S05: High Performance Computing with CUDA

The Democratization of Parallel Computing

David Luebke, NVIDIA Research

Parallel Computing's Golden Age

- 1980s, early `90s: a golden age for parallel computing
  - Particularly *data-parallel computing*
- Machines
  - Connection Machine CM-1/2/5, MasPar, Cray X-MP/Y-MP
  - True supercomputers: exotic, powerful, expensive
- Algorithms, languages, & programming models
  - Solved a wide variety of problems
  - Various parallel algorithmic models developed
  - P-RAM, V-RAM, circuit, hypercube, etc.

Distributed Computing's Golden Age

- But…impact of data-parallel computing limited
  - Thinking Machines sold 7 CM-1s (100s of systems total)
  - MasPar sold ~200 systems
- Commercial and research activity largely subsides
  - Massively-parallel machines replaced by clusters of ever-more powerful commodity microprocessors
  - Beowulf, Legion, grid computing, …
- Enter the era of *distributed computing*

Massively parallel computing loses momentum to inexorable advance of commodity technology

The Democratization of Parallel Computing

- GPU Computing with CUDA brings data-parallel computing to the masses
  - Over 40,000,000 CUDA-capable GPUs by end of 2007!
  - A 500 GFLOPS "developer kit" costs $200
- Data-parallel supercomputers are everywhere!
  - CUDA makes it even more accessible
  - We're already seeing innovations in data-parallel computing

Parallel computing is now a commodity technology!

The "New" Moore's Law

- Computers no longer get faster, just wider
  - Many people (outside this room) have not gotten this memo
- You *must* re-think your algorithms to be aggressively parallel!
  - Not just a good idea – the only way to gain performance
  - Otherwise: if its not fast enough now, it never will be
  - Data-parallel computing offers the most scalable solution

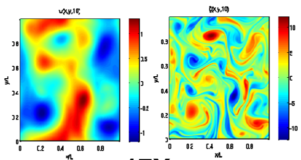
GPU computing with CUDA provides a scalable data-parallel platform in a familiar environment - C

S05: High Performance Computing with CUDA 5

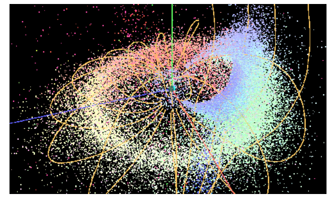
S05: High Performance Computing with CUDA

Motivation

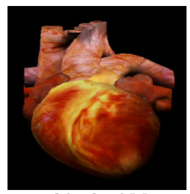
45X



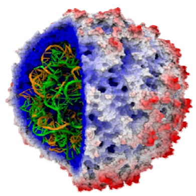
17X



100X

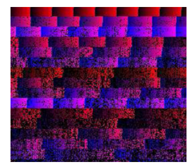


40-170X



110X

Motivation

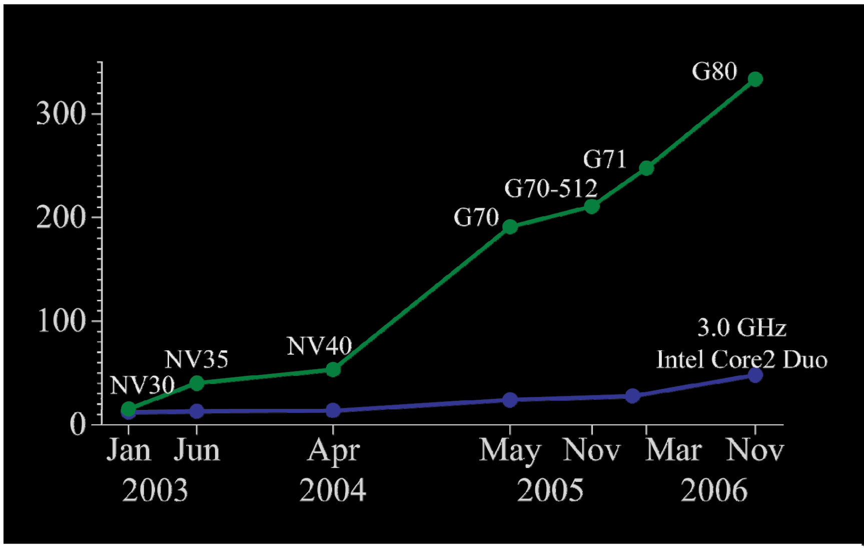


35X

GPUs Are Fast

- Roughly 10x CPU bandwidth and computation
- Impressive microbenchmark performance
  - Raw math: 472 GFLOPS sustained (8800 Ultra)
  - Raw bandwidth: 86 GB per second (Tesla C870)
- More impressive useful application performance
  - Dense n-body computations: 240 GFLOPS
    - 12 billion interactions per second
  - Case studies on molecular dynamics, seismic processing

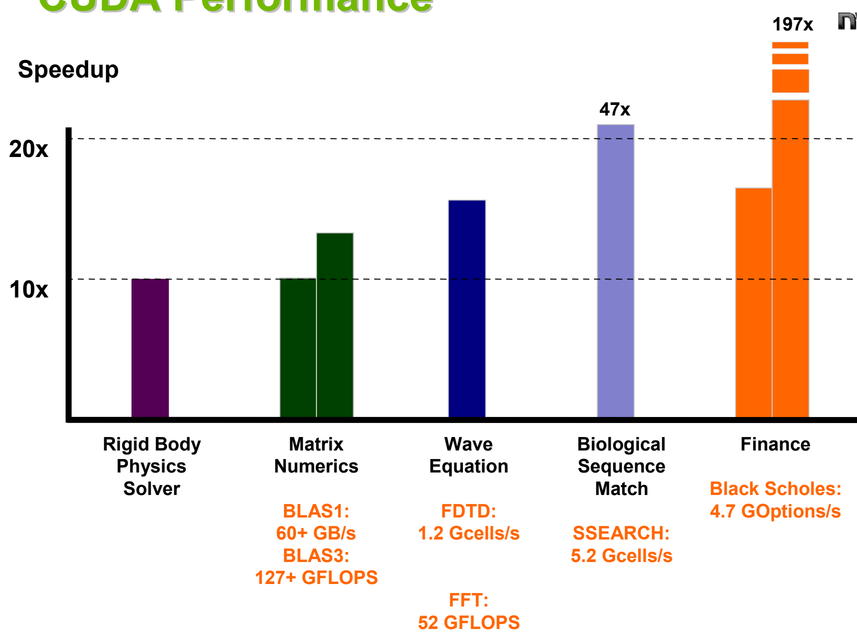
GPUs Are Fast and Getting Faster



S05: High Performance Computing with CUDA

9

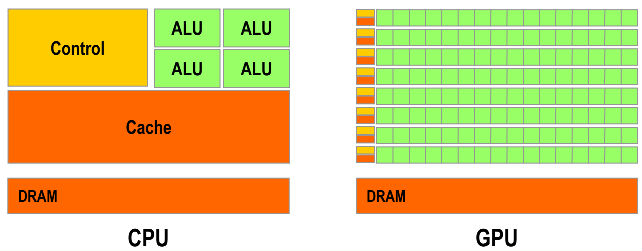
CUDA Performance



10

Why Are GPUs So Fast?

- GPU originally specialized for math-intensive, highly parallel computation
- So, more transistors can be devoted to data processing rather than data caching and flow control



- Commodity industry: provides economies of scale
- Competitive industry: fuels innovation

11

S05: High Performance Computing with CUDA

GPU Computing Overview

Problem: GPGPU

- OLD: *GPGPU* – trick the GPU into general-purpose computing by casting problem as graphics
  - Turn data into images ("texture maps")
  - Turn algorithms into image synthesis ("rendering passes")
- Promising results, but:
  - Tough learning curve, particularly for non-graphics experts
  - Potentially high overhead of graphics API
  - Highly constrained memory layout & access model
  - Need for many passes drives up bandwidth consumption

Solution: GPU Computing

- NEW: *GPU Computing* with CUDA
  - CUDA = *Compute Unified Driver Architecture*
  - Co-designed hardware & software for direct GPU computing
- Hardware: fully general data-parallel architecture
  - General thread launch
  - Global load-store
  - Parallel data cache
  - Scalar architecture
  - Integers, bit operations
  - Double precision (soon)
- Software: program the GPU in C
  - Scalable data-parallel execution/memory model
  - C with minimal yet powerful extensions

CUDA Performance Advantages

- Performance:
  - BLAS1: 60+ GB/sec
  - BLAS3: 127 GFLOPS
  - FFT: 52 benchFFT* GFLOPS
  - FDTD: 1.2 Gcells/sec
  - SSEARCH: 5.2 Gcells/sec
  - Black Scholes: 4.7 GOptions/sec
  - VMD: 290 GFLOPS
- How:
  - Leveraging the parallel data cache
  - GPU memory bandwidth
  - GPU GFLOPS performance
  - Custom hardware intrinsics
    - `__sinf()`, `__cosf()`, `__expf()`, `__logf()`, …

All benchmarks are compiled code!

A New Platform: Tesla

- HPC-oriented product line
  - C870: board (1 GPU)
  - D870: deskside unit (2 GPUs)
  - S870: 1u server unit (4 GPUs)



The Future of GPUs

- GPU Computing drives new applications
  - Reducing “Time to Discovery”
  - 100x Speedup changes science and research methods
- New applications drive the future of GPUs and GPU Computing
  - Drives new GPU capabilities
  - Drives hunger for more performance

Motivation, Summarized

- GPUs are the new massively parallel computers
  - The most successful data-parallel processor in history
  - CUDA enables a rich variety of parallel algorithms
- There is tremendous potential for acceleration
- There is tremendous scope for innovation

S05: High Performance Computing with CUDA

Tutorial Overview

Tutorial Goals

- A detailed introduction to high performance computing with CUDA
- We emphasize:
  - Understanding the architecture & programming model
  - Core computational building blocks
  - Libraries and tools
  - Optimization strategy & tactics
- Case studies to bring it all together

Tutorial Prerequisites

- Tutorial intended to be accessible to any savvy computer or computational scientist
- Helpful but not required: familiarity with data-parallel algorithms and programming
- Target audience
  - Scientists and engineers interested in dramatic speedups
  - HPC researchers interested in GPU computing
  - Attendees wishing a survey of this exciting field

Speakers

- In order of appearance:
  - David Luebke **NVIDIA**
  - Ian Buck **NVIDIA**
  - Massimiliano Fatica **NVIDIA**
  - John Owens **University of California Davis**
  - Mark Harris **NVIDIA**
  - John Stone **University of Illinois Urbana-Champaign**
  - Jim Phillips **University of Illinois Urbana-Champaign**
  - Bernard Deschizeaux **CGGVeritas**

Schedule

8:30	**Introduction** *Motivation, GPU computing & CUDA, tutorial overview*		Luebke
9:00	**Programming CUDA** *Execution & memory model, C extensions, examples*		Buck
10:00	Break		
10:30	**Using CUDA Libraries** *CuBLAS, CuFFT, calling CUDA from MATLAB*		Fatica

23

Schedule

11:15	**Building blocks** *Data parallel algorithms in CUDA*		Owens
12:00	Lunch		
1:30	**Building blocks continued**		Owens
1:45	**Optimizing CUDA** *Hiding latency, efficient memory use, exploiting PDC*		Harris
3:00	Break		

24

Schedule

Case Studies

3:30 Molecular visualization & analysis — **Stone**

Direct Coulomb summation, integrating CUDA & VMD

4:00 Molecular dynamics — **Phillips**

NAMD, optimization walkthroughs, performance

4:30 Seismic processing — **Deschizeaux**

Seismic imaging, data flow, optimization, results

5:00 Wrap!