GPUs: The Hype, the Reality, and the Future

David Black-Schaffer
Uppsala University
But first...

A CRASH COURSE IN GPUs
What is a GPU?

Graphics Processing Unit

Chips for drawing triangles fast

Nvidia  AMD  Imagination  ARM
Who makes GPUs?

Desktop/Laptop
- Intel
- Nvidia
- AMD

Mobile/Embedded
Who makes GPUs?

**Desktop/Laptop**
- Intel
- Nvidia
- AMD

**Sell devices**

**Mobile/Embedded**
- ARM
- Qualcomm
- Imagination

**Sell IP (mostly)**
Why do we have GPUs?

**Graphics requires performance**

**Video games make money**

(Nvidia’s quarterly GPU revenue is $800M.)
Understanding graphics

How do we compute each pixel?

Run a “shader” program for every pixel
- Every pixel is independent
- Millions of pixels

Perfectly parallel program!

Just add more processors to run faster
Philosophy

**CPU**

How fast *each* program runs

• A few sequential tasks
• Care about latency

• *Hard* to accelerate:
  Need *faster* processors

**GPU**

How fast *all* pixels are drawn

• Millions of parallel tasks
• Care about *throughput*

• *Easy* to accelerate:
  Just need *more* processors
Construction

CPU

Run a few things really fast
Latency Optimized

GPU

Run many things more slowly
Throughput Optimized
In real life

CPU

1 Pipeline
L1
L2
Memory Ordering & Execution
Instruction Decode & Microcode
Instruction Fetch & I Cache
Paging
BP

Latency Optimized
Intel Nehalem EX

GPU

192 Pipelines
LM

Throughput Optimized
Nvidia Kepler
Summary

• **GPUs have very parallel software (graphics)**
  – Millions of independent parallel pixels

• **GPUs are throughput focused**
  – Don’t care about individual pixels, just the total
  – Easy to go faster:
    Add more cores to get more throughput

• **CPUs are latency focused**
  – Don’t care about all programs, just the individual
  – Hard to go faster:
    Need a faster core to get better latency
THE HYPE
How fast are GPUs?

**Performance**

- Intel Xeon: 0 SP GFLOPS
- AMD FirePro W9000: 12x faster
- Nvidia K20: 11x faster

**Power**

- Intel Xeon: 100 Watts
- AMD FirePro W9000: 2.4x power
- Nvidia K20: 2.0x power

12x faster

2.4x power
How fast are GPUs?

- Longer battery life
- Lower TCO
- Faster at fixed power
GPU Hype

12x faster and 5x more efficient.

I want that.

(But I’m not doing graphics...)
What can you expect?

<table>
<thead>
<tr>
<th>Developer</th>
<th>Speed Up</th>
</tr>
</thead>
<tbody>
<tr>
<td>Massachusetts General Hospital</td>
<td>300x</td>
</tr>
<tr>
<td>University of Rochester</td>
<td>160x</td>
</tr>
<tr>
<td>University of Amsterdam</td>
<td>150x</td>
</tr>
<tr>
<td>Harvard University</td>
<td>130x</td>
</tr>
<tr>
<td>University of Pennsylvania</td>
<td>130x</td>
</tr>
<tr>
<td>Nanyang Tech, Singapore</td>
<td>130x</td>
</tr>
<tr>
<td>University of Illinois</td>
<td>125x</td>
</tr>
<tr>
<td>Boise State</td>
<td>100x</td>
</tr>
</tbody>
</table>

![Graph showing normalized core performance]

---

**Debunking the 100X GPU vs. CPU Myth:**

An Evaluation of Throughput Computing on CPU and GPU

Victor W Lee†, Changkyu Daehyun Kim†, Anthony Demmel, Srinivas Devadas, Brian Fail†, Michael Deisher†, Mikhail Smelyanskiy†, John Walpole†, and Pradeep Dubey†

†Throughput Computing Group, Intel Corporation

Intel Corporation

**ABSTRACT**

Recent advances in computing have led to the phenomenon of Data-Intensive Computing (DIC), where the volume of data being generated is so large that it cannot be processed in real-time on CPUs. GPU-accelerated systems are now the de facto standard for DIC.

---

**100x**

**3x**
Real-world performance

• Press release:
  – “NVIDIA today announced that four leading applications... have added support for multiple GPU acceleration, enabling them to cut simulation times from days to hours.”

• GROMACS
  – 2-3x

• LAMPS
  – 2-8x

• QMCPACK
  – 3x

2x is AWESOME!
Real-world performance

• Press release:
  – “NVIDIA today announced that four leading applications... have added support for multiple GPU acceleration, enabling them to cut simulation times from days to hours.”

• GROMACS
  – 2-3x

• LAMPS
  – 2-8x

• QMCPACK
  – 3x

2x is AWESOME! (But is it enough to re-write your code?)
Intel’s response: Xeon Phi

- 60 Pentium cores
- Same price as high-end GPUs
- Legacy code (Linux, OpenMP, MPI)

**Performance**

<table>
<thead>
<tr>
<th></th>
<th>SP GFLOPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intel Xeon</td>
<td>6x</td>
</tr>
<tr>
<td>AMD FirePro W9000</td>
<td>11x</td>
</tr>
<tr>
<td>Nvidia K20</td>
<td>12x</td>
</tr>
<tr>
<td>Intel Phi</td>
<td></td>
</tr>
</tbody>
</table>

**Power**

<table>
<thead>
<tr>
<th></th>
<th>Watts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intel Xeon</td>
<td></td>
</tr>
<tr>
<td>AMD FirePro W9000</td>
<td>2.4x</td>
</tr>
<tr>
<td>Nvidia K20</td>
<td>2.0x</td>
</tr>
<tr>
<td>Intel Phi</td>
<td>2.0x</td>
</tr>
</tbody>
</table>
Intel’s response: Xeon Phi

- 60 Pentium cores
- Same price as high-end GPUs
- Legacy code (Linux, OpenMP, MPI)

“Good enough” legacy compatibility
“Good enough” performance/efficiency
THE REALITY
12x faster and 5x more efficient.

Why do my programs only go 2-5x faster?
12x faster and 5x more efficient.

Why do my (non-graphics) programs only go 2-5x faster?
#1: Slow connection

PCle is slow

Moving data between CPUs and GPUs is slow

CPU/GPU communications is the major bottleneck in desktops/laptops/servers.

Embedded GPUs typically share DRAM directly and avoid this problem.
Embedded share DRAM

CPU and GPU on the same chip

Apple A6
2 CPU cores + 3 GPU cores

CPU/GPU can communicate quickly, but much lower GPU bandwidth.
#2: Not everything is parallel

Speedup limited by how parallel your software is:

- **CPU** (1 core): 90% parallelizable, 10% not parallelizable, 10s
- **CPU + GPU** (1001 cores): 10% parallelizable, 1s, or 10x faster with 1000x as many cores

Hard to find 90% or 99% of a real program that you can parallelize.
Tools are immature or low-level

Programming models:

<table>
<thead>
<tr>
<th></th>
<th>CUDA</th>
<th>OpenCL</th>
<th>OpenACC</th>
</tr>
</thead>
<tbody>
<tr>
<td>mature</td>
<td>low-level</td>
<td>only Nvidia</td>
<td></td>
</tr>
<tr>
<td>immature</td>
<td>low-level</td>
<td>Nvidia, AMD, Intel, ARM, Altera</td>
<td></td>
</tr>
<tr>
<td>immature</td>
<td></td>
<td>Nvidia, Cray, (AMD)</td>
<td></td>
</tr>
</tbody>
</table>

“...writing either OpenCL or CUDA falls into the realm of ‘heroic programming’.”
—Mark Bull, EPCC

—<br>AMD Hessian Kernel. 4 CPU cores + 6 GPU compute units.
What about HPC?

(Top 500 super computers in the world, by % of FLOPS)

The cost of Energy
#1: Tianhe-2
24MW (6.4 for cooling)
1 hour = $24M
1 min = $400k
1 sec = $7000

You make money by selling chips, not FLOPs

CPU Intel, 69%
Nvidia, 5%
AMD GPU, 0%
Intel Phi, 9%
CPU Other, 17%

Clearspeed
Cell
AMDA GPU
Nvidia GPU
Intel Phi
CPU (e.g., Intel)
Are GPUs right for you?

Answer: Yes
  – They’re incredibly fast and awesome

Answer: Maybe
  – They’re fast, but hard to program

Answer: No
  – I need more memory/synchronization/legacy

Key Question: How much performance do you need to justify re-writing your code?
GPUs: everywhere

- Mobile
- Laptop/Desktop
- Consumer

- Datacenter
- Supercomputer
- Base Station
GPUs: everywhere we have a screen

✓ GPU

Mobile

Laptop/Desktop

Consumer

GPU

But GPUs are more efficient. Why won’t they use them?

Datacenter

Supercomputer

Base Station
The bigger trend: Heterogeneity

GPUs win because they are specialized for the task

Heterogeneous processing:
- Different hardware for different tasks
- Better efficiency and performance

Not CPU vs. GPU:
- “big” processor cores for latency:
- “small” processor cores for throughput:
Heterogeneity *everywhere*

- **Mobile**: CPUs + GPUs
- **Laptop/Desktop**: CPUs + GPUs
- **Consumer**: CPUs + GPUs

**GPUs where we need graphics**

- **Datacenter**: Big CPU + little CPU
- **Supercomputer**: Big CPU + little CPU
- **Base Station**: Big CPU + little CPU

Heterogeneity *everywhere for efficiency*
Heterogeneity today

Nvidia Tegra 3
4 fast cores + 1 slow core

ARM big.LITTLE
Small CPU core + Big CPU core

AMD Fusion
4 CPU cores + 10 GPU cores

Match the software to the hardware for efficiency and performance
Why Heterogeneity?

Power scaling has stopped
We can still make smaller transistors...
...but they don’t use less power

Power-limited everywhere
– Battery in mobile
– Cooling in everything else

The only way to get more performance is to improve efficiency
• Heterogeneity matches the hardware to the software
• Better efficiency $\rightarrow$ better performance

Can’t build more cores to get more performance
Can’t burn more power to go faster

(This is why Intel put 60 older, simpler Pentium cores on Xeon Phi and not 60 Xeon cores.)
Take-away messages

GPUs excel due to **nice software** (graphics is easy to scale)

**End of power scaling:**
Must improve efficiency to increase performance

**Heterogeneity is the key:**
Match the hardware to the software (GPUs are one example)

**Tools are immature:**
Hard to justify re-writing applications

**Future: heterogeneous systems**
“Big” and “little” cores (but GPUs only where we have screens)
QUESTIONS?