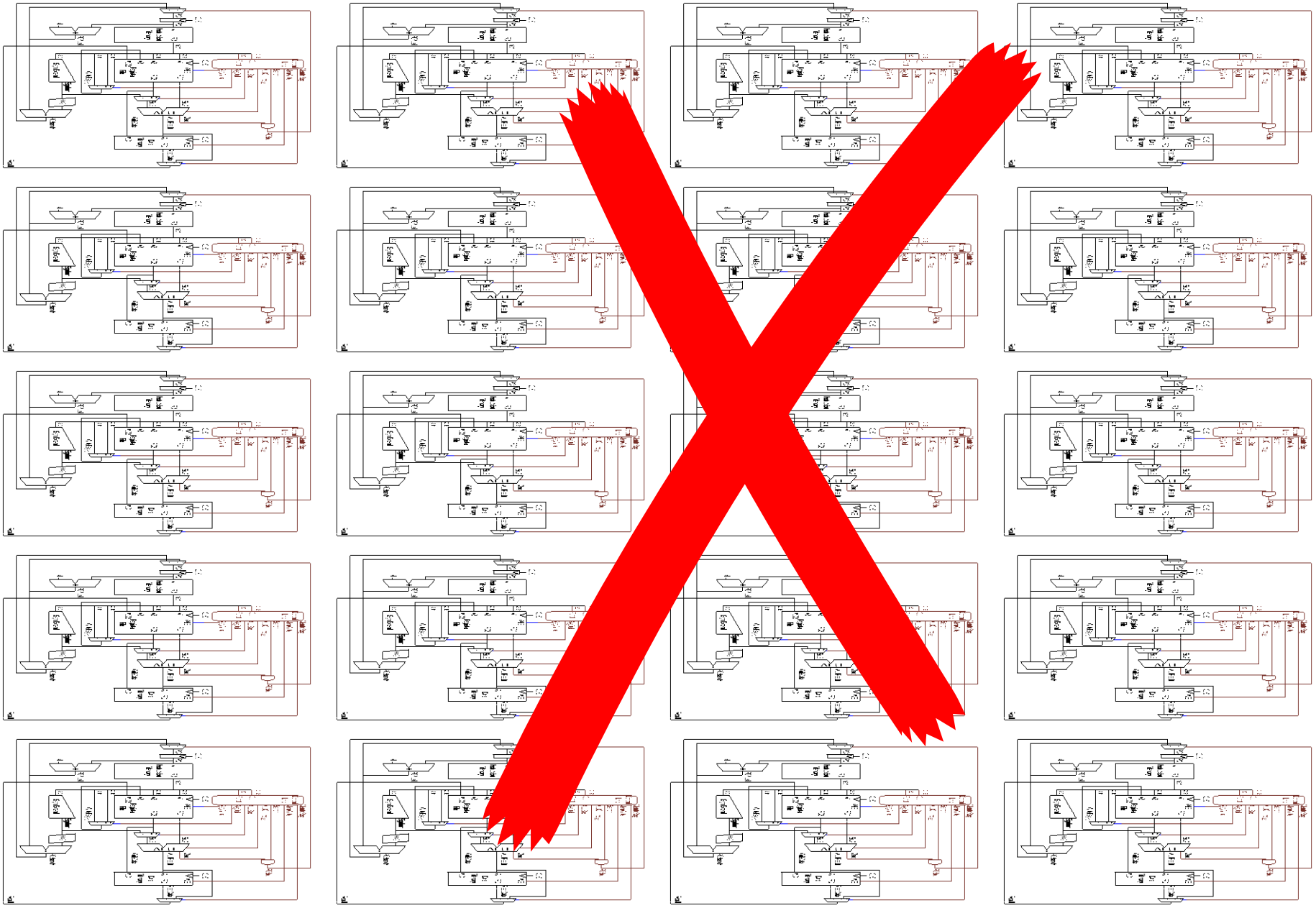
Lots of one-off solutions, custom code
Write you own dedicated library, then program with it
Burden on the programmer to explicitly manage everything

Bottom line: it’s hard!

Source: Ricardo Guimarães Herrmann



Source: CS 251



Source: CS 251

Source: Google

The datacenter *is* the computer!

It's all about the right level of abstraction

Moving beyond the von Neumann architecture

What's the "instruction set" of the datacenter computer?

Hide system-level details from the developers

No more race conditions, lock contention, etc.

No need to explicitly worry about reliability, fault tolerance, etc.

Separating the *what* from the *how*

Developer specifies the computation that needs to be performed

Execution framework ("runtime") handles actual execution

MapReduce is the first instantiation of this idea… but not the last!

Source: Wikipedia (Japanese rock garden)

Course Administrivia

Source: http://www.flickr.com/photos/artmind_etcetera/6336693594/

Four in One!

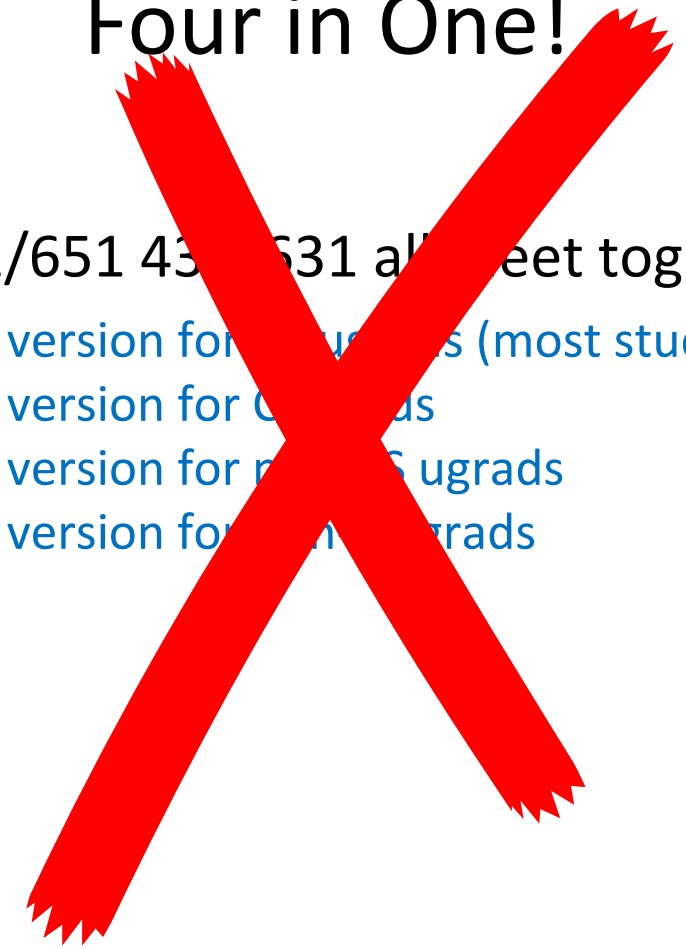
CS 451/651 43[illegible]631 a[illegible]eet together

CS 451: version fo[illegible]s (most students)

CS 651: version for [illegible]ds

CS 431: version for [illegible] ugrads

CS 631: version fo[illegible]rads



Two in One!

CS 451/651

CS 451: version for CS ugrads (most students)

CS 651: version for CS grads

CS 431/631

CS 431: version for non-CS ugrads

CS 631: version for non-CS grads

Course instructors

Ali Abedi: The guy talking right now

TAs: Ryan Clancy, Zheng Ma, Yuqing Xie, Wei Tu, Abdul Naik

Important Coordinates

Course website:
https://www.student.cs.uwaterloo.ca/~cs451/

Lots of info there, read it!
("I didn't see it" will not be accepted as an excuse)

Communicating with us:
Piazza for general/private questions (link on course homepage)

Bespin
http://bespin.io/

Course Design

This course focuses on algorithm design and "thinking at scale"

Not the "mechanics" (API, command-line invocations, et.)

You're expected to pick up MapReduce/Spark with minimal help

Components of the final grade:

6 (CS 431/631) or 8 (CS 451/651) *individual* assignments

Final exam

Additional *group* final project (CS 631/651)

Expectations (CS 451)

Your background:

Pre-reqs: CS 341, CS 348, CS 350
Comfortable in Java and Scala (or be ready to pick it up quickly)
Know how to use Git
Reasonable “command-line”-fu skills
Experience in compiling, patching, and installing open source software
Good debugging skills

You are:

Genuinely interested in the topic
Be prepared to put in the time
Comfortable with rapidly-evolving software

MapReduce/Spark Environments (CS 451)

See “Software” page in course homepage for instructions

Single-Node Hadoop: Linux Student CS Environment

Everything is set up for you, just follow instructions

We’ll make sure everything works

Single-Node Hadoop: Local installations

Install all software components on your own machine

Requires at least 4GB RAM and plenty of disk space

Works fine on Mac and Linux, YMMV on Windows

Important: For your convenience only!

We’ll provide basic instructions, but not technical support

Distributed Hadoop: Datasci Cluster

Assignment Mechanics (CS 451)

We'll be using private GitHub repos for assignments

Complete your assignments, push to GitHub
We'll pull your repos at the deadline and grade

Note late policy (details on course homepage)

Late by up to 24 hours: 25% reduction in grade
Late 24-48 hours: 50% reduction in grade
Late by more the 48 hours: not accepted

By assumption, we'll pull and mark at deadline:
If you want us to hold off, you must let us know!

Important: Register for (free) GitHub educational account!
https://education.github.com/discount_requests/new

Assignment Mechanics (CS 431)

Assignments will use Python and Jupyter

Everything you need to know is in the assignment itself

Assignments will generally be submitted using Git

Details are on the course website for the appropriate assignment

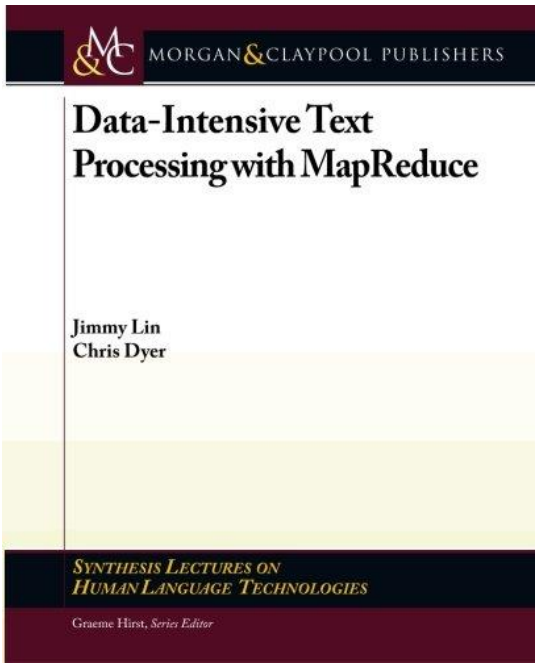
Assignment Mechanics (CS 431)

Note late policy (details on course homepage)
Late by up to 24 hours: 25% reduction in grade
Late 24-48 hours: 50% reduction in grade
Late by more the 48 hours: not accepted

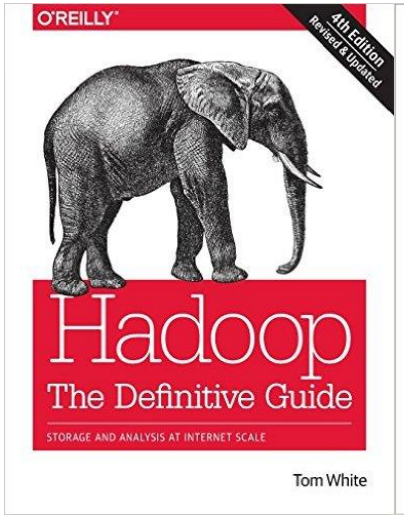
By assumption, we'll pull and mark at deadline:
If you want us to hold off, you must let us know!

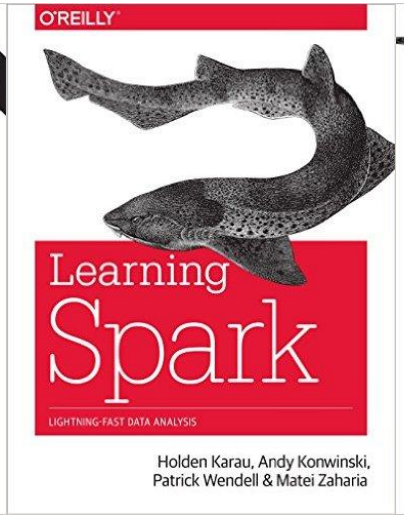
Course Materials

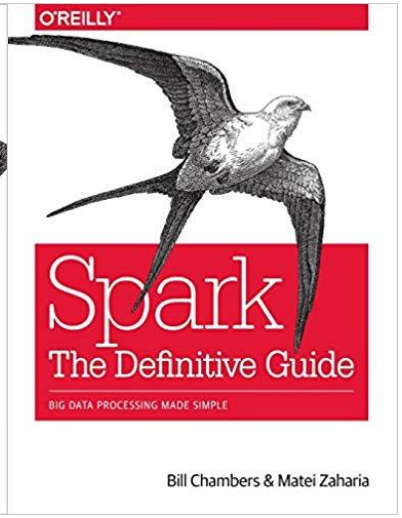
One (required) textbook +
Three (optional but recommended) books +
Additional readings from other sources as appropriate



(optional but recommended)







Note: 4th Edition

If you're not (yet) registered:

Register for the wait list at:

By sending Ali an email at **ali.abedi@uwaterloo.ca**

Priority for unregistered students

CS students

Have all the pre-reqs

Final opportunity to take the course (e.g., 4B students)

Continue to attend class until final decision

Once the course is full, it is *full*

Note: late registration is not an excuse for late assignments