Scalability

David Wihl
Harvard CS109B, Spring 2018
Feb 28, 2018

Outline

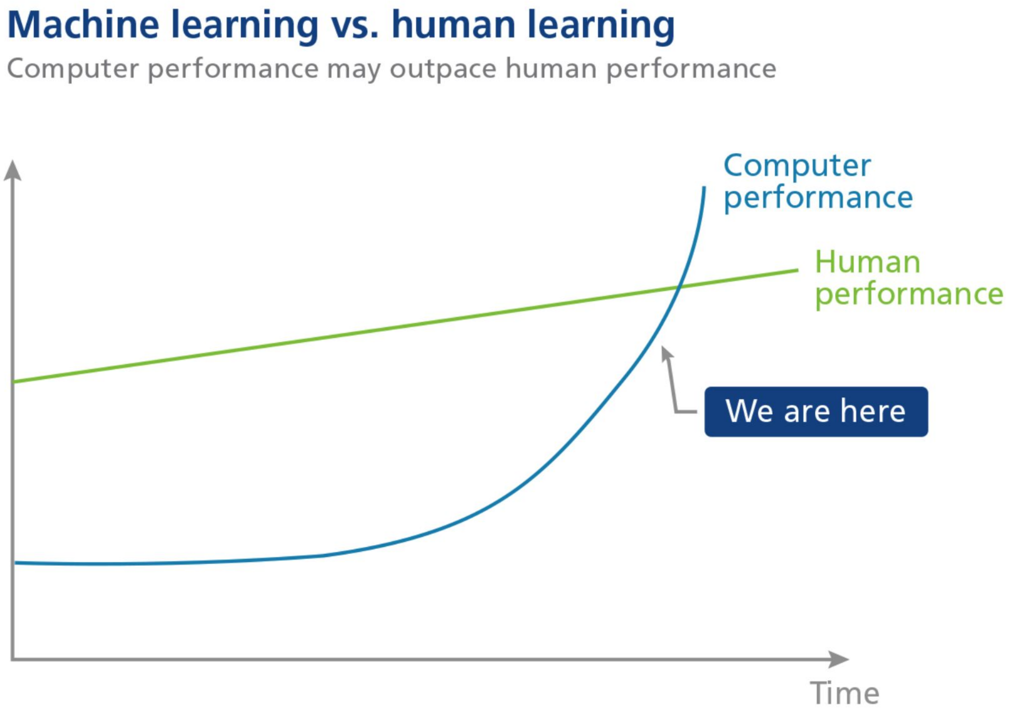
- Why Scalability is Important
- Scale up - in-machine scalability
- Scale out - “Data Center as Computer”
- Future Directions

Preparation for Deep Learning

Data Size > RAM memory

Compute Requirements > Hours of CPU time

Current methods are sample inefficient



Interesting Problems in Computer Science

P=NP?

Cryptography / One way functions

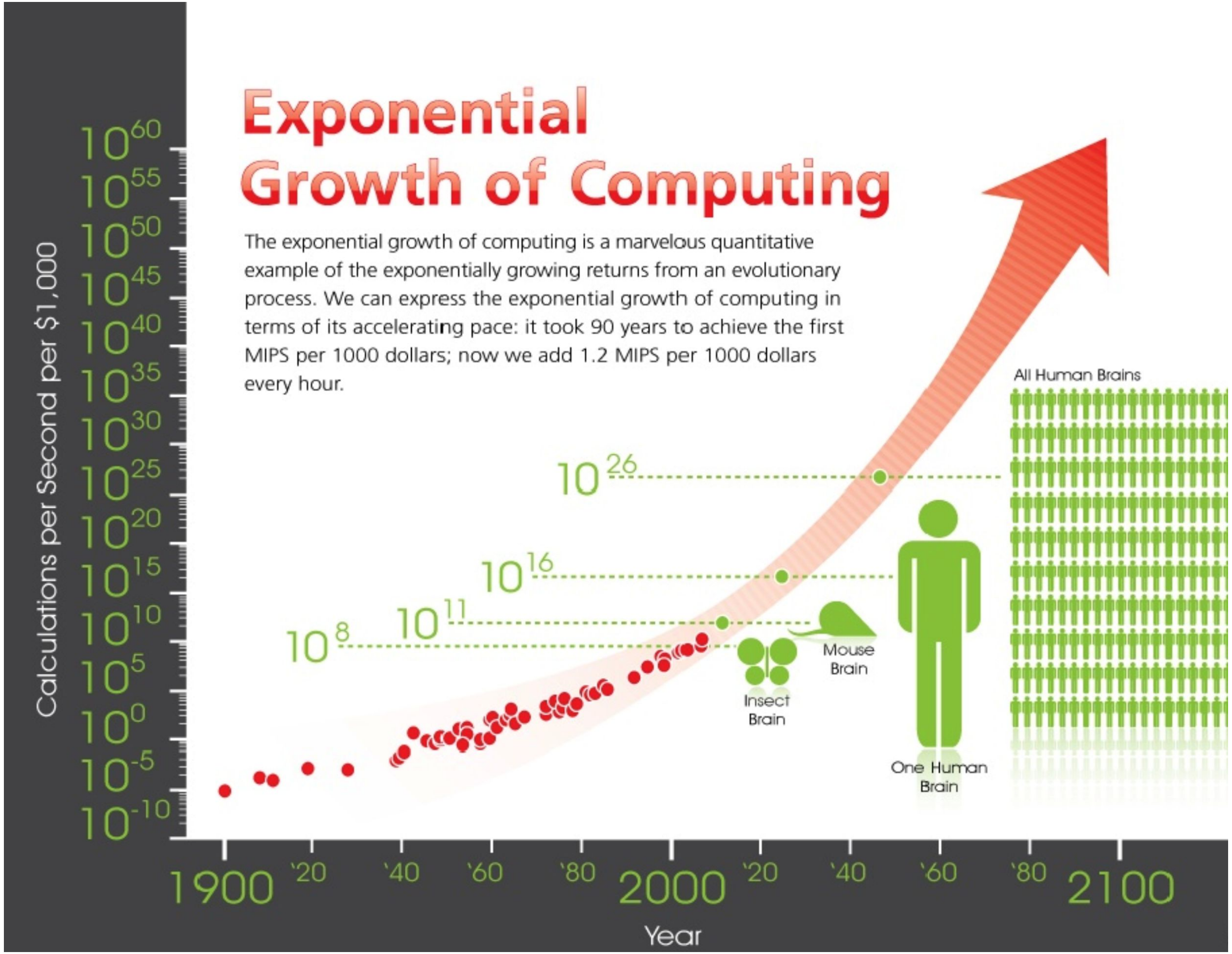
Machine Learning, including analysis of large datasets

Non-numeric information: images, music / sounds, text analysis

Global Challenges

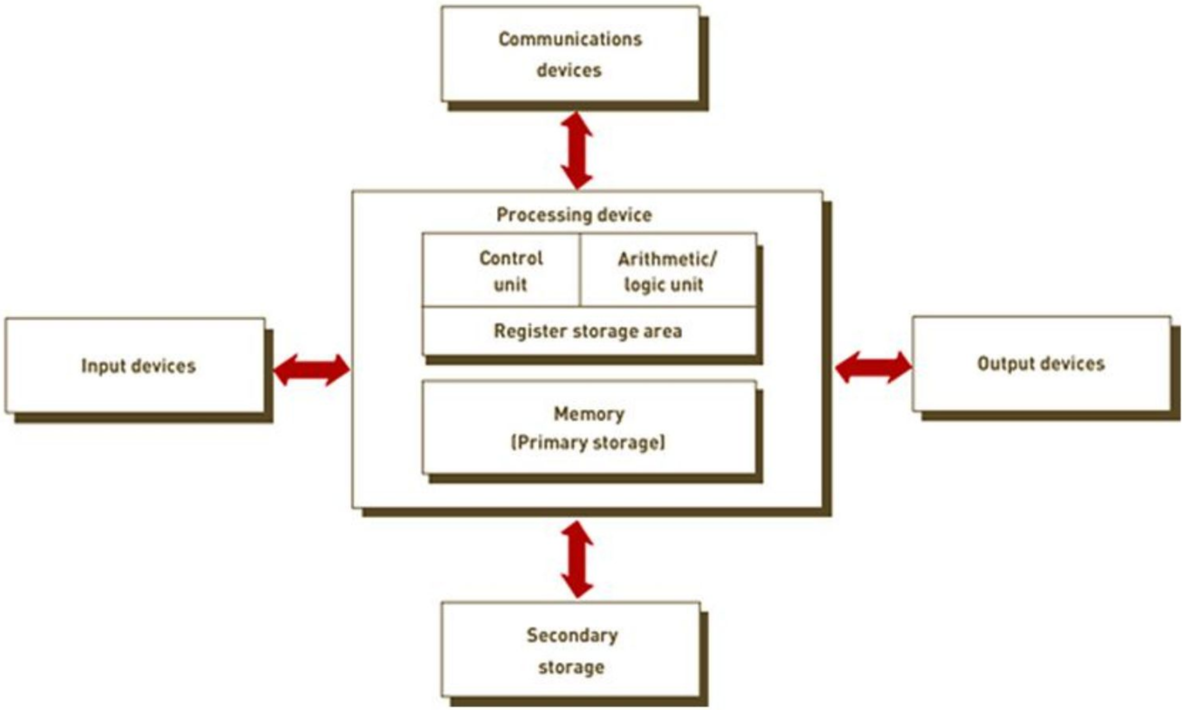
Bio / genomics

Environmental, including energy

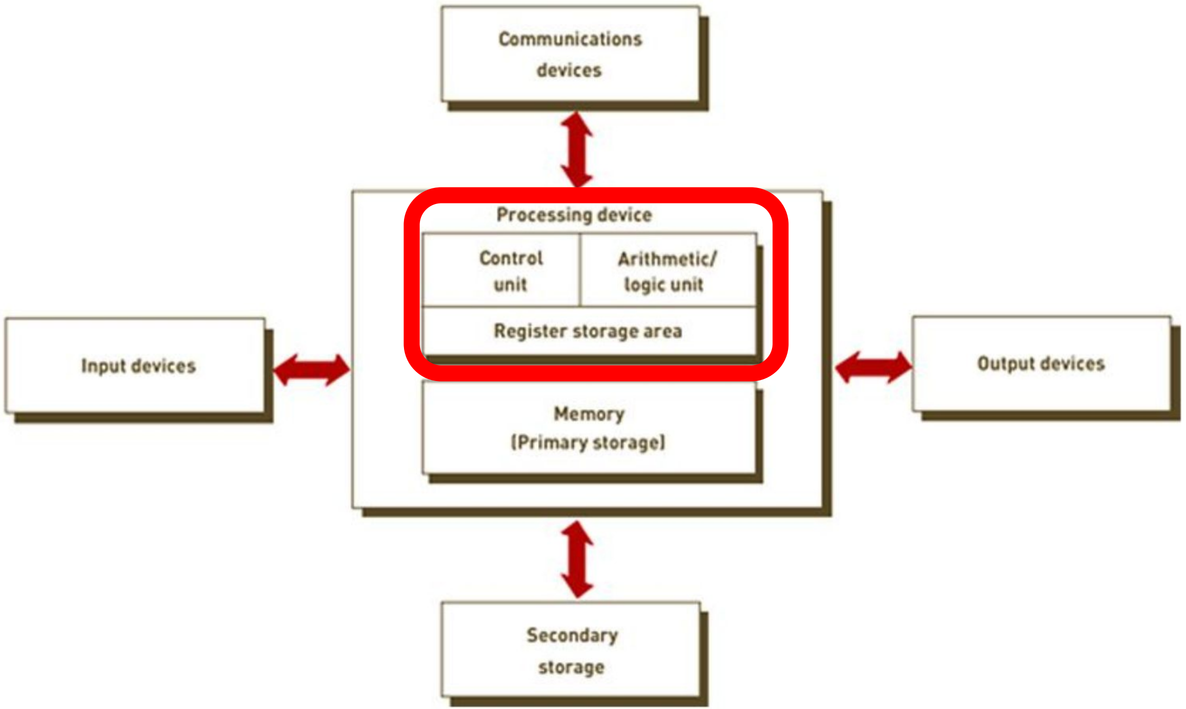


Exponential Growth of Computing
The exponential growth of computing is a marvelous quantitative example of the exponentially growing returns from an evolutionary process. We can express the exponential growth of computing in terms of its accelerating pace: it took 90 years to achieve the first MIPS per 1000 dollars; now we add 1.2 MIPS per 1000 dollars every hour.
Calculations per Second per $1,000
10^{60}
10^{55}
10^{50}
10^{45}
10^{40}
10^{35}
10^{30}
10^{25}
10^{20}
10^{15}
10^{10}
10^{5}
10^{0}
10^{-5}
10^{-10}
10^{26}
10^{16}
10^{11}
10^{8}
Insect Brain
Mouse Brain
One Human Brain
All Human Brains
1900
'20
'40
'60
'80
2000
'20
'40
'60
'80
2100
Year

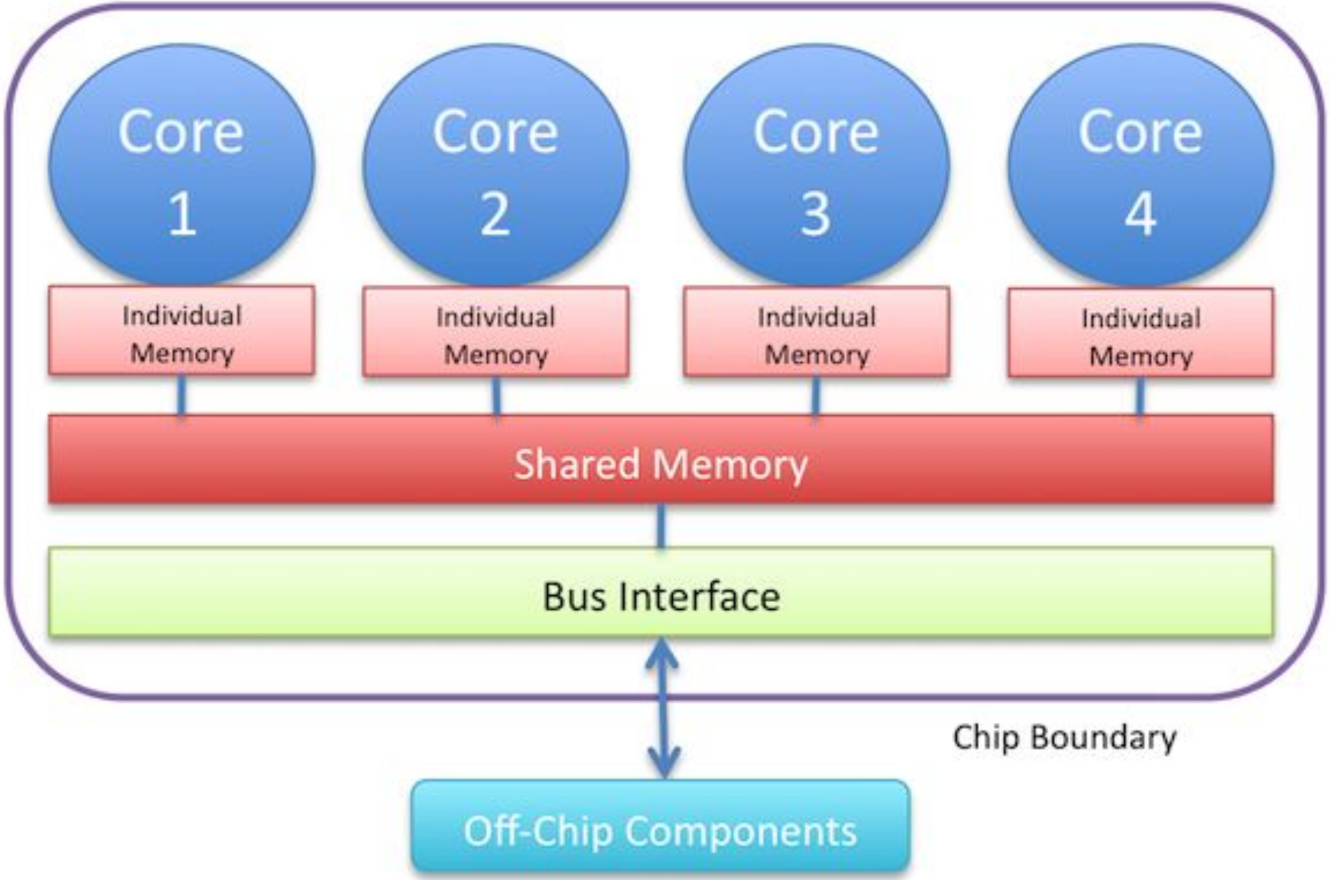
Scale Up - Performance of a Single Node



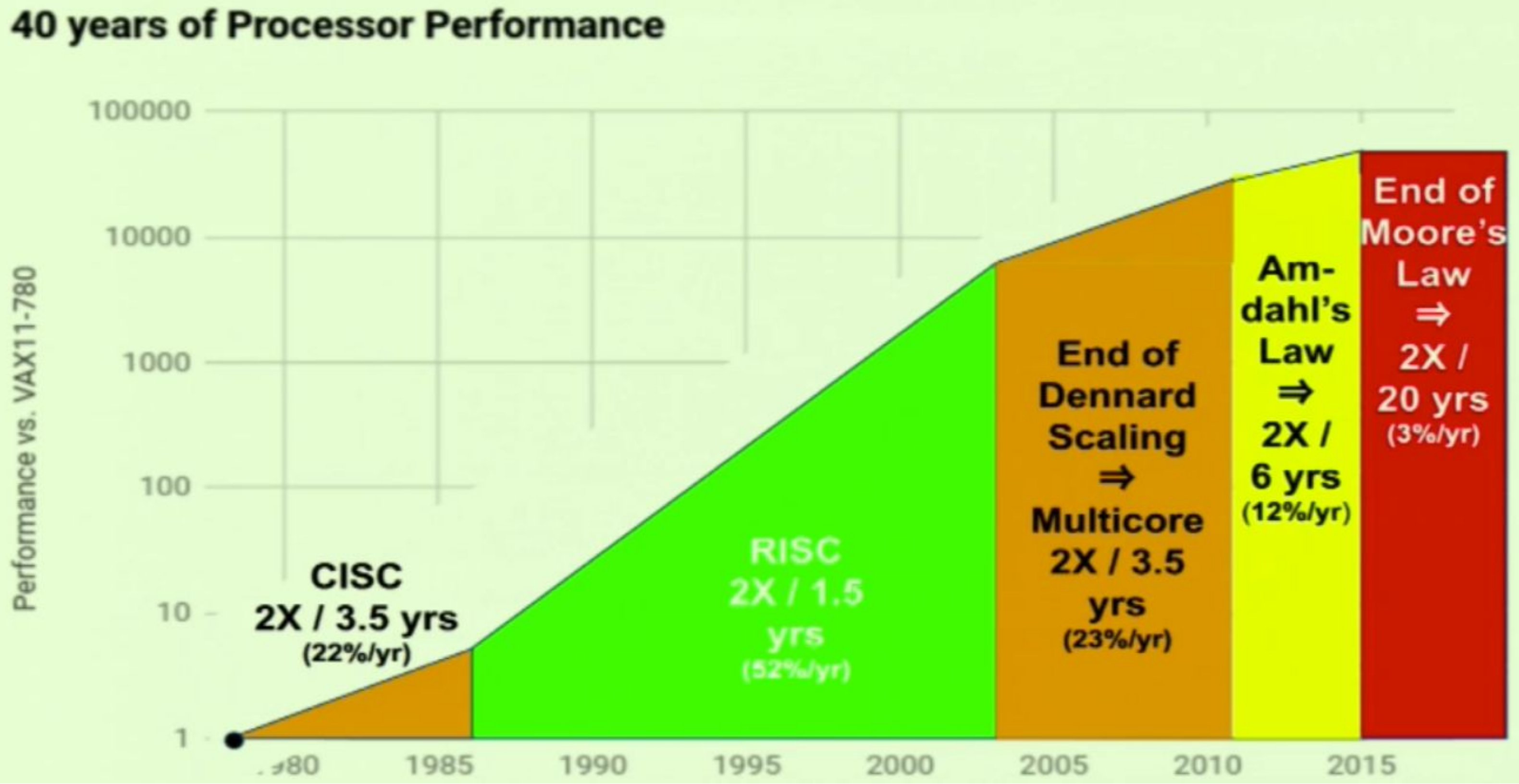
Scale Up - Performance of a Single Node



Multi-core Processor



End of Growth of Performance?



https://software.intel.com/en-us/blogs/2014/02/19/why-has-cpu-frequency-ceased-to-grow

Amdahl's Law: Limits of Parallelization

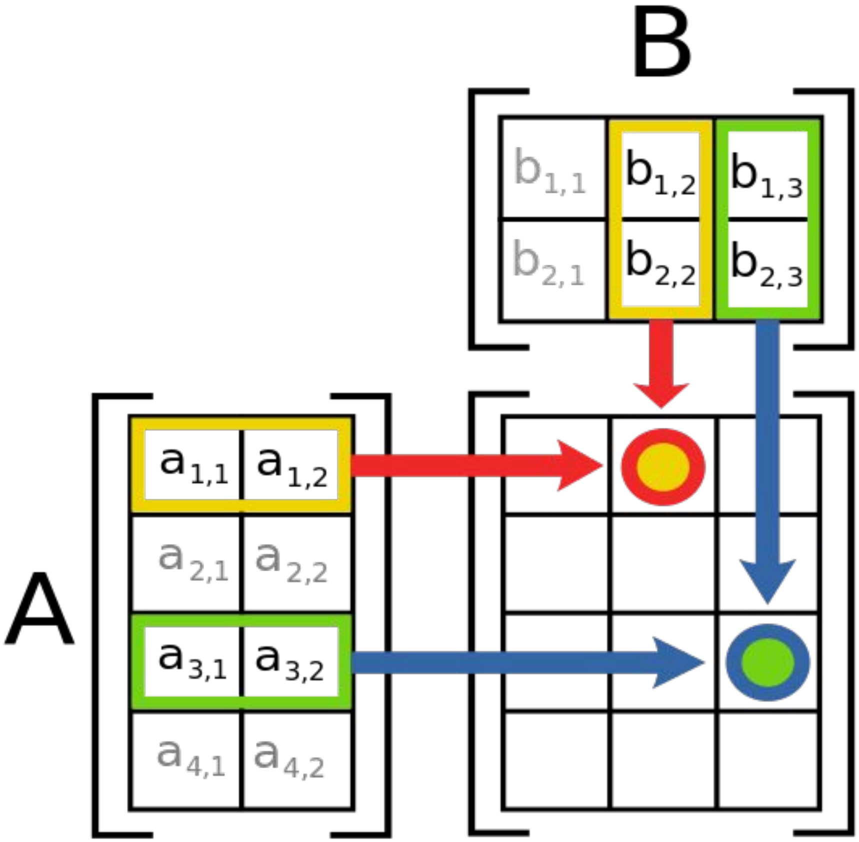
$$S_{\text{latency}}(s) = \frac{1}{(1-p) + \frac{p}{s}}$$

Example: 30% of execution, *p*=0.3, executing on two cores *s*=2

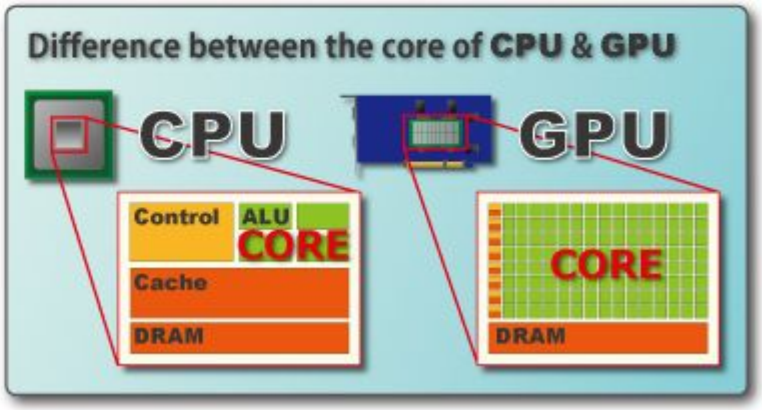
$$= \frac{1}{1 - 0.3 + \frac{0.3}{2}} = 1.18.$$

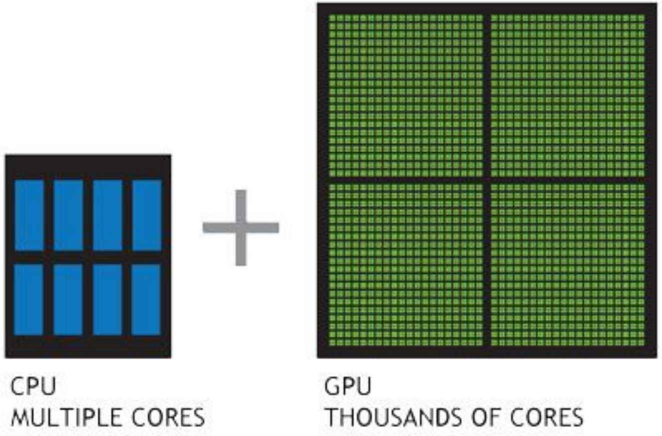
https://en.wikipedia.org/wiki/Amdahl%27s_law

Parallelizable Floating Point Operations



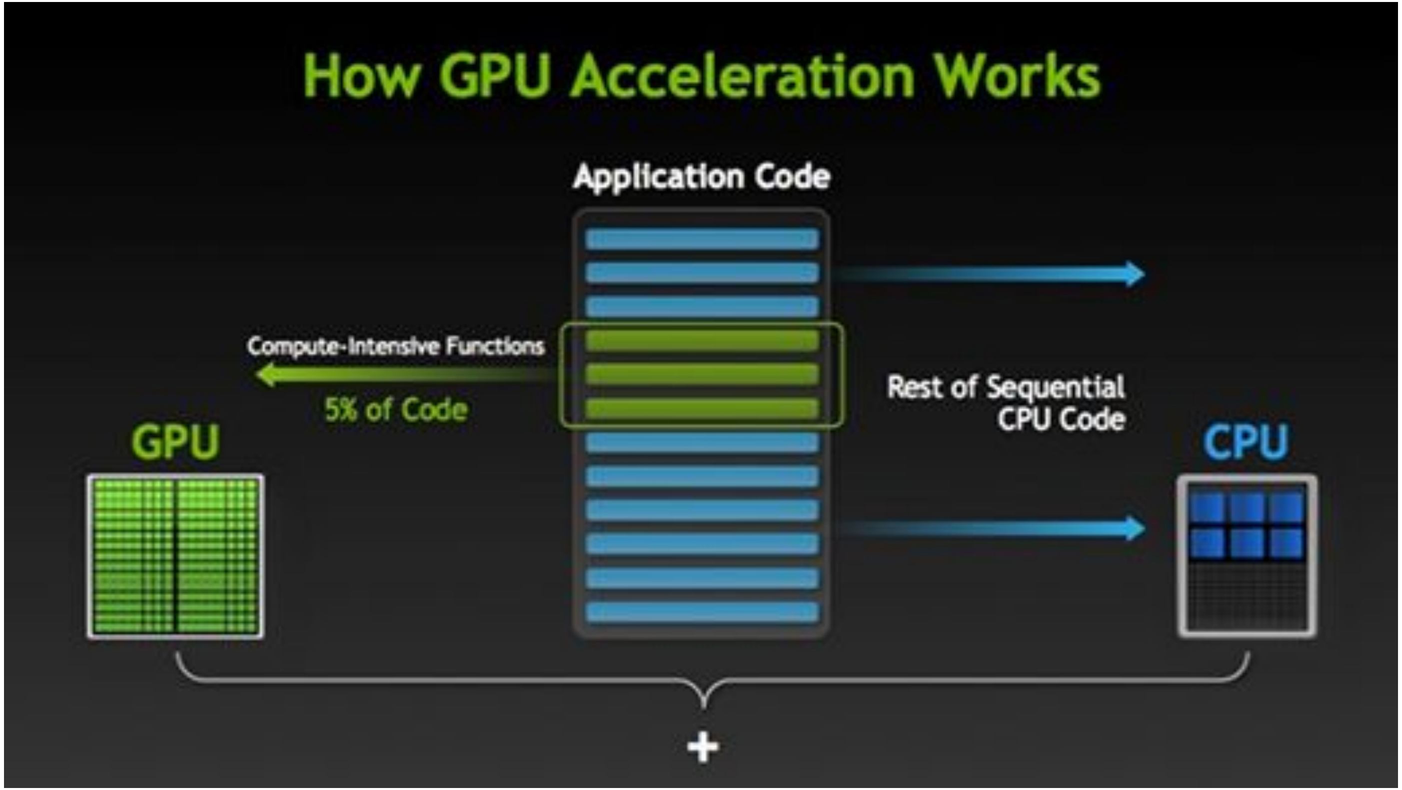
Domain Specific Architectures: Enter the VPU, er GPU

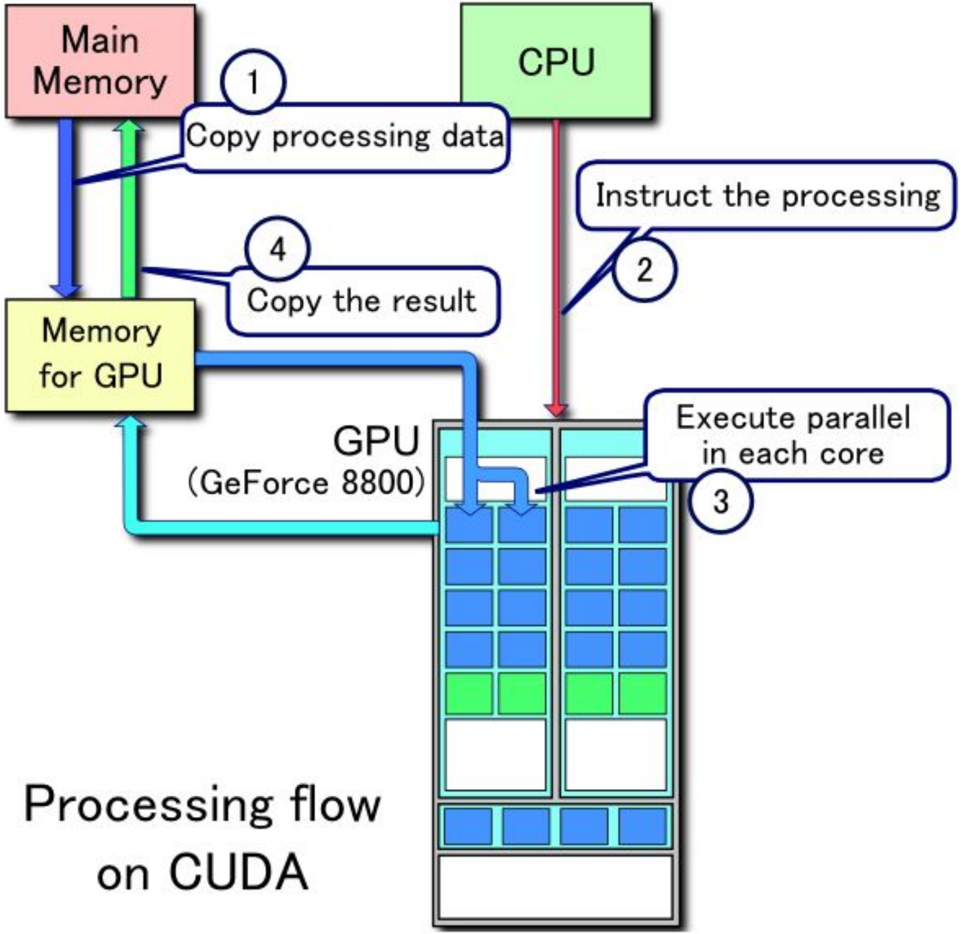





VALENTINE'S GIFT
REALITY
EXPECTATION
QUADRO

Requires new programming model:





Processing flow on CUDA

Abstracted by TensorFlow

Even More Specialized - The TPU

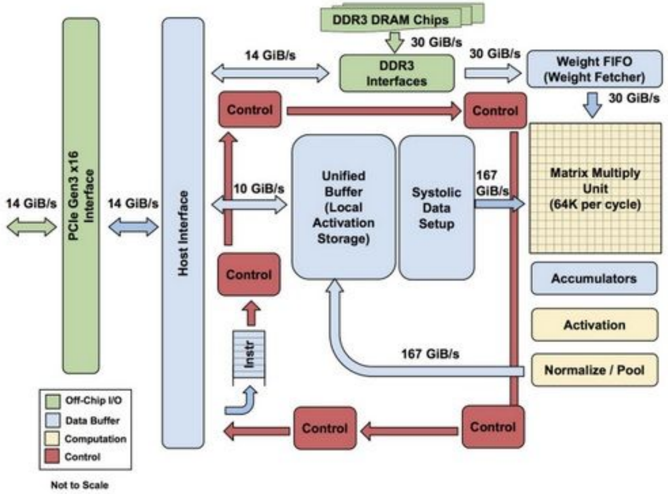
"A Deep Neural Network Accelerator for the Datacenter"

Reduce **inference** phase by 10X vs. GPUs

Run-time vs. Train-time

Systolic Execution - Matrices, not 1D vectors

Low precision: 8 bit FP, good enough for ML



Saved building multiple datacenters!

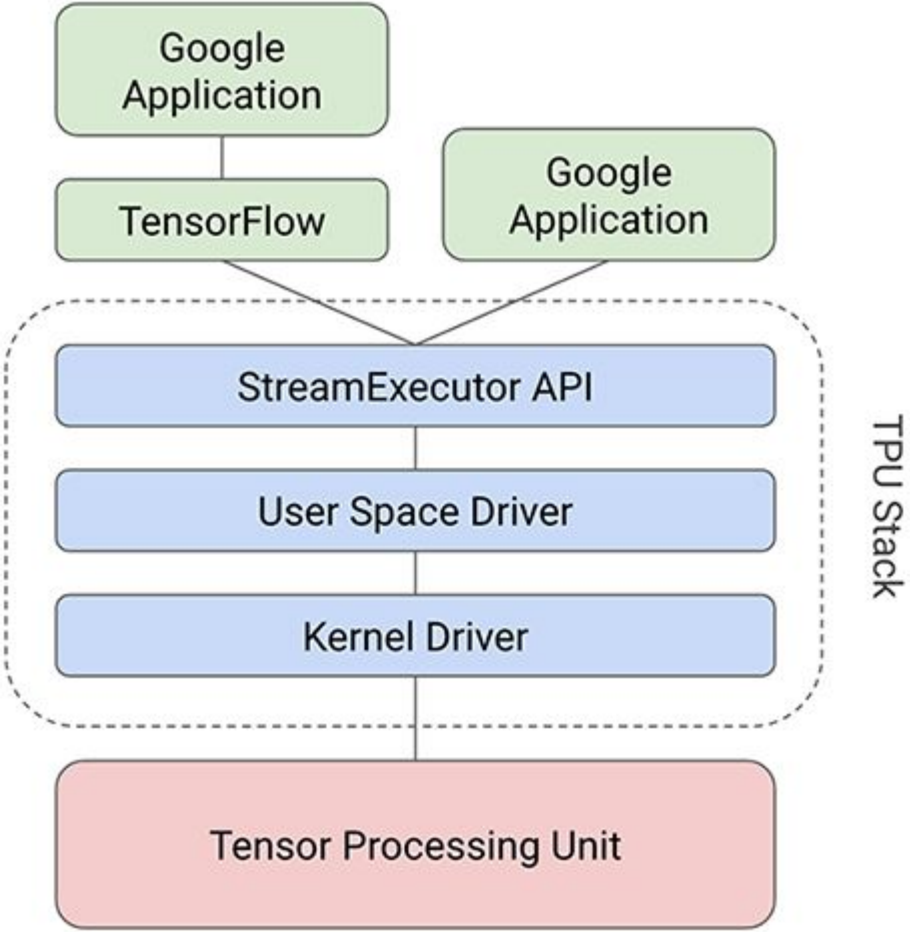
Cliff Young, BA and PhD from Harvard
https://video.seas.harvard.edu/media/CS+Colloquium+Cliff+Young+2017-04-20/1_soxmc716

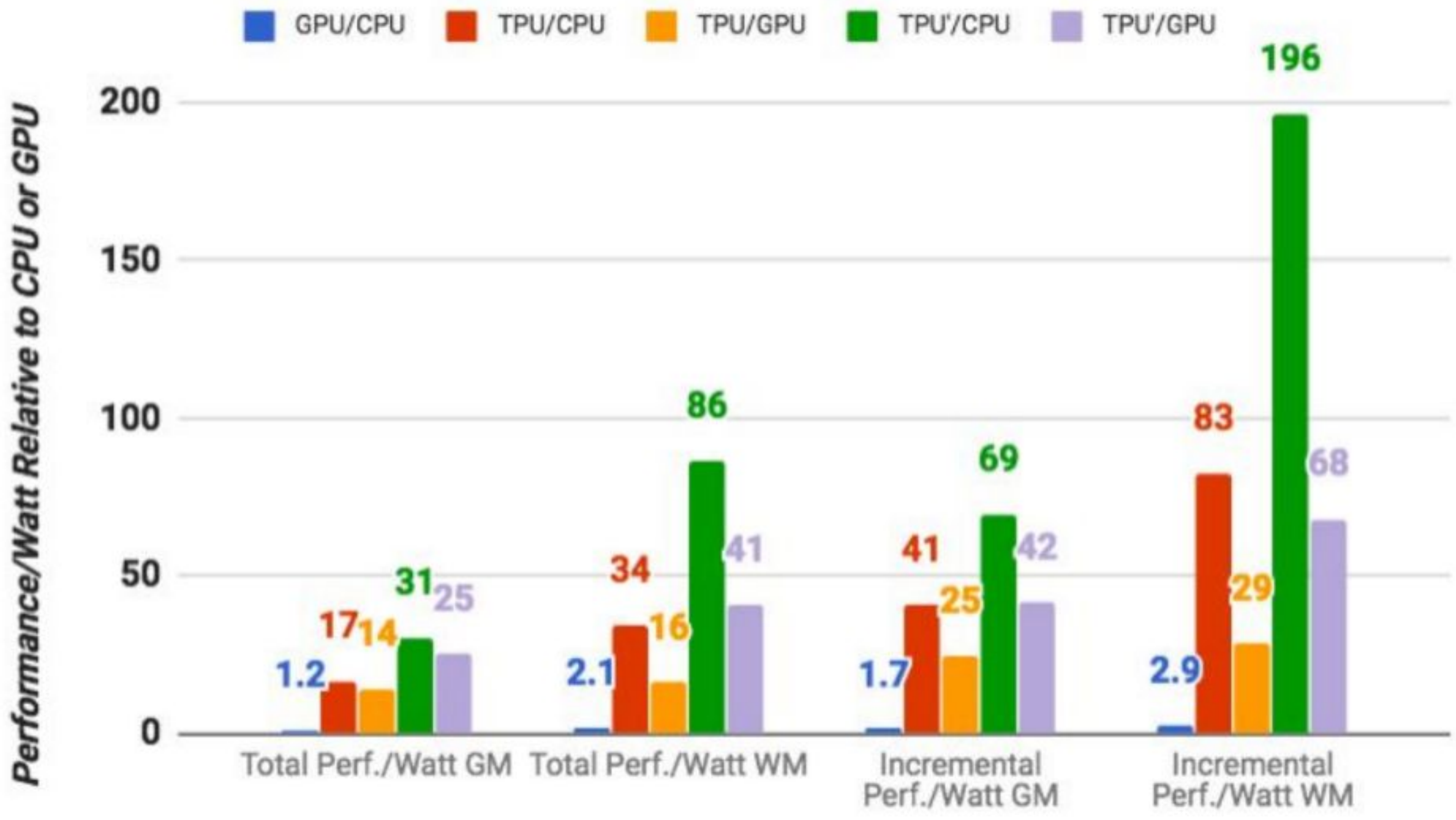
TPU Limitations

Google Specific Chip

Available to non-Googlers via Google Compute Cloud

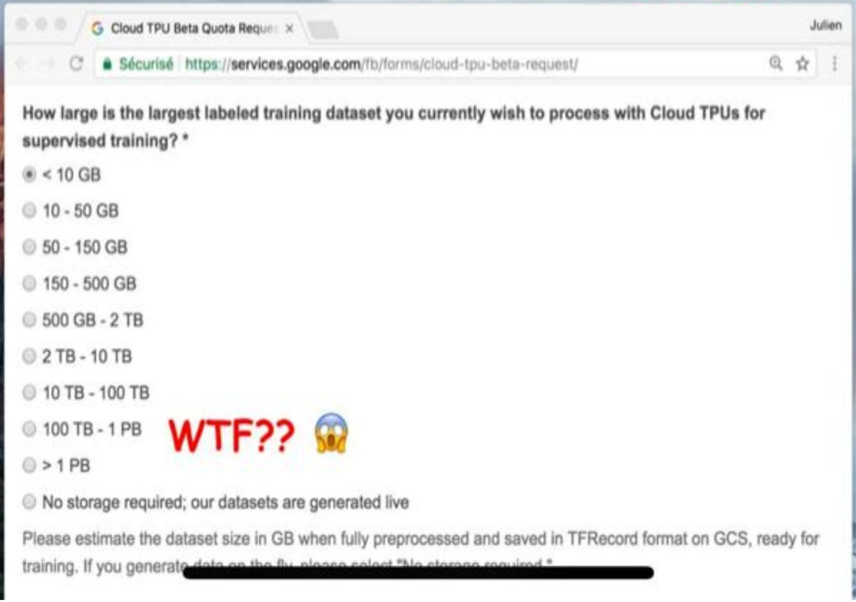
Expect to other versions coming from NVidia, Intel, etc.





GPU/CPU
TPU/CPU
TPU/GPU
TPU'/CPU
TPU'/GPU
Performance/Watt Relative to CPU or GPU
200
150
100
50
0
1.2
17
14
31
25
2.1
34
16
86
41
1.7
41
25
69
42
2.9
83
29
196
68
Total Perf./Watt GM
Total Perf./Watt WM
Incremental Perf./Watt GM
Incremental Perf./Watt WM

Google's TPU form makes me FOMO


Cloud TPU Beta Quota Reque
Julien
Sécurisé https://services.google.com/fb/forms/cloud-tpu-beta-request/

How large is the largest labeled training dataset you currently wish to process with Cloud TPUs for supervised training? *

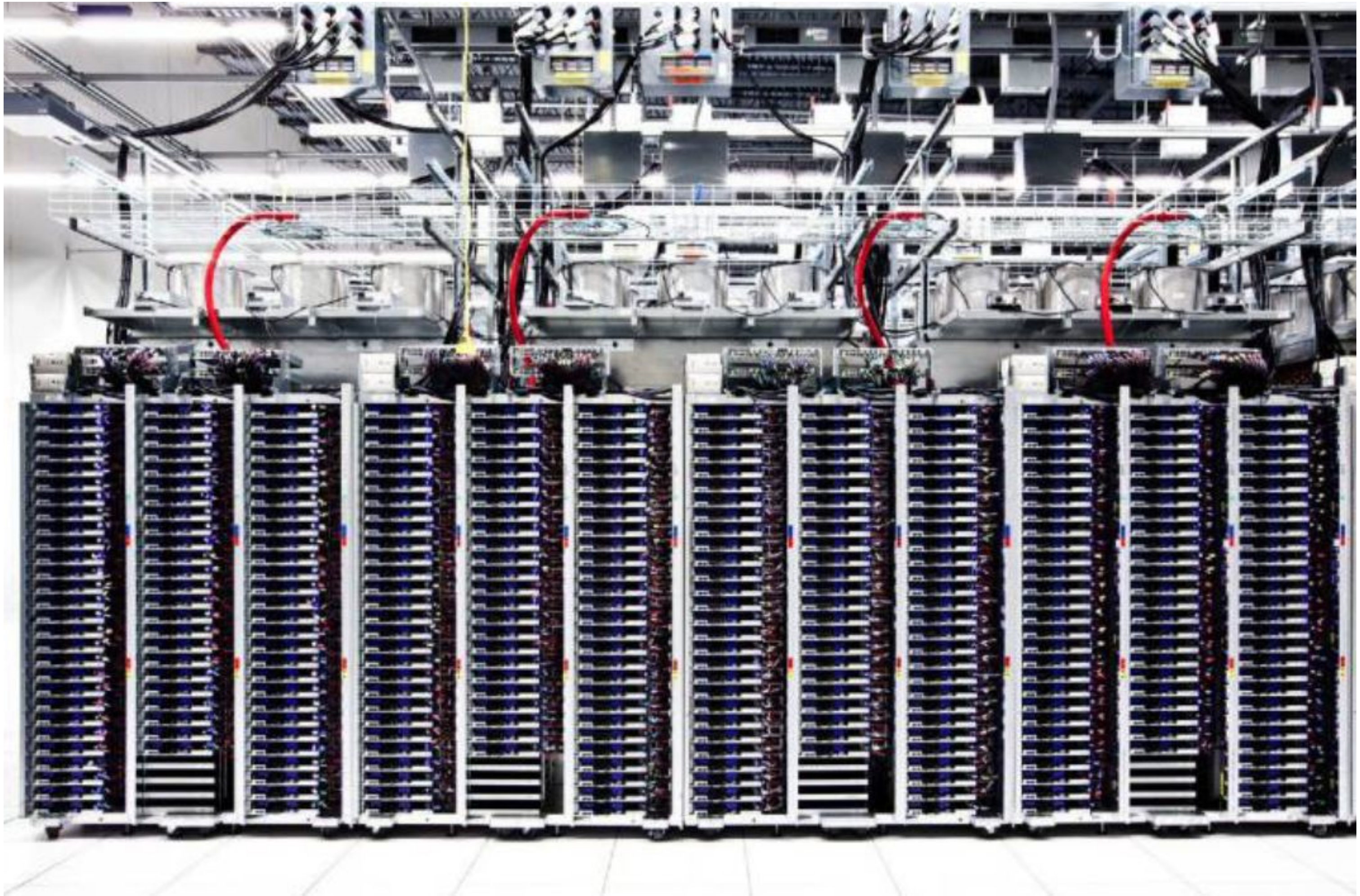
- < 10 GB
- 10 - 50 GB
- 50 - 150 GB
- 150 - 500 GB
- 500 GB - 2 TB
- 2 TB - 10 TB
- 10 TB - 100 TB
- 100 TB - 1 PB
- > 1 PB
- No storage required; our datasets are generated live

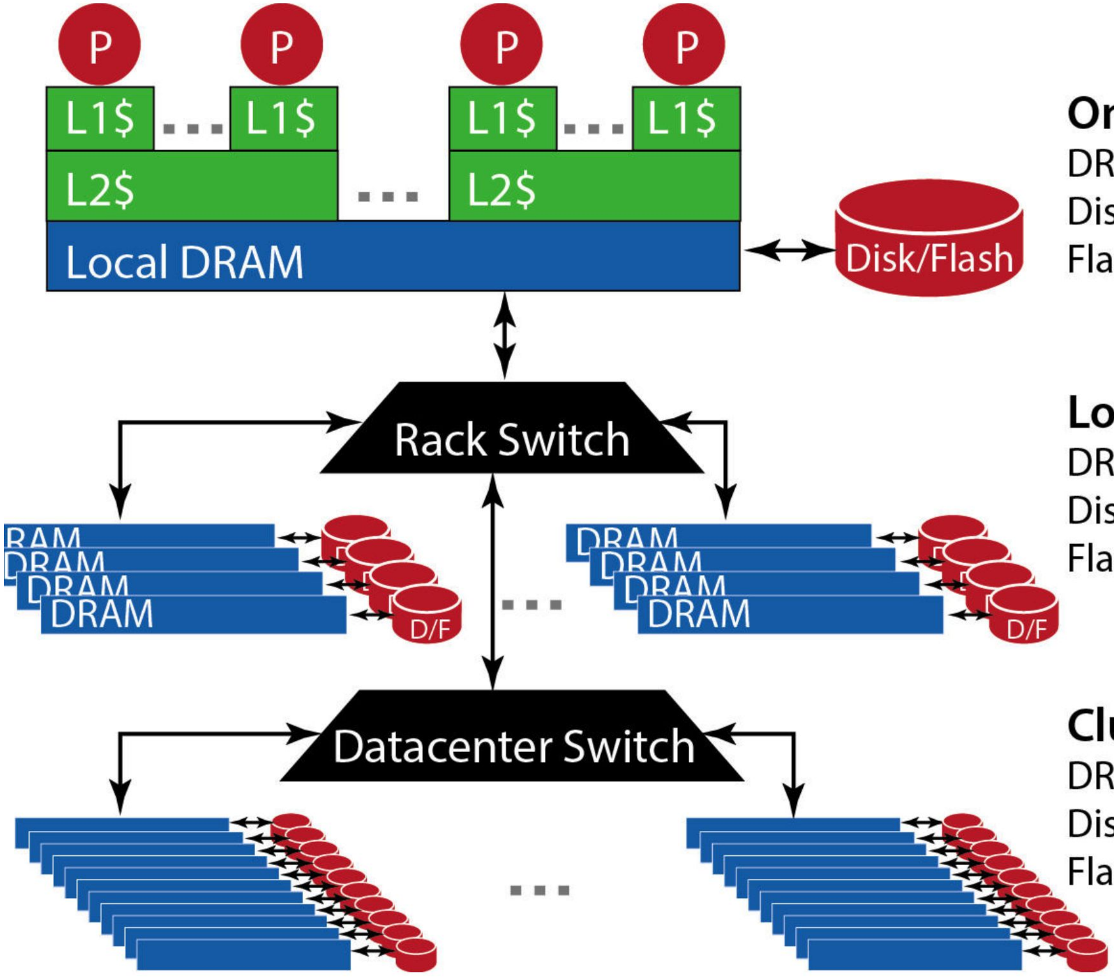
Please estimate the dataset size in GB when fully preprocessed and saved in TFRecord format on GCS, ready for training. If you generate

WTF??

Scale Out

The Datacenter as a Computer





One Server

DRAM: 16 GB, 100 ns, 20 GB/s

Disk: 2T B, 10 ms, 200 MB/s

Flash: 128 GB, 100 us, 1 GB/s

Local Rack (80 servers)

DRAM: 1 TB, 300 us, 100 MB/s

Disk: 160 TB, 11 ms, 100 MB/s

Flash: 20 TB, 400 us, 100 MB/s

Cluster (30 racks)

DRAM: 30 TB, 500 us, 10 MB/s

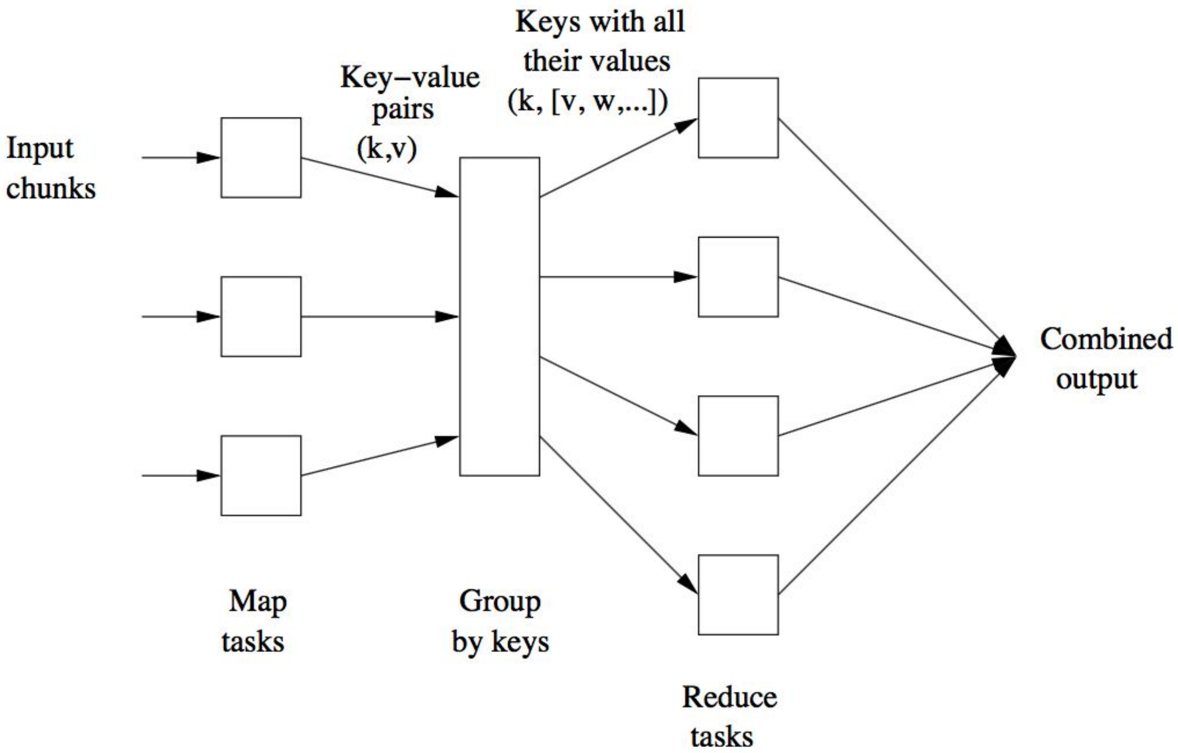
Disk: 4.80 PB, 12 ms, 10 MB/s

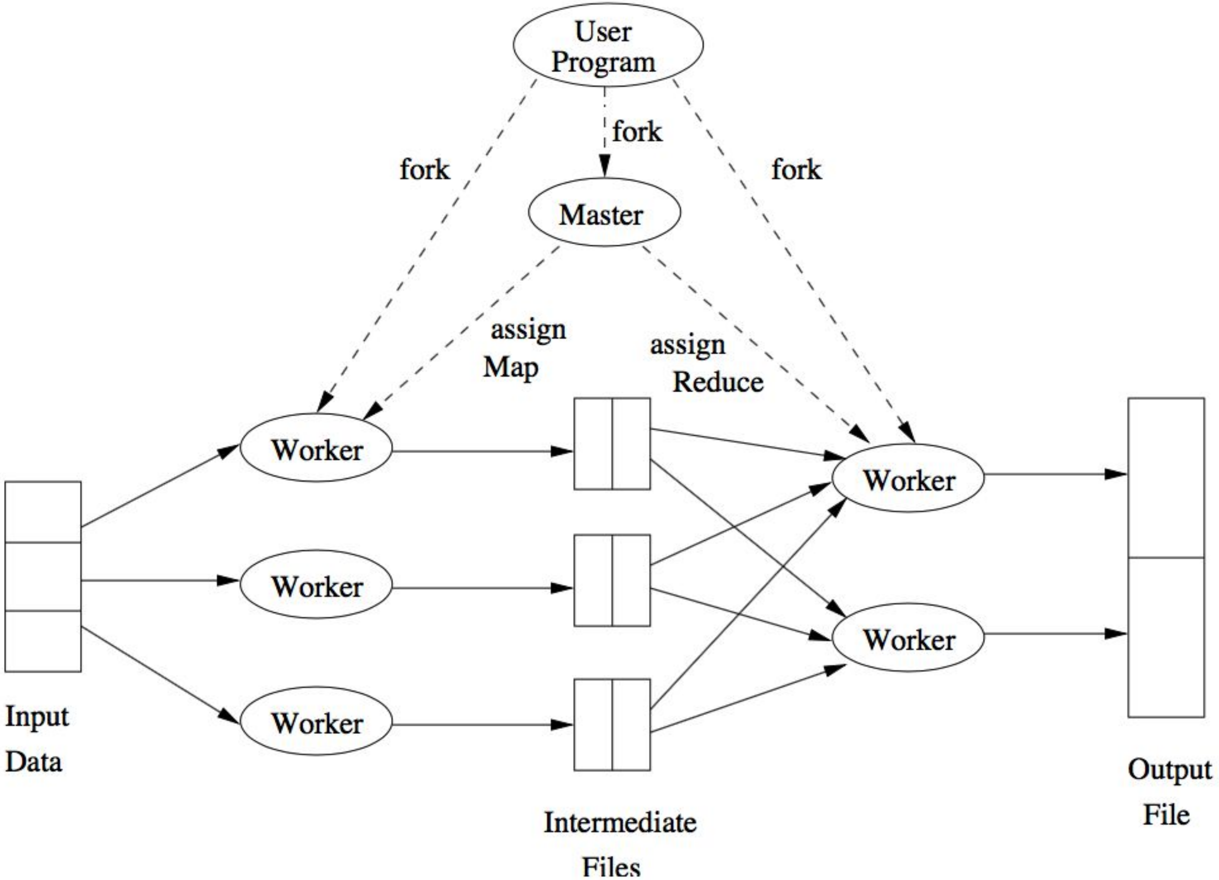
Flash: 600 TB, 600 us, 10 MB/s

Datacenter Issues

- Power: internal and external
- Cooling
- Networking, between racks, between datacenters
- Storage fabric
- Security: physical and data
- Server and Failure management

How to Scale Across Many Nodes: Divide & Conquer




User
Program
fork
fork
fork
Master
assign
Map
assign
Reduce
Worker
Worker
Worker
Worker
Worker
Input
Data
Intermediate
Files
Output
File

Scalable Solutions

Matrix operations, including Machine Learning

Finding Similar Items

Mining data

Link analysis

Clustering Algorithms

Recommendation Systems

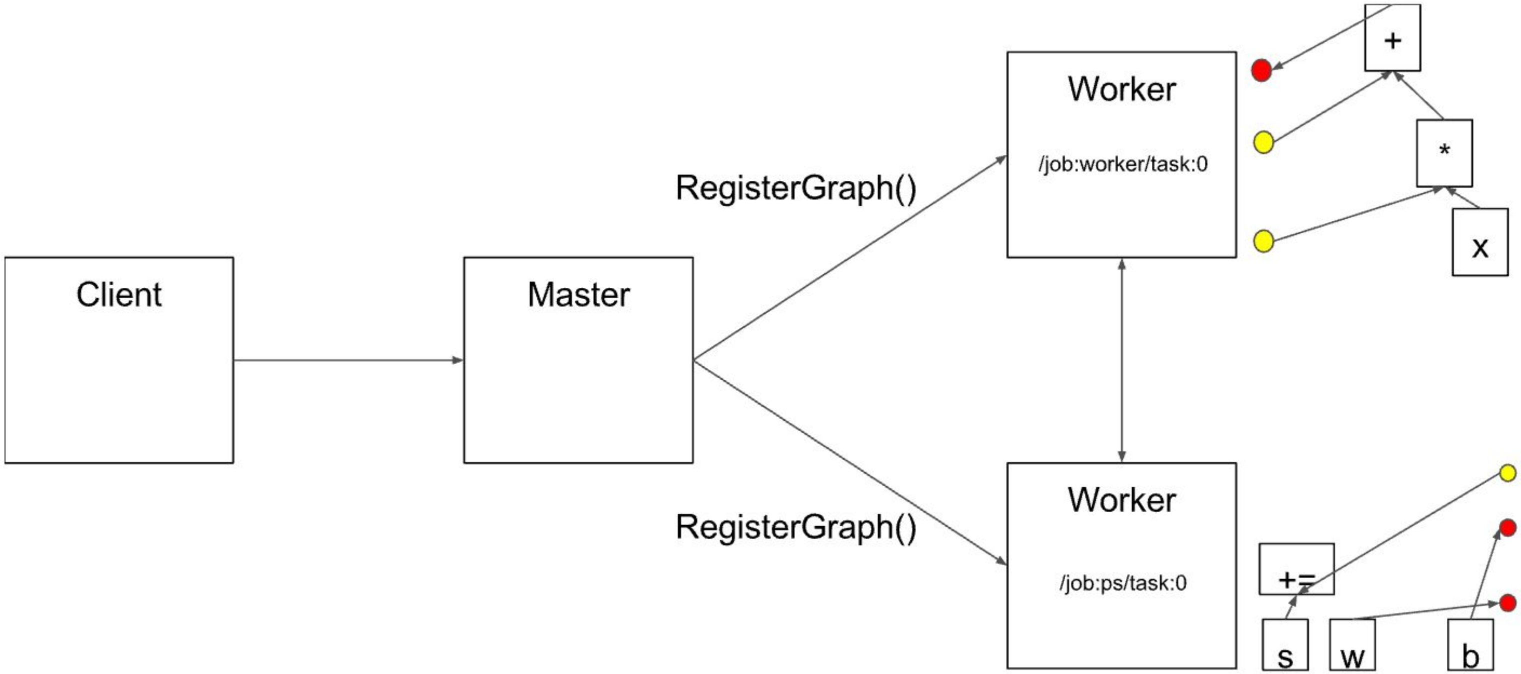
Dimensionality Reduction

Issues:

$$S_{\text{latency}}(s) = \frac{1}{(1-p) + \frac{p}{s}}$$

- Amdahl’s law still applies
- Copy overhead
  - Networking overhead
- Node failures
- Combine / Reconciliation step
- Synchronization
  - shared resources
  - wait time

Machine Learning at Datacenter Scale



https://www.tensorflow.org/extend/architecture See also **PyTorch Distributed**

Future Directions

Abstract scalability behind services:

- Google BigTable / BigQuery, AutoML
  - Specialized APIs: NLP, Images, Translation
- AWS: S3 storage, Comprehend, DynamoDB

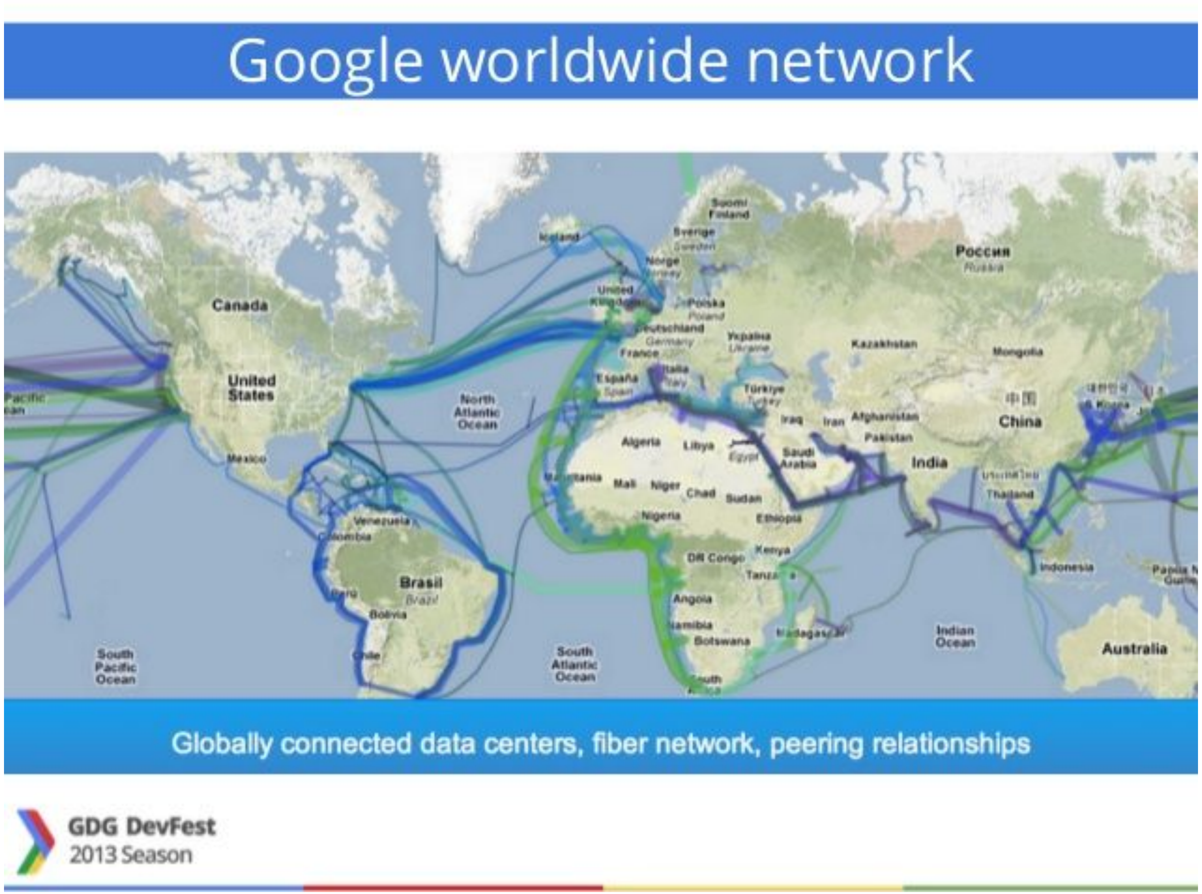
Datacenters have economies of scale and high capital costs

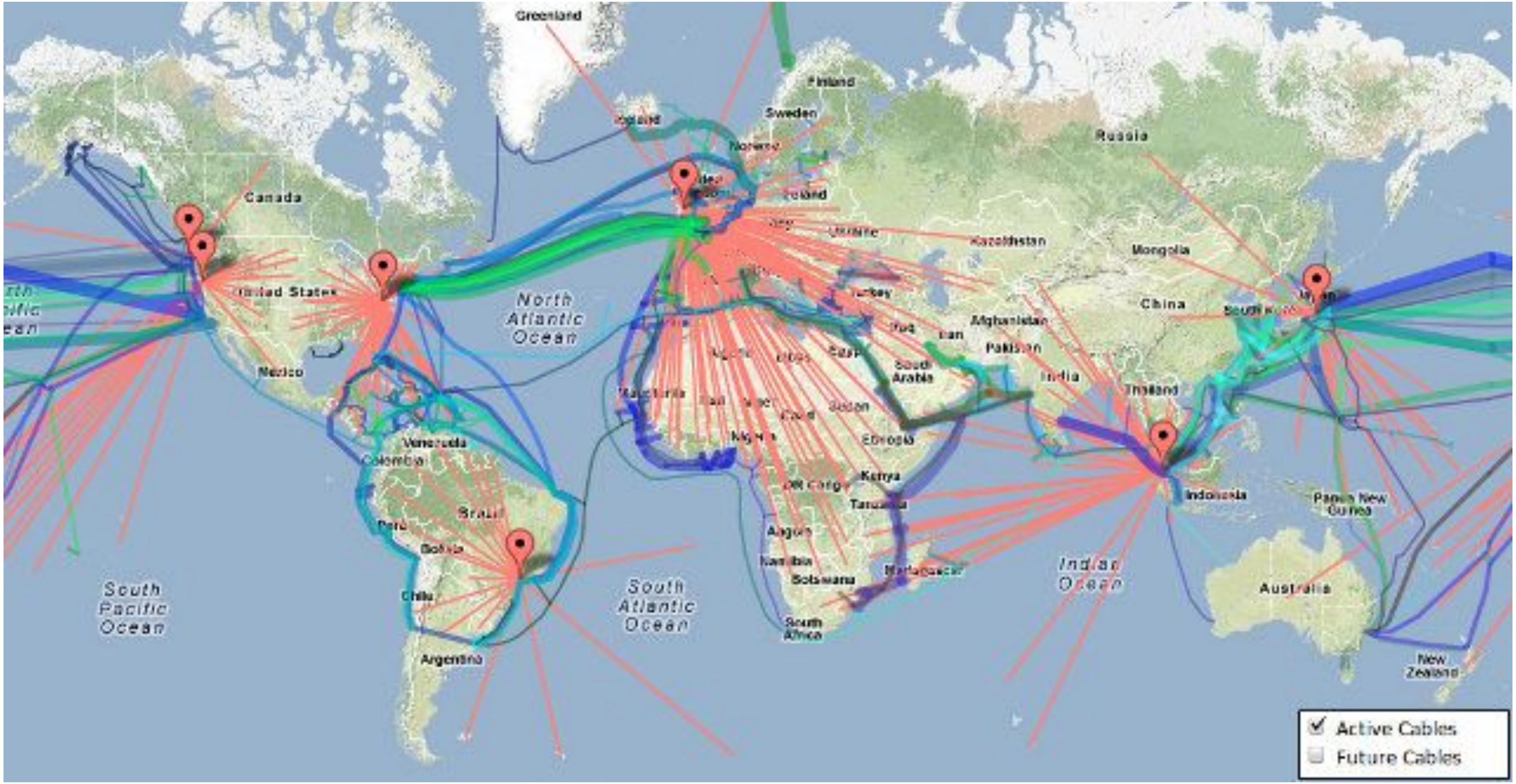
→ Market consolidation (AWS, Google, Microsoft Azure)

Proprietary APIs and services

→ Lock in

Planet Scale





https://stackoverflow.com/questions/3907572/where-are-aws-data-center-locations-listed

References

Mining massive datasets: http://www.mmds.org/

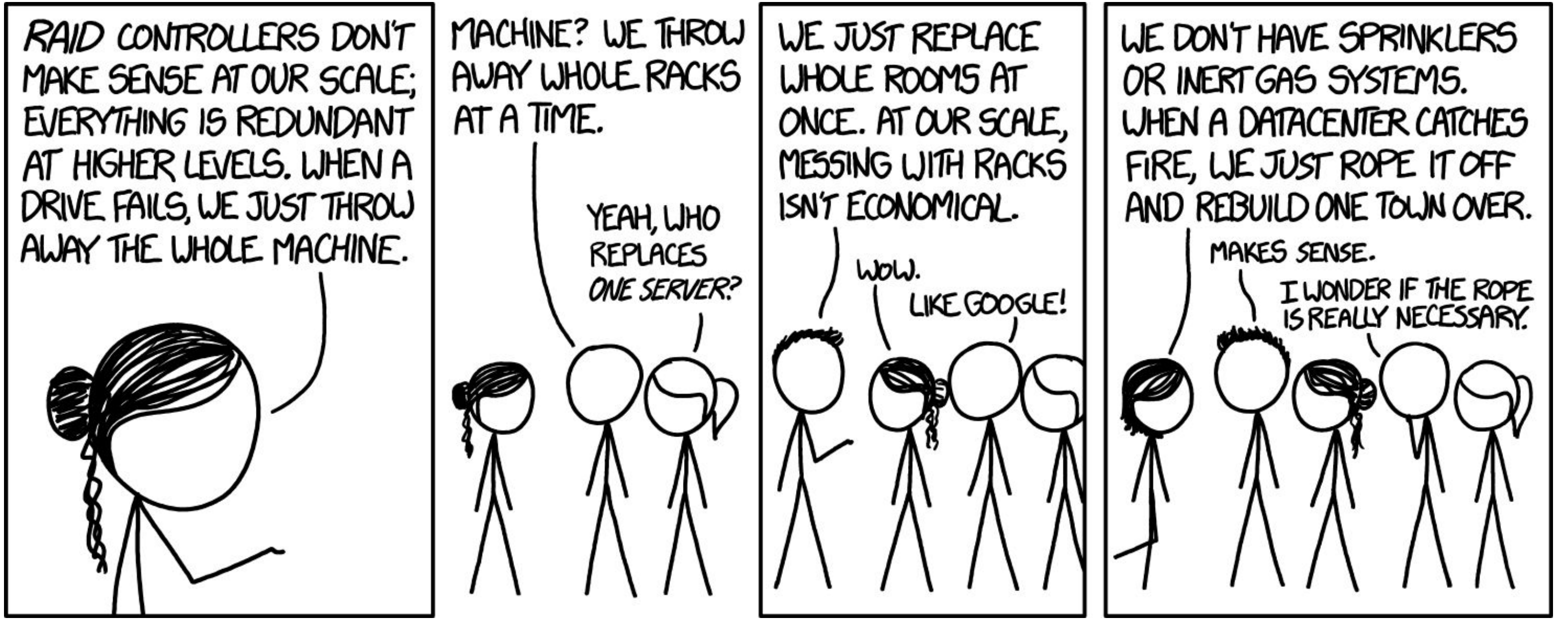
http://highscalability.com/all-time-favorites/

http://www.lecloud.net/tagged/scalability

The Datacenter as a Computer book https://research.google.com/pubs/pub41606.html

TPUs https://cloud.google.com/blog/big-data/2017/05/an-in-depth-look-at-googles-first-tensor-processing-unit-tpu

Quiz Time



https://xkcd.com/1737/