MIC-GPU: High-Performance Computing for Medical Imaging on Programmable Graphics Hardware (GPUs)

SPIE Medical Imaging

Parallel Programming Primer

Klaus Mueller, Ziyi Zheng, Eric Papenhausen

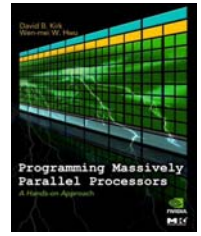
Stony Brook University

Computer Science

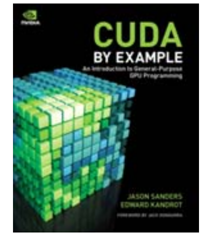
Stony Brook, NY

Recommended Literature

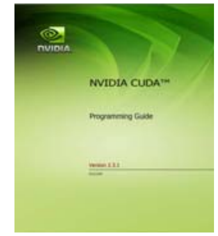
SPIE Medical Imaging



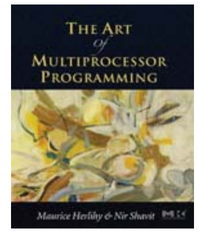
text book

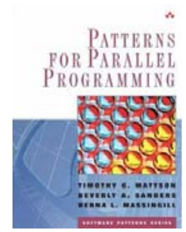


reference book



programming guides available from nvidia.com

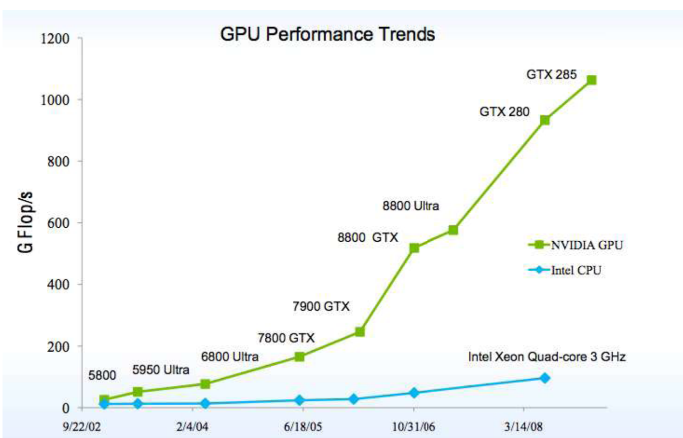




more general books on parallel programing

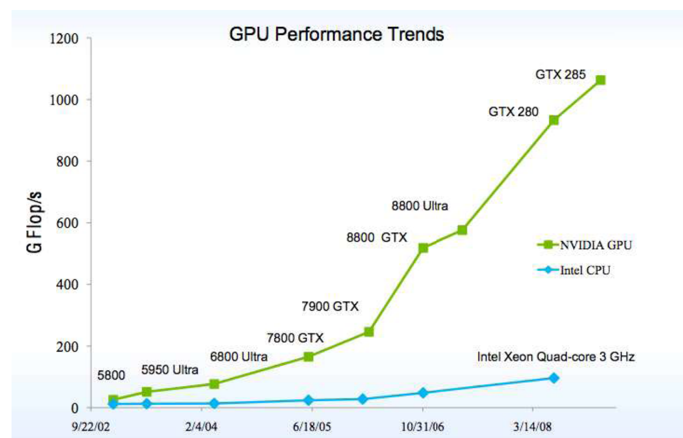
Speedup Curves

SPIE Medical Imaging



Speedup Curves

SPIE Medical Imaging



but wait, there is more to this…..

Amdahl's Law

Governs theoretical speedup

$$S = \frac{1}{(1-P)+\frac{P}{S_{parallel}}} = \frac{1}{(1-P)+\frac{P}{N}}$$

P: parallelizable portion of the program
S: speedup
N: number of parallel processors

Amdahl's Law

Governs theoretical speedup

$$S = \frac{1}{(1-P)+\frac{P}{S_{parallel}}} = \frac{1}{(1-P)+\frac{P}{N}}$$

P: parallelizable portion of the program
S: speedup
N: number of parallel processors

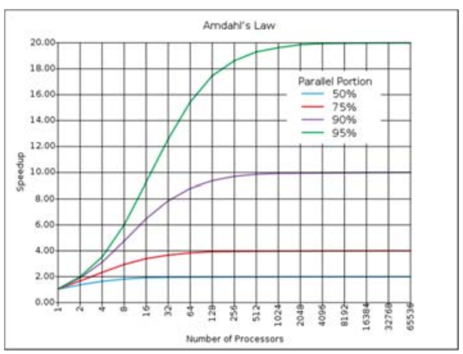
P determines theoretically achievable speedup

- example (assuming infinite N): P=90% → S=10
  P=99% → S=100

Amdahl's Law

How many processors to use

- when P is small → a small number of processors will do
- when P is large (embarrassingly parallel) → high N is useful



Focus Efforts on Most Beneficial

Optimize program portion with most 'bang for the buck'

- look at each program component
- don't be ambitious in the wrong place

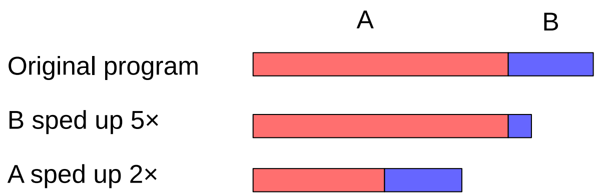
Focus Efforts on Most Beneficial

Optimize program portion with most 'bang for the buck'

- look at each program component
- don't be ambitious in the wrong place

Example:

- program with 2 independent parts: A, B (execution time shown)



- sometimes one gains more with less

Beyond Theory....

Limits from mismatch of parallel program and parallel platform

- man-made 'laws' subject to change with new architectures

Beyond Theory....

Limits from mismatch of parallel program and parallel platform

- man-made 'laws' subject to change with new architectures

Memory access patterns

- data access locality and strides vs. memory banks

Beyond Theory....

Limits from mismatch of parallel program and parallel platform

- man-made 'laws' subject to change with new architectures

Memory access patterns

- data access locality and strides vs. memory banks

Memory access efficiency

- arithmetic intensity vs. cache sizes and hierarchies

Beyond Theory....

Limits from mismatch of parallel program and parallel platform
- man-made 'laws' subject to change with new architectures

Memory access patterns
- data access locality and strides vs. memory banks

Memory access efficiency
- arithmetic intensity vs. cache sizes and hierarchies

Enabled granularity of program parallelism
- MIMD vs. SIMD

Beyond Theory....

Limits from mismatch of parallel program and parallel platform
- man-made 'laws' subject to change with new architectures

Memory access patterns
- data access locality and strides vs. memory banks

Memory access efficiency
- arithmetic intensity vs. cache sizes and hierarchies

Enabled granularity of program parallelism
- MIMD vs. SIMD

Hardware support for specific tasks → on-chip ASICS

Beyond Theory....

Limits from mismatch of parallel program and parallel platform
- man-made 'laws' subject to change with new architectures

Memory access patterns
- data access locality and strides vs. memory banks

Memory access efficiency
- arithmetic intensity vs. cache sizes and hierarchies

Enabled granularity of program parallelism
- MIMD vs. SIMD

Hardware support for specific tasks → on-chip ASICS

Support for hardware access → drivers, APIs

Device Transfer Costs

Transferring the data to the device is also important
- computational benefit of a transfer plays a large role
- transfer costs are (or can be) significant

Device Transfer Costs

Transferring the data to the device is also important

- computational benefit of a transfer plays a large role
- transfer costs are (or can be) significant

Adding two ($N \times N$) matrices:

- transfer back and from device: 3 N^2 elements
- number of additions: N^2

→ operations-transfer ratio = 1/3 or O(1)

Device Transfer Costs

Transferring the data to the device is also important

- computational benefit of a transfer plays a large role
- transfer costs are (or can be) significant

Adding two ($N \times N$) matrices:

- transfer back and from device: 3 N^2 elements
- number of additions: N^2

→ operations-transfer ratio = 1/3 or O(1)

Multiplying two ($N \times N$) matrices:

- transfer back and from device: 3 N^2 elements
- number of multiplications and additions: N^3

→ operations-transfer ratio = O(N) grows with N

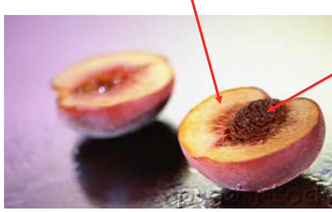
Programming Strategy

Use GPU to complement CPU execution

- recognize parallel program segments and only parallelize these
- leave the sequential (serial) portions on the CPU

parallel portions (enjoy)

sequential portions (do not bite)



PPP (Peach of Parallel Programming – Kirk/Hwu)

Course Schedule

1:30 – 1:45:	Introduction (Klaus)
1:45 – 2:00:	Parallel programming primer (Klaus)
2:00 – 2:15:	GPU hardware (Ziyi)
2:15 – 3:00:	CUDA API, threads (Ziyi)
	Coffee Break
3:30 – 4:00:	CUDA memory optimization (Eric)
4:00 – 4:15:	CUDA programming environment (Ziyi)
4:15 – 4:45:	Parallelism in CT reconstruction (Klaus)
4:45 – 5:25:	CT reconstruction examples (Eric)
5:25 – 5:30:	Closing remarks (Klaus)

Course Schedule

1:30 – 1:45:	Introduction
1:45 – 2:15:	Introductory code examples
2:15 – 2:30:	Parallel programming primer
2:30 – 3:00:	Parallelism in CT reconstruction
	Coffee Break
3:30 – 3:45:	GPU hardware
3:45 – 4:30:	CUDA API, threads, memory, performance optimization
4:30 – 4:45:	CUDA programming environment
4:45 – 5:25:	CT reconstruction examples
5:25 – 5:30:	Closing remarks