Practical Tips On Handling Big Data

Dr. Brian J. Spiering

hi, brian.

Data Science Faculty @GalvanizeU
@BrianSpiering

Roadmap

1. Defining “Big Data” (aka, you probably don’t have Big Data)
2. How to avoid Big Data (and associated problems)
3. Okay, I really have Big Data. What should I do?

1

Defining “Big Data”
(aka, you probably don’t have Big Data)

What is Big Data?

What is Big Data?

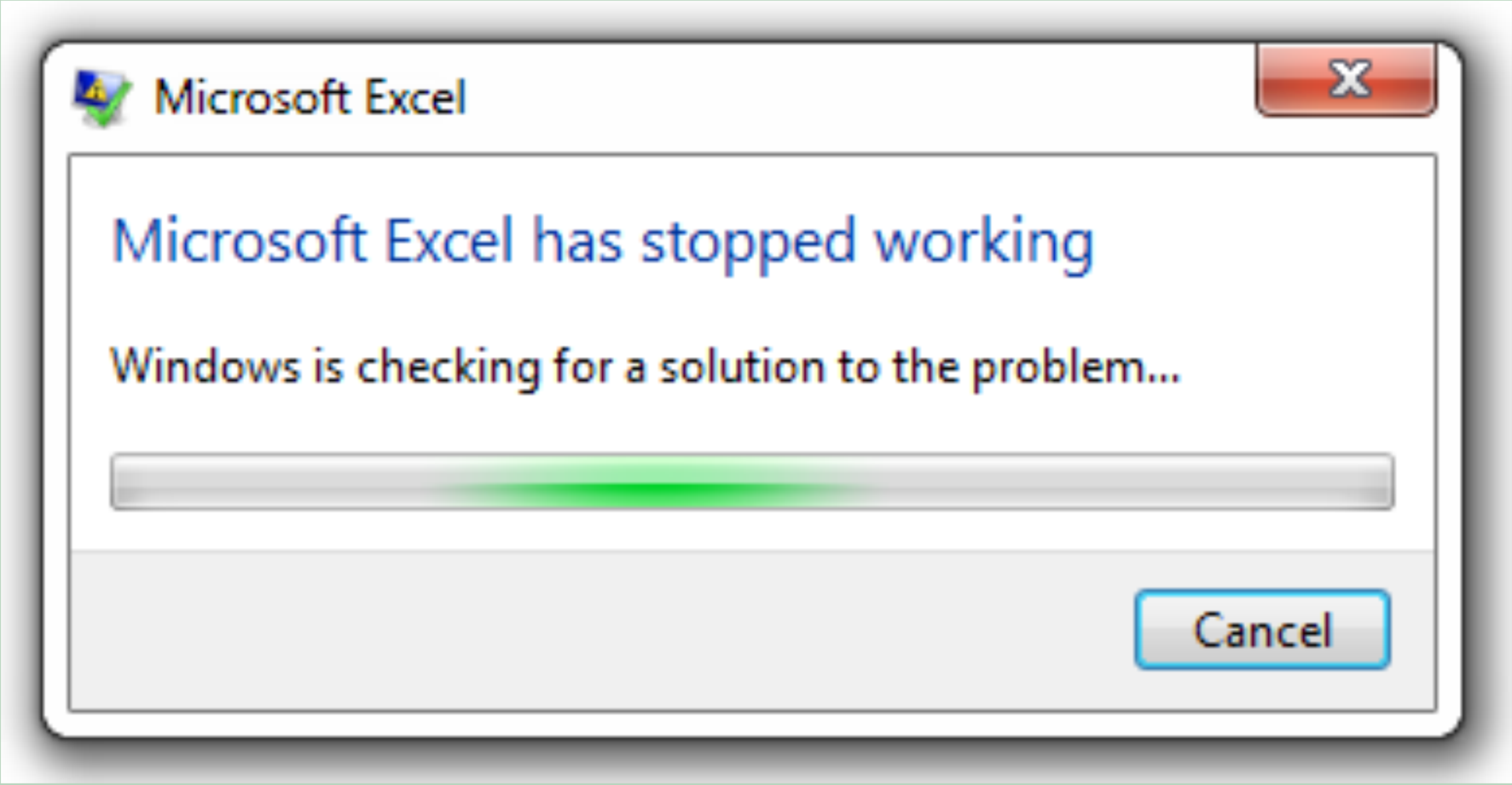
“Data sets with sizes beyond the ability of commonly used software tools to capture, curate, manage, and process data within a tolerable amounts of time.”

What is Big Data?

~~“Data sets with sizes beyond the ability of commonly used software tools to capture, curate, manage, and process data within a tolerable amounts of time.”~~

Data that doesn’t find on a single machine.

What is not Big Data?



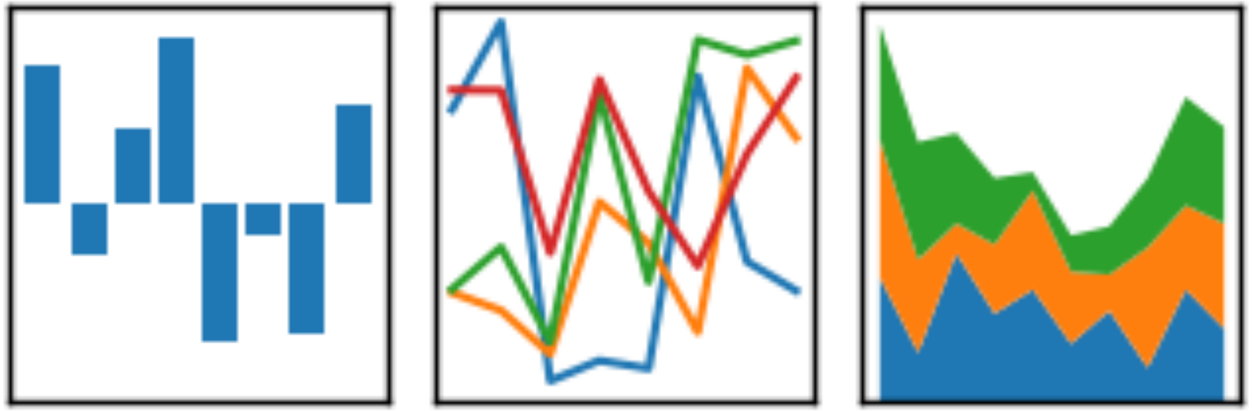
2

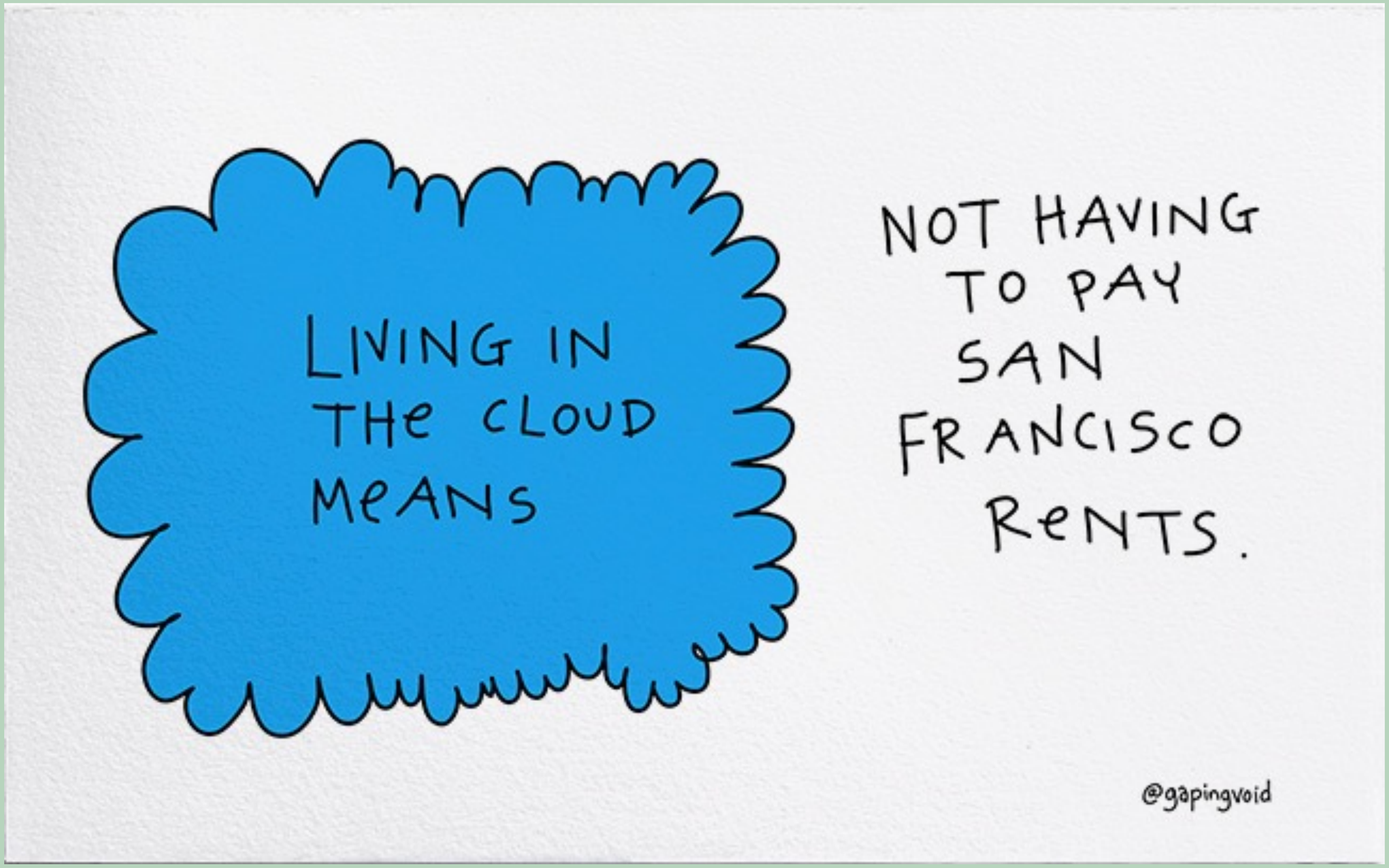
How to avoid Big Data (and associated problems)

Handling Medium Data

pandas

$$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$$




LIVING IN THE CLOUD MEANS
NOT HAVING TO PAY SAN FRANCISCO RENTS.
@gapingvoid

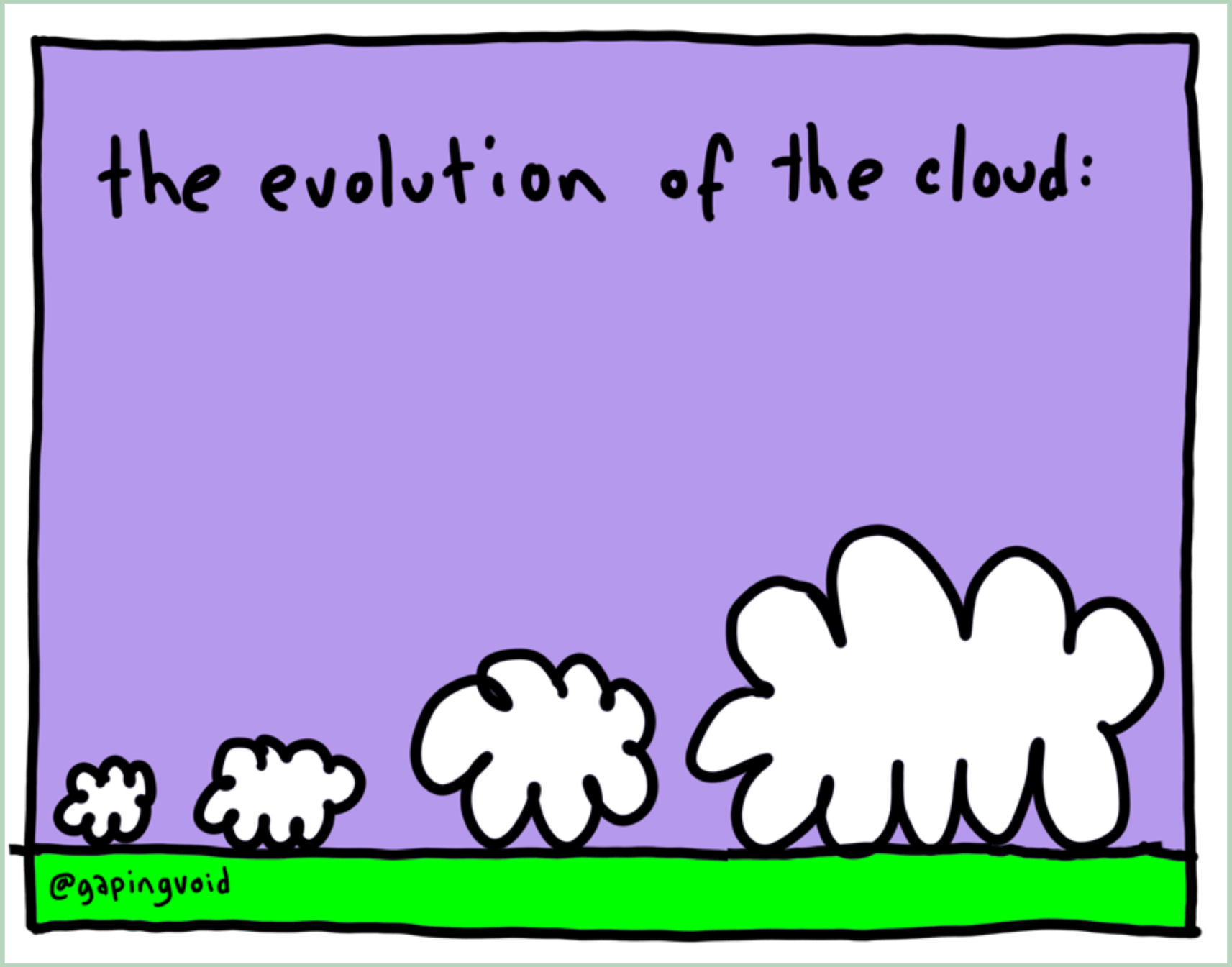
Now available: X1 instances, the largest Amazon EC2 memory-optimized instance with 2 TB of memory

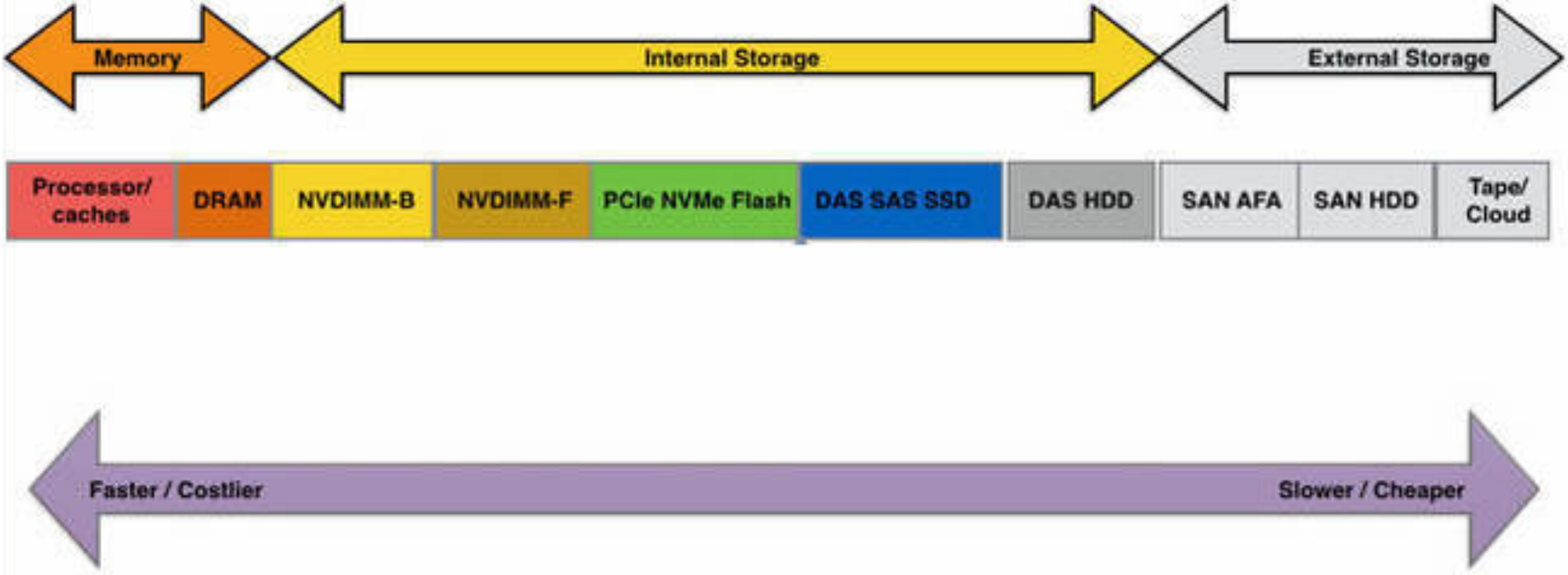
Posted On: May 18, 2016

We are excited to announce Amazon EC2 X1 instances. X1 instances extend the elasticity, simplicity, and cost savings of the AWS cloud to enterprise-grade applications with large dataset requirements. X1 instances are ideal for running in-memory databases like SAP HANA, big data processing engines like Apache Spark or Presto, and high performance computing (HPC) applications. X1 instances are certified by SAP to run production environments of the next-generation Business Suite S/4HANA, Business Suite on HANA (SoH), Business Warehouse on HANA (BW), and Data Mart Solutions on HANA on the AWS cloud.

X1 instances offer 2 TB of DDR4 based memory, 8x the memory offered by any other Amazon EC2 instance. Each X1 instance is powered by four Intel® Xeon® E7 8880 v3 (Haswell) processors and offers 128 vCPUs. In addition, X1 instances offer 10 Gbps of dedicated bandwidth to Amazon Elastic Block Store (Amazon EBS) and are EBS-optimized by default at no additional cost. X1 instances are available with the following specification:

Instance Type	vCPUs	Instance Memory (GiB)	Instance Storage (GB)	Network Bandwidth	Dedicated EBS Bandwidth	3-Year Partial Upfront RI Price per Hour*
x1.32xlarge	128	1,952	2 x 1,920 SSD	10 Gbps	10 Gbps	$3.970


the evolution of the cloud:
@gapingvoid


Memory
Internal Storage
External Storage
Processor/ caches
DRAM
NVDIMM-B
NVDIMM-F
PCIe NVMe Flash
DAS SAS SSD
DAS HDD
SAN AFA
SAN HDD
Tape/ Cloud
Faster / Costlier
Slower / Cheaper

	Nanoseconds (ns)	Microseconds (µs)	Milliseconds (ms)	If L1 Access is 1 second
L1 Cache Reference	0.5			1 sec
L2 Cache Reference	7			14 secs
DRAM Access	100			3 mins, 20 secs
IBM TLC PCM	1,000	1		32 mins, 40 secs
NVDIMM-F (Diablo)	5,000	5		2 hours, 46 mins, 40 secs
Intel Octane 3D XPoint	7,000	7		3 hours, 53 mins, 20 secs
Micron 9100 NVMe PCIe SSD Write	30,000	30		16 hours, 40 mins
Mangstor NX NVMeF Array Write	30,000	30		16 hours, 40 mins
DSSD D5 NVMeF Array	100,000	100		2 days, 7 hours, 33 mins, 20 secs
Mangstor NX NVMeF Array Read	110,000	110		2 days, 13 hours, 6 mins, 40 secs
NVMe PCIe SSD Read	110,000	110		2 days, 13 hours, 6 mins, 40 secs
Micron 9100 NVMe PCIe SSD Read	120,000	120		2 days, 18 hours,m40 mins
Random 4K read from SSD	150,000	150		3 days, 11 hours, 20 mins
IBM FlashSystem A9000	250,000	250		5 days, 18 hours, 53 mins, 20 secs
Sequential 1MB Read from DRAM	250,000	250		
Round Trip in Data Centre	500,000	500	0.5	11 days, 13 hours, 46 mins, 40 secs
Sequential 1MB Read from SSD	1,000,000	1,000	1	23 days, 3 hours, 33 mins, 20 secs
IBM DS8888 - minimum less than	1,000,000	1,000	1	
Isilon node minimum	5,000,000	5,000	5	115 days, 17 hours, 46 minutes, 40 secs
Disk Seek	10,000,000	10,000	10	231 days, 11 hours, 33 mins, 20 secs
Sequential 1MB Read from Disk	20,000,000	20,000	20	1 year, 97 day.s 23 hours, 6 mins, 40 secs
DAS Disk Access	100,000,000	100,000	100	6 years, 119 days, 19 hours, 33 mins, 20 secs
Send Packet CA->Netherlands->CA	150,000,000	150,000	150	9 years, 185 days, 5 hours, 20 mins
SAN Array Access	300,000,000	300,000	300	19 years, 5 days, 10 hours, 40 mins

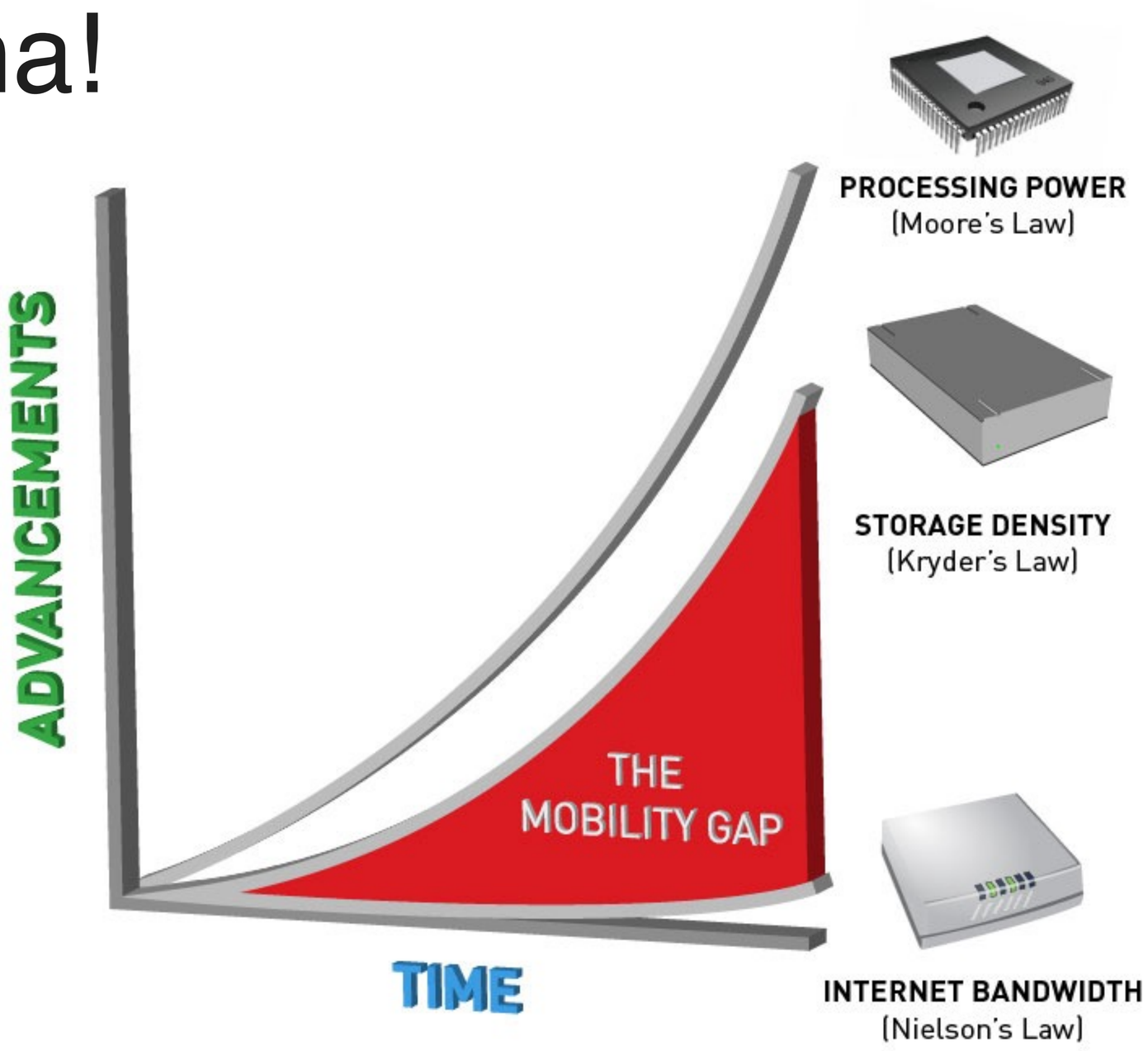
Cache

RAM

Disk

Data Center

Big Data Gotcha!



Scaling Out

1. Single Local Machine < 10s GB*
2. Single Cloud Machine < 2 TB*
3. Cloud Cluster of Machines > 2 TB*

* Summer 2016


Home
Moments
Notifications
Messages
Search Twitter
Justin Basilico
@JustinBasilico
Follow
"Math is a lot cheaper than hardware" - @smolix at #DataSmt
RETWEET
1
LIKE
1
1:57 PM - 20 Jul 2015
1
1
Reply to @JustinBasilico @smolix
Justin Bas
@JustinBasilico
Machine learning
@Netflix, recomm
@AlgoFoundry co
time travel
Joined Septer
Trends
#AllStarGame #insideintercom #GlockABook #berkmtg #DCXSanFran #resilience Jon Lester Windsor Jennifer Aniston #AJUNDebate
© 2016 Twitter About Help Terms Privacy Cookies Ads info

Matrix Multiplication

$$c_{ij} = \sum_{k=1}^{n} a_{ik} \cdot b_{kj} = a_{i1} \cdot b_{1j} + a_{i2} \cdot b_{2j} + \ldots + a_{in} \cdot b_{nj}$$

column j

row i $\hookrightarrow$

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \ldots & a_{1n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{i1} & a_{i2} & a_{i3} & \ldots & a_{in} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \ldots & a_{nn} \end{bmatrix} \cdot \begin{bmatrix} b_{11} & b_{12} & \ldots & b_{1j} & \ldots & b_{1n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ b_{i1} & b_{i2} & \ldots & b_{ij} & \ldots & b_{in} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \ldots & b_{nj} & \ldots & b_{nn} \end{bmatrix} =$$

$$= \begin{bmatrix} c_{11} & c_{12} & \ldots & c_{1j} & \ldots & c_{1n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ c_{i1} & c_{i2} & \ldots & \mathbf{c_{ij}} & \ldots & c_{in} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ c_{n1} & c_{n2} & \ldots & c_{nj} & \ldots & c_{nn} \end{bmatrix}$$

entry on row i
column j

Matrix Multiplication: Imperative Implementation

```
A = [[2, 3], [3, 5]]
B = [[1, 2], [5, -1]]

rows_A = len(A)
cols_A = len(A[0])
rows_B = len(B)
cols_B = len(B[0])

C = [[0 for row in range(cols_B)] for col in range(rows_A)]

for i in range(rows_A):
        for j in range(cols_B):
            for k in range(cols_A):
                C[i][j] += A[i][k] * B[k][j]
print(C)
[[17, 1], [28, 1]]
```

Matrix Multiplication: Functional Implementation

```
import numpy as np

A = np.matrix( ((2,3), (3, 5)) )
B = np.matrix( ((1,2), (5, -1)) )

A*B
matrix([[17,  1],
        [28,  1]])
```

Matrix Multiplication

$$c_{ij} = \sum_{k=1}^{n} a_{ik} \cdot b_{kj} = a_{i1} \cdot b_{1j} + a_{i2} \cdot b_{2j} + \ldots + a_{in} \cdot b_{nj}$$

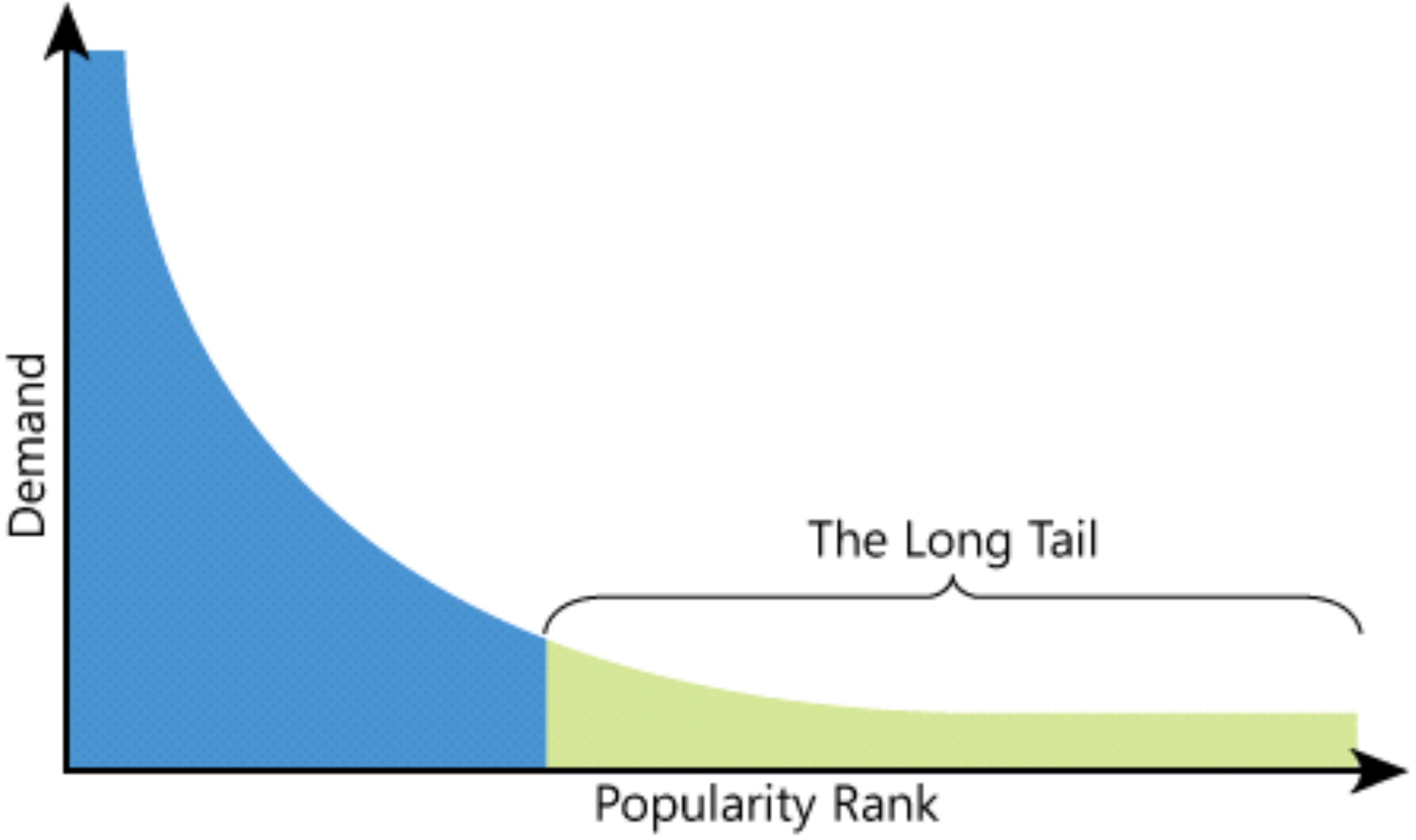
column j

row i $\hookrightarrow$

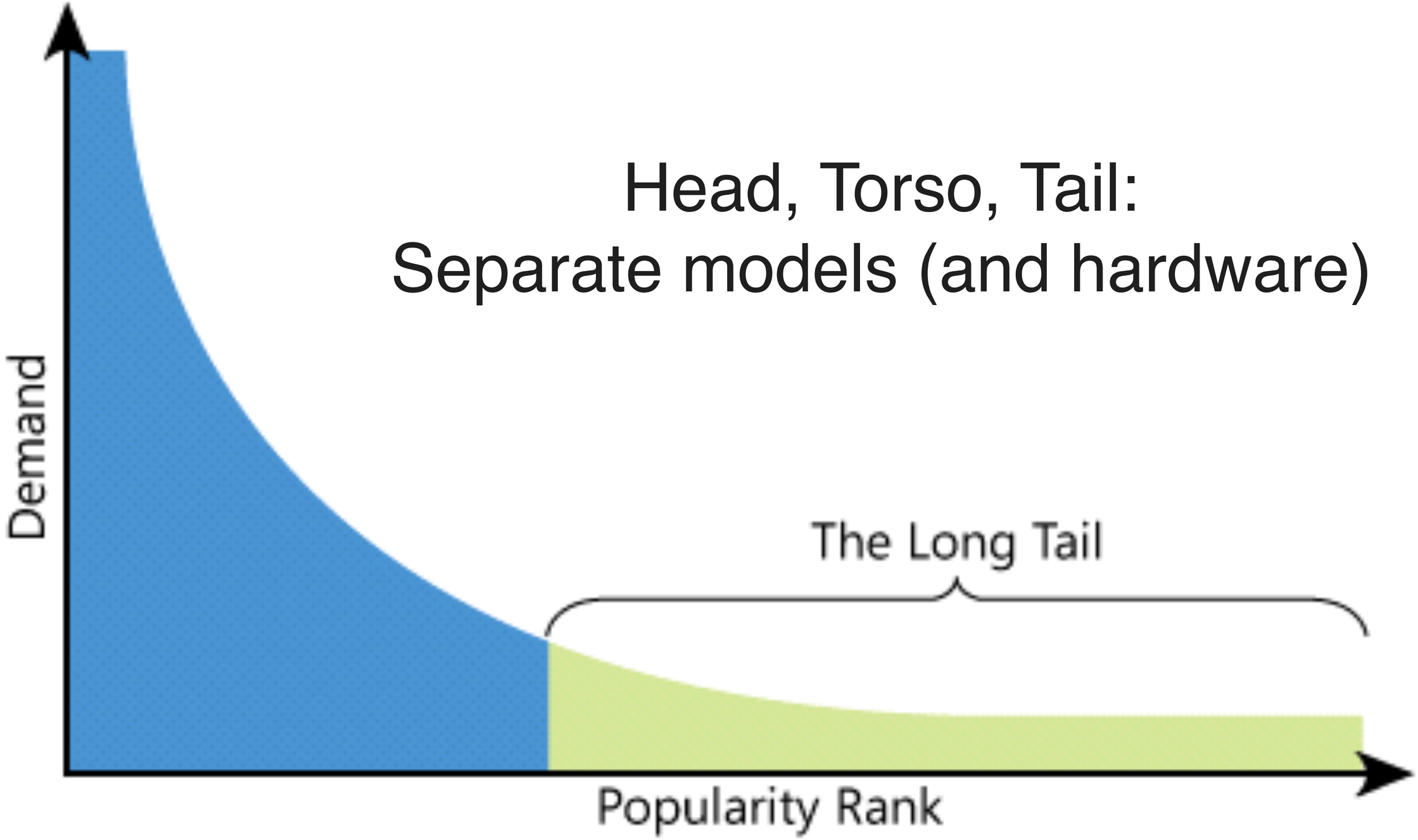
$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \ldots & a_{1n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{i1} & a_{i2} & a_{i3} & \ldots & a_{in} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \ldots & a_{nn} \end{bmatrix} \cdot \begin{bmatrix} b_{11} & b_{12} & \ldots & b_{1j} & \ldots & b_{1n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ b_{i1} & b_{i2} & \ldots & b_{ij} & \ldots & b_{in} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \ldots & b_{nj} & \ldots & b_{nn} \end{bmatrix} =$$

$$= \begin{bmatrix} c_{11} & c_{12} & \ldots & c_{1j} & \ldots & c_{1n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ c_{i1} & c_{i2} & \ldots & \mathbf{c_{ij}} & \ldots & c_{in} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ c_{n1} & c_{n2} & \ldots & c_{nj} & \ldots & c_{nn} \end{bmatrix}$$

entry on row i
column j


Demand
The Long Tail
Popularity Rank

Head, Torso, Tail:
Separate models (and hardware)



3

Okay, I really have Big Data. What should I do?

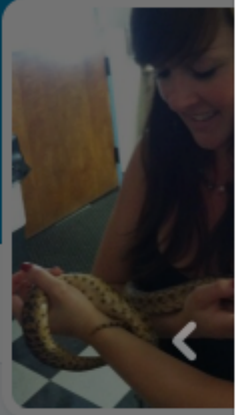
“But my data is more than 5TB!
(and I need it in memory)”

“But my data is more than 5TB!
(and I need it in memory)”

Your life sucks now…
You are stuck with
distributed computing


APACHE
Spark
TM

Home Moments Notifications Messages Search Twitter



Ellie McDo
@EllieMcDonagh

Joined Novem

Ellie McDonagh
@EllieMcDonagh

Follow

Robert Gentleman, Genentech: "make big data as small as possible as quick as is possible" to enable sharing #bigdatamed

RETWEETS 15 LIKES 13

11:34 AM - 21 May 2014

15 13

Reply to @EllieMcDonagh

Trends

#AllStarGame #GlockABook #WTUS16 #MasterChefBR #RyanTownHall David Ortiz
#Calistoga Jennifer Aniston #UNSGcandidates O Canada

Big Data is functional

map

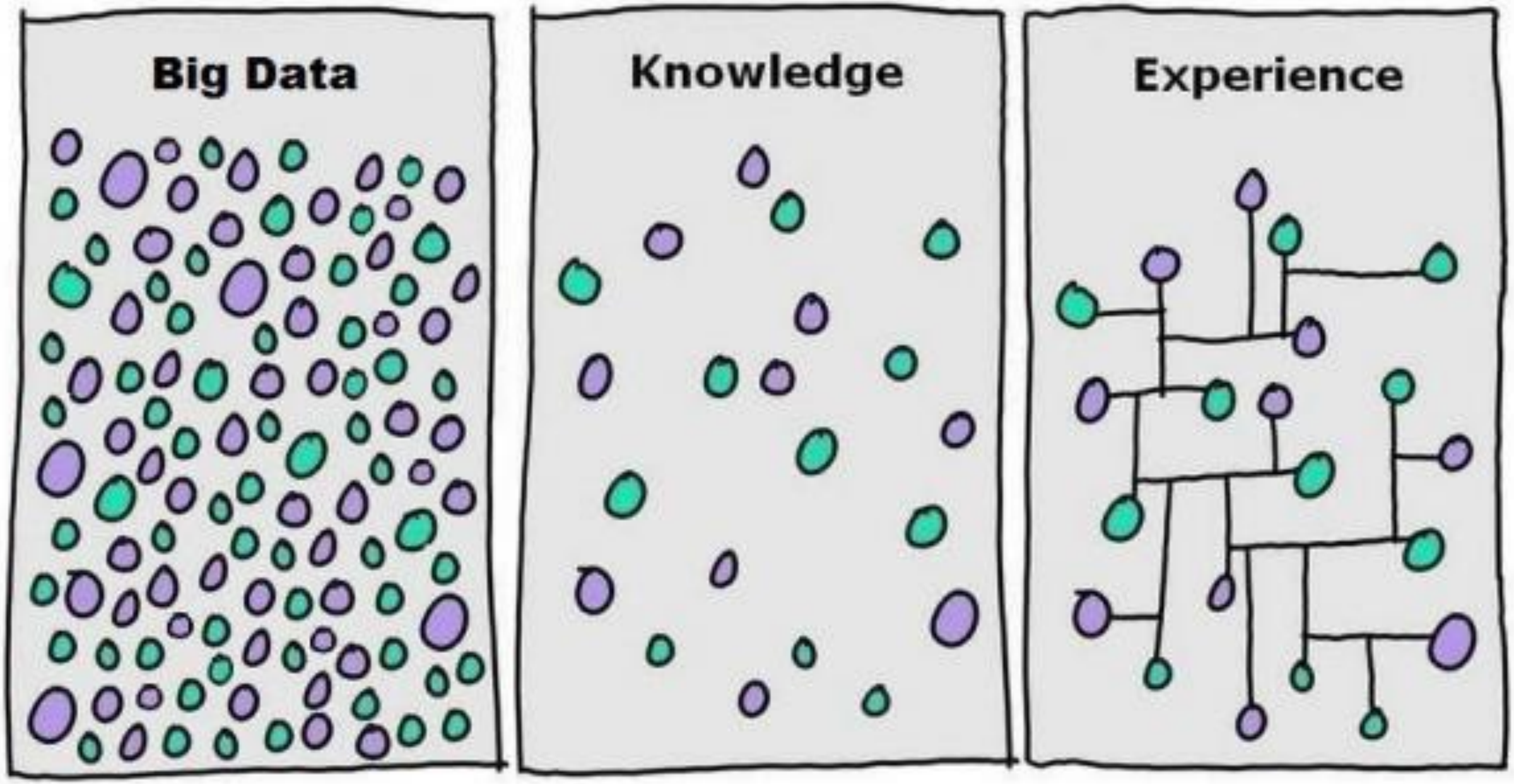
```
from numpy import square

list(map(square, [1, 2, 3]))
[1, 4, 9]
```

reduce

```
from functools import reduce
import operator as op

reduce(op.add, [1, 2, 3])
6
```


Big Data
Knowledge
Experience

What to do:

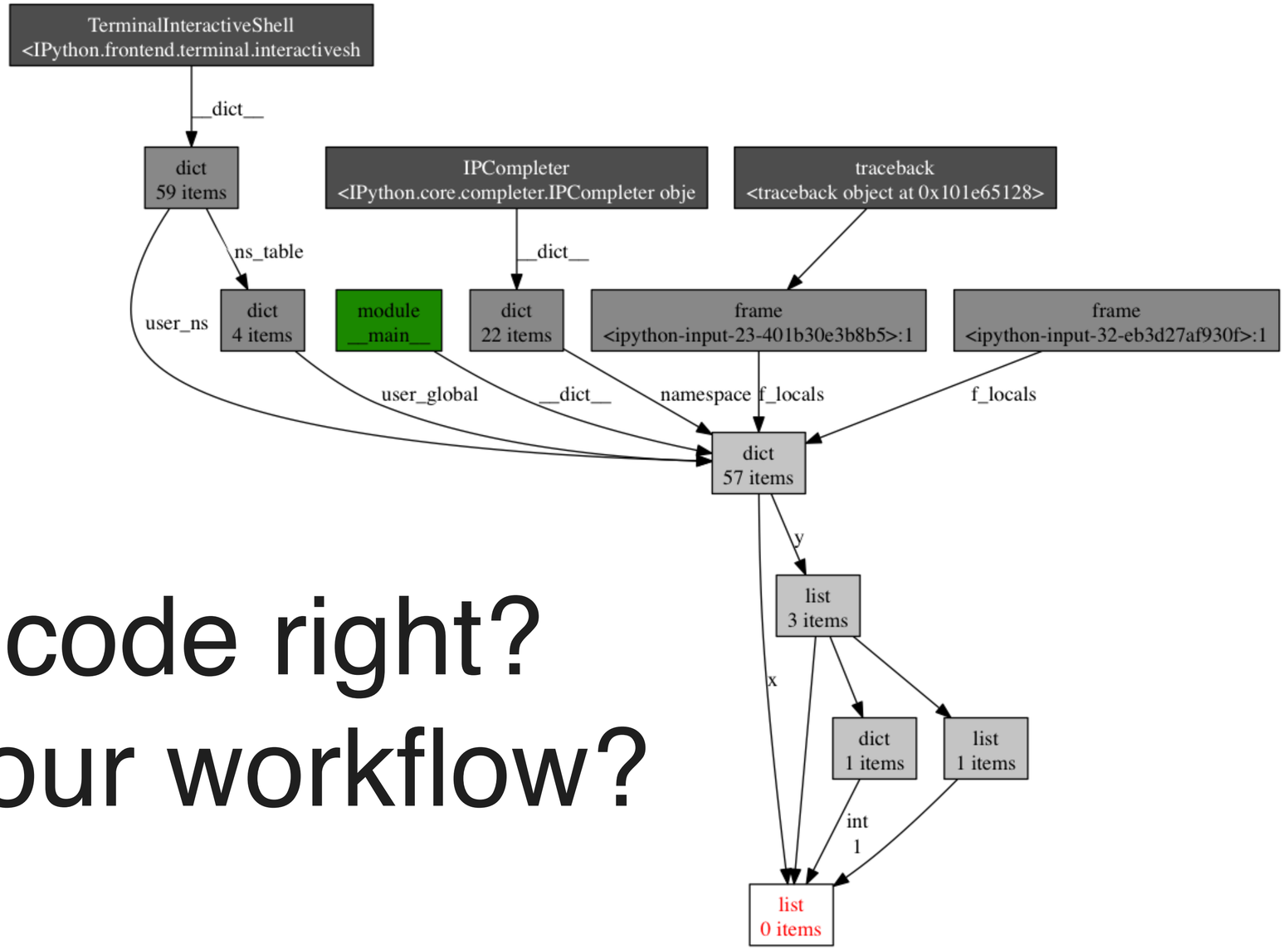
1. Learn some math tricks (linear algebra)
2. Learn how to optimize your code
3. Learn how to use cloud compute
4. Learn a Big Data Framework

Where have we been?

1. Defining “Big Data” (aka, you probably don’t have Big Data)
2. How to avoid Big Data (and associated problems)
3. Okay, I really have Big Data. What should I do?

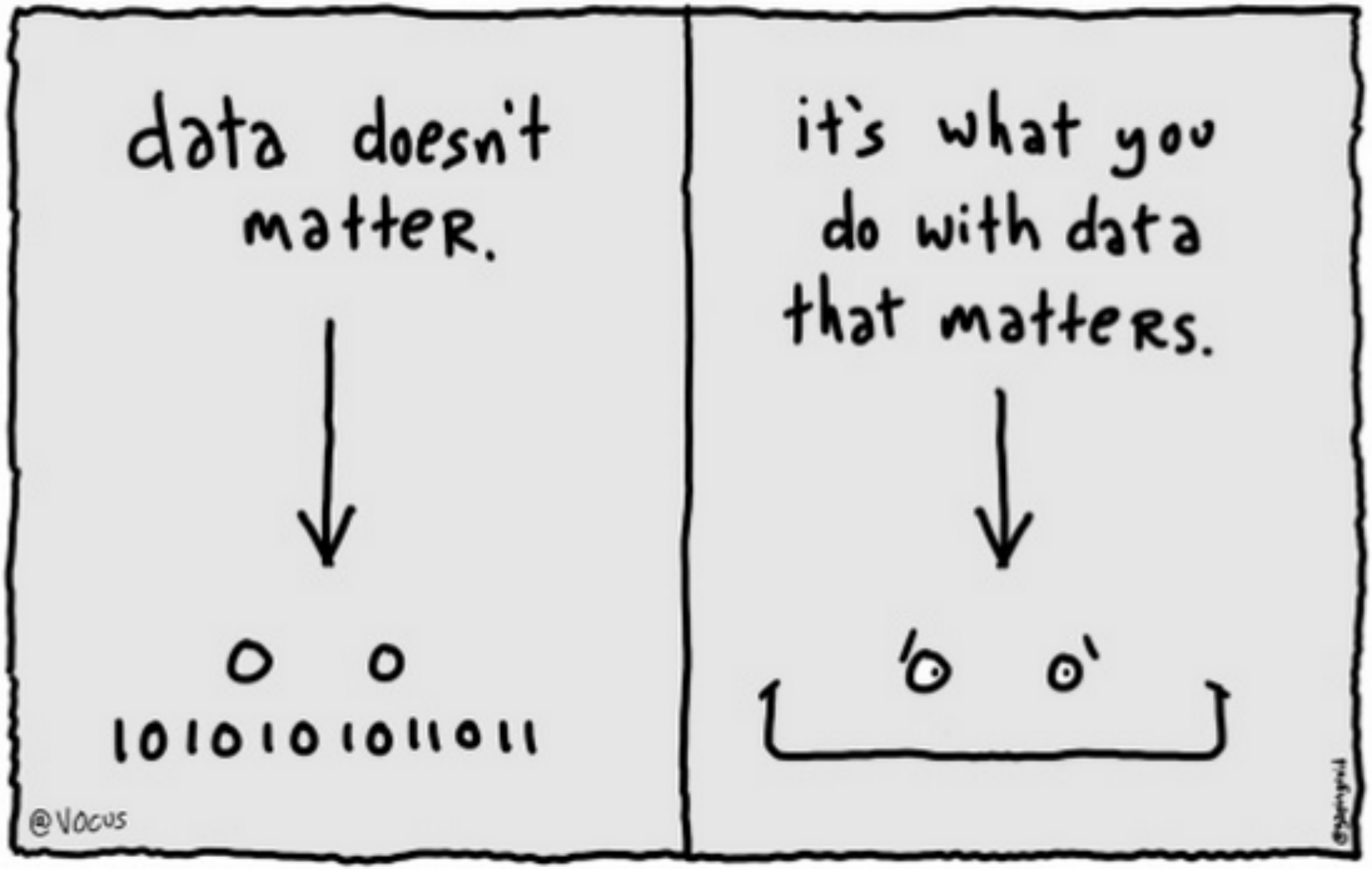
Thank You!
Questions?

drop me a line.



You profile your code right?
Do you profile your workflow?

Do you have the data?
Do you have access to the data?
Do you have a use case for the data?


data doesn't matter.
0 0
1010101011011
@Vocus
it's what you do with data that matters.
0 0

The 5 I’s for new technology:

1. Information - What is it?
2. Inspiration - Why is it important?
3. Install - How do I set it up?
4. Implementation - How do I use it?
5. Integration - How does it work with what else I already use?