Workshop 2: GPU computing frameworks

Jeremy Jacobson
Lecturer

EMORY COLLEGE OF ARTS AND SCIENCES
Department of Quantitative Theory and Methods

Overview

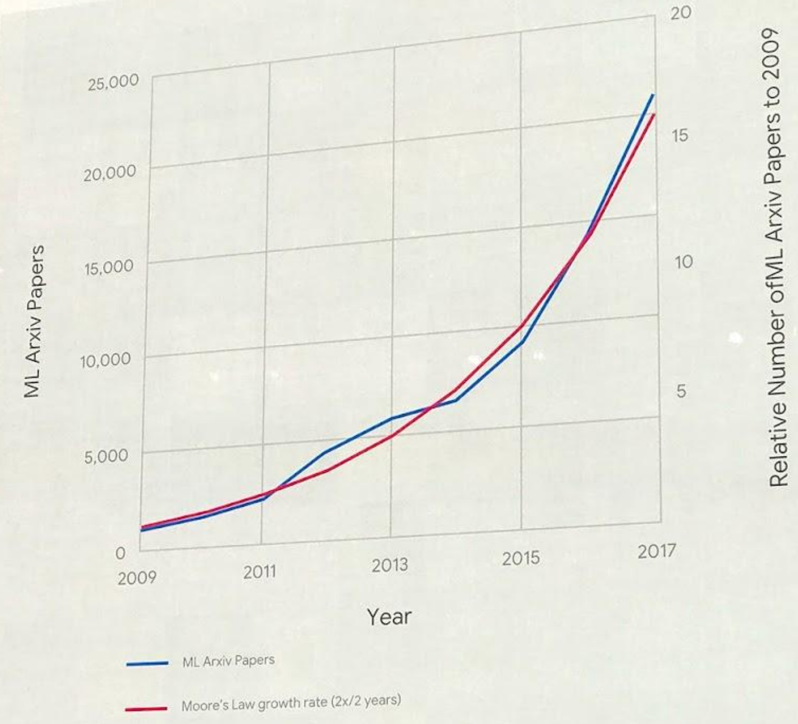
Main points

- (10 minutes) **PhD student Juan Estrada Sosa**
  - Postgres install and use via Docker on the QTM server
- (5 minutes) **Workshop 1 review**
- (30 minutes) **Python vrs Numpy vrs Pytorch**
  - We will revisit the matrix multiply speed improvements suggested in workshop 1 using these frameworks.

Workshop 1 review in three plots

loore's Law
lowdown in
tel Processors
transistor slowing down faster,
fab costs.
0.1.E+07
0.1.E+06
0.1.E+05
0.1.E+04
0.1.E+03
0.1.E+02
0.1.E+01
0.1.E+00
1974
1978
1982
1986
1990
1994
1998
2002
2006
2010
2014
10X
Moore's Law (1975 Version)
Transistor Density

Deep learning is causing a machine learning revolution



From "A New Golden Age in Computer Architecture: Empowering the Machine-Learning Revolution."
Dean, J., Patterson, D., & Young, C. (2018). IEEE Micro, 38(2), 21-29.

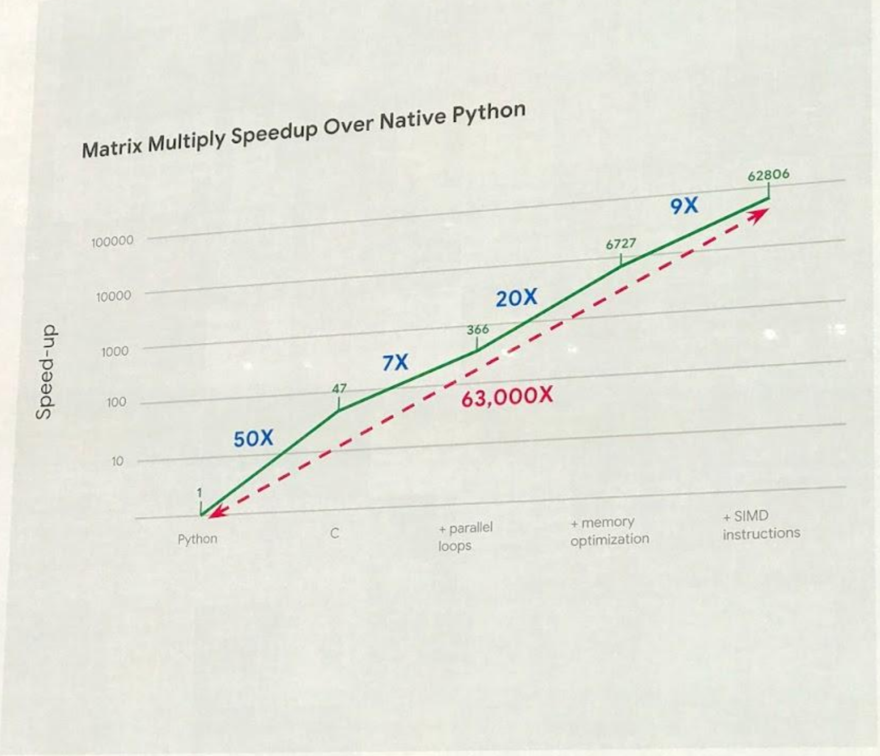
What Opportunities Left?

- SW-centric
  - Modern scripting languages are interpreted, dynamically-typed and encourage reuse
  - Efficient for programmers but not for execution
- HW-centric
  - Only path left is Domain Specific Architectures
  - Just do a few tasks, but extremely well
- Combination
  - Domain Specific Languages & Architectures

A6000 (our GPU) exterior

What's the Opportunity?

Matrix Multiply: relative speedup to a Python version (18 core Intel)



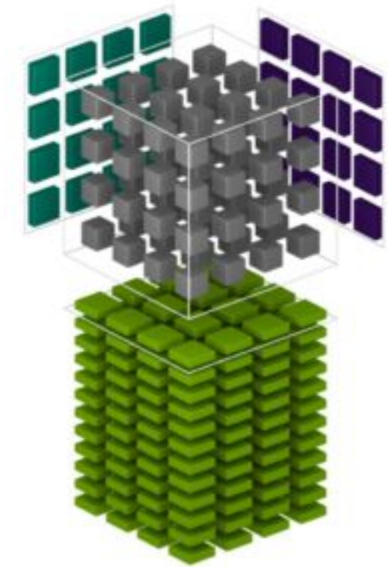
from: "There's Plenty of Room at the Top," Leiserson, et. al., to appear.

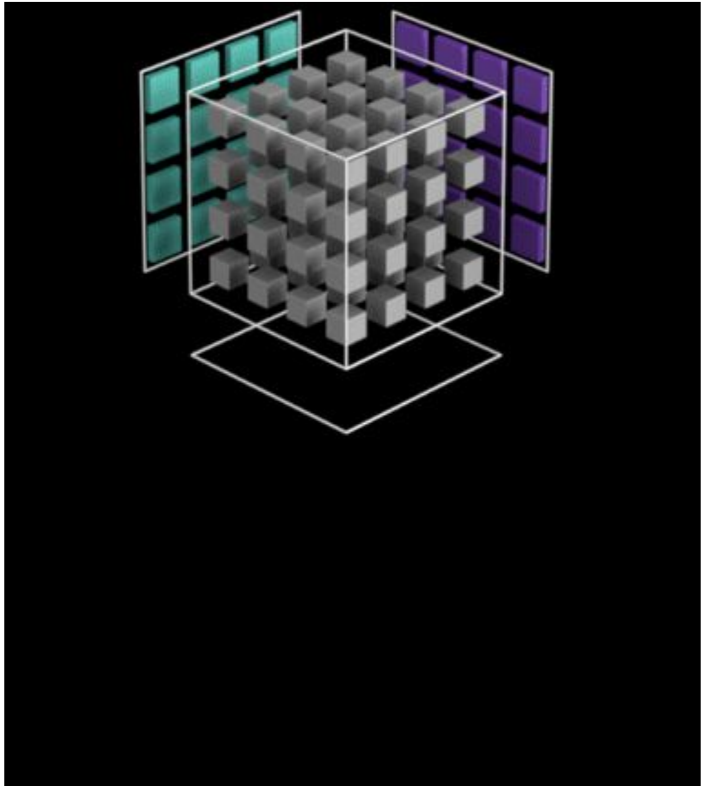
Matrix multiply on a GPU

- Pure Python
- Numpy
- PyTorch

We continue by working through the workshop notebook

Each Tensor Core provides a 4x4x4 matrix processing array which performs the operation **D** = **A * B + C**





Graphics Card	NVIDIA RTX A6000	NVIDIA A40
GPU Codename	GA102	GA102
GPU Architecture	NVIDIA Ampere	NVIDIA Ampere
GPCs	7	7
TPCs	42	42
SMs	84	84
CUDA Cores / SM	128	128
CUDA Cores / GPU	10752	10752
Tensor Cores / SM	4 (3rd Gen)	4 (3rd Gen)
Tensor Cores / GPU	336 (3rd Gen)	336 (3rd Gen)
RT Cores	84 (2nd Gen)	84 (2nd Gen)
GPU Boost Clock (MHz)	1800	1740
Peak FP32 TFLOPS (non-Tensor)[1]	38.7	37.4
Peak FP16 TFLOPS (non-Tensor)[1]	38.7	37.4
Peak BF16 TFLOPS (non-Tensor)[1]	38.7	37.4
Peak INT32 TOPS (non-Tensor)[1,3]	19.4	18.7
Peak FP16 Tensor TFLOPS with FP16 Accumulate[1]	154.8/309.6[2]	149.7/299.4[2]
Peak FP16 Tensor TFLOPS with FP32 Accumulate[1]	154.8/309.6[2]	149.7/299.4[2]
Peak BF16 Tensor TFLOPS with FP32 Accumulate[1]	154.8/309.6[2]	149.7/299.4[2]
Peak TF32 Tensor TFLOPS[1]	77.4/154.8[2]	74.8/149.6[2]

Matrix multiply on a GPU

We continue by working through the workshop notebook

References

- Slides from David Patterson’s talk at Google Cloud Faculty Institute which was not recorded. He gave a talk on similar topics to the ACM and a recording is available here.
- NVIDIA Ampere architecture whitepaper
- CUDA programming guide