UPPSALA UNIVERSITET

Classification of algorithms and metrics for CPU/GPU architectures

Dimitar Lukarski

Division of Scientific Computing
Department of Information Technology
Uppsala Programming for Multicore Architectures Research Center (UPMARC)
Uppsala University

Programming of Parallel Computers, March, 2014

UPPSALA UNIVERSITET

Outline
Intro and motivation
Computational intensity
Hardware model
Accelerators
Top500
Programming
Performance only?
Open issues
Assignment
References

What You Should Know...

- Programming in Fortran/C/C++/Java
- Basics in hardware - CPU (FP, Cache), RAM, HDD,...
- Concepts of parallel programming
- To have practice - OpenMP, pthreads, MPI,...
- Know some metric - Amdahl's law, Gustafson's law,...
- Parallel programming is FUN!

D. Lukarski, March, 2014, Uppsala

Outline

Why multi/many-cores?

Power wall:

- Power(Fast Core) > Power(2xSlower Cores)
- Giga-Hertz era has come to an end
- Heat and cooling

Memory wall:

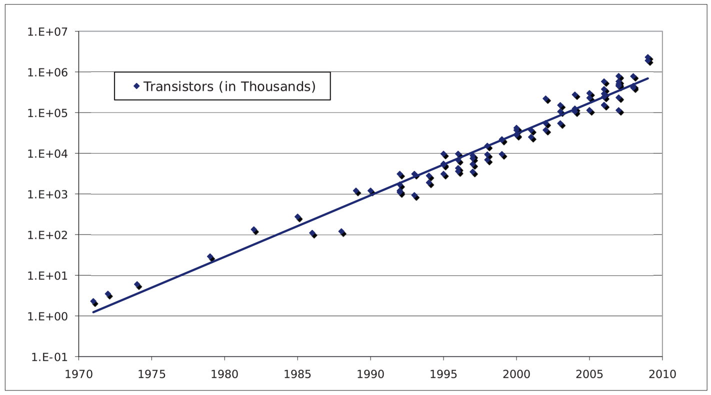
- More cycles are needed to get data from the main memory
- Memory bandwidth become performance limit in many cases

Instruction level of parallelism (ILP) wall:

- Complex mechanism −> larger area of the chip
- Limited scalability

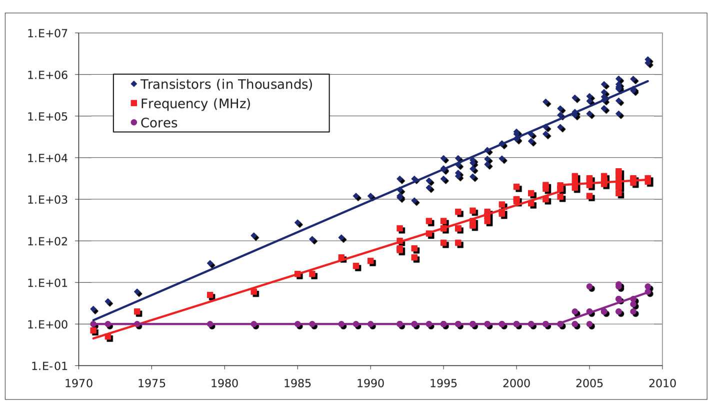
UPPSALA UNIVERSITET

Moore's Law



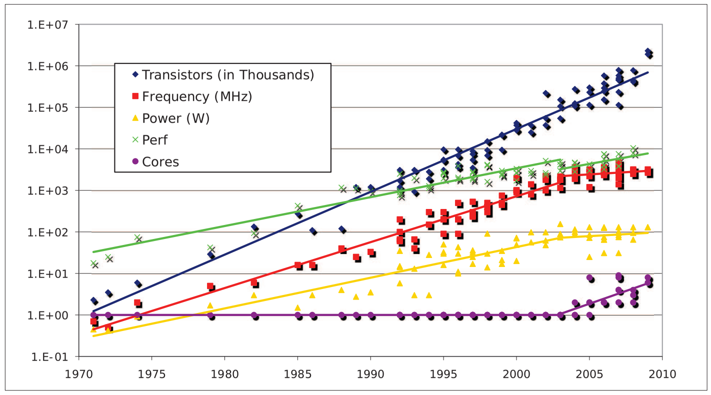
UPPSALA UNIVERSITET

Moore's Law, Frequency, Cores



UPPSALA UNIVERSITET

Moore's Law, Frequency, Cores, Power, Perf



UPPSALA UNIVERSITET

Outline
Intro and motivation
Computational intensity
Hardware model
Accelerators
Top500
Programming
Performance only?
Open issues
Assignment
References

Power Efficiency

Power consumption $P \sim f\ V^2 \sim V^3$

- f - clock
- V - voltage

From 1 core to 2 cores CPUs

- Decrease by 75% both f and V
- P will decease by 0.84 $(= 2 \times 0.75^3)$

50% more performance and 15% less energy

- Performance gain: $1.5\times$ (2×0.75)
- Performance/Watt: $1.8\times$ (1.5×0.84)

This is only a theoretical model!

Multi/many-cores Device are Everywhere!

- Clusters, HPC systems
- PCs
- PCs + GPUs
- Tablets
- Mobile phones
- Embedded systems-GPS, DVD players, PS3, Xbox,...

UPPSALA
UNIVERSITET

Outline
Intro and motivation
Computational intensity
Hardware model
Accelerators
Top500
Programming
Performance only?
Open issues
Assignment
References

D. Lukarski, March, 2014, Uppsala

UPPSALA UNIVERSITET

What is the Limit of Our Computer?

- If I buy faster memory - will my program run faster?
- If I buy faster (perf) CPU - will my program run faster?
- If I add a fast accelerator - will my program run faster?
- If yes - how fast?

It all depends on your algorithm!

UPPSALA UNIVERSITET

Computational Intensity

$CI = \frac{\#\ \text{Flop}}{\#\ \text{Data transfer}},$

Flop - floating point operations per data
Data transfer - the number of data needed for the operation

UPPSALA UNIVERSITET

Computational Intensity - Dot Product

Example
$x^T y = \alpha$, where $x, y \in \mathbb{R}^N$ and $\alpha \in \mathbb{R}$

To compute α we need to perform $2N$ flops (mult and add per element).
The data transfer is $2N + 1$.

$$CI = \frac{2N}{2N+1} = \frac{O(N)}{O(N)}$$

UPPSALA UNIVERSITET

Outline
Intro and motivation
Computational intensity
Hardware model
Accelerators
Top500
Programming
Performance only?
Open issues
Assignment
References

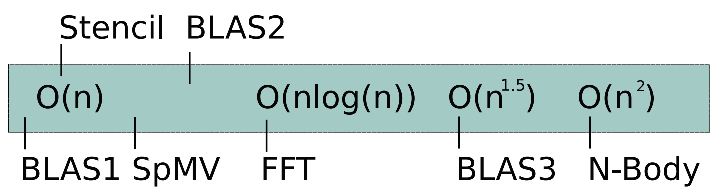
Computational Intensity - Dense Matrix-Matrix Multiplication

Example
$A \times B = C$, where $A, B, C \in \mathbb{R}^{N \times N}$

To compute C we need to perform N^2 dot products,
for each dot product we need perform $2N$ flops (mult and add per element).
The data transfer is $N^2 + N^2 + N^2$.

$$CI = \frac{2N^3}{3N^2} = \frac{O(N^{1.5})}{O(N)}$$

Computational Complexity of Algorithms



Note: Complexity $\neq$ Intensity

Hardware Model on Multi/Many-core Chips

Computation
- Flop per sec
- Vector intrinsic (MMX, SSE, ...)
- More cores – > more computation power

Communication
- Byte per sec
- Different memory controllers
- Different interconnects between the cores

Locality
- Local stores (CPU:Caches, GPU:Shared memory)
- Different structures, performance, capabilities

UPPSALA UNIVERSITET

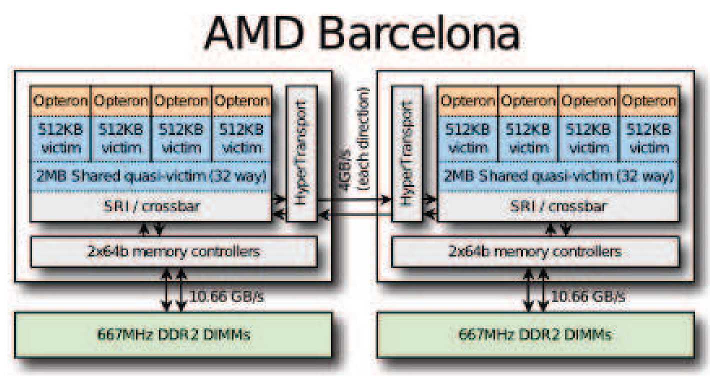
Outline
Intro and motivation
Computational intensity
Hardware model
Accelerators
Top500
Programming
Performance only?
Open issues
Assignment
References

Hardware Model on Multi/Many-core Chips

Hardware
- Different structure/organization
- Different memory sizes/bandwidth

Program
- Different memory usage
- Different memory access (pattern)
- Different computational intensity

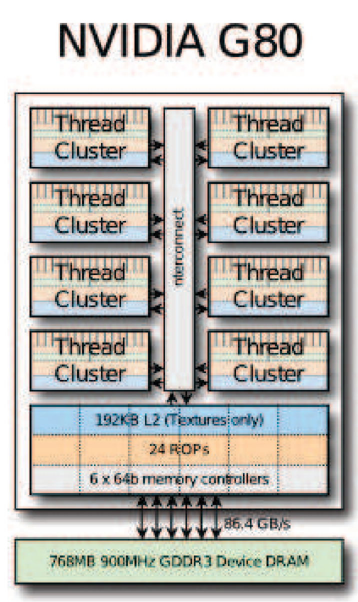
A Multi-core CPU



UPPSALA
UNIVERSITET

Outline
Intro and motivation
Computational intensity
Hardware model
Accelerators
Top500
Programming
Performance only?
Open issues
Assignment
References

A GPU



D. Lukarski, March, 2014, Uppsala

Instruction Level Parallelism

A+B=C
D+F=M
C+M=E

- Read A, Read, B, Add A+B, Write C
- Read D, Read, F, Add D+F, Write M
- Read C, Read, M, Add C+M, Write E

Pipelining

- Read A,B,D,F
- In parallel compute C, when A,B are ready
- In parallel compute M, when D,F are ready
- In parallel compute E, when C,M are ready
- Write C,M,E

UPPSALA UNIVERSITET

Outline
Intro and motivation
Computational intensity
Hardware model
Accelerators
Top500
Programming
Performance only?
Open issues
Assignment
References

Instruction Level Parallelism

In most of the cases, this is handled by the compiler/hardware!

To improve, be aware of

- Branching - if-else cases
- General dependencies (e.g. parallel but hidden in memory pointers)

Vector Intrinsic - SIMD

A+B=C
D+F=M
all are int/float/double

Combine (A,D) + (B,F) = (C,M), where (,) is twice the size of int/float/double

- Single Instruction, Multiple Data (SIMD)
- 128, 256, 512-bit width
- Typical usage - image processing

UPPSALA UNIVERSITET

Outline
Intro and motivation
Computational intensity
Hardware model
Accelerators
Top500
Programming
Performance only?
Open issues
Assignment
References

Fused Multiply-Add

A*B+C=C

There is FMADD operation which can be performed in one cycle

- Supported on many devices - CPUs, GPUs
- Higher accuracy
- Typically, the compiler takes care of that

D. Lukarski, March, 2014, Uppsala

Memory Prefetching

Prefetch data from main memory while computing something else.

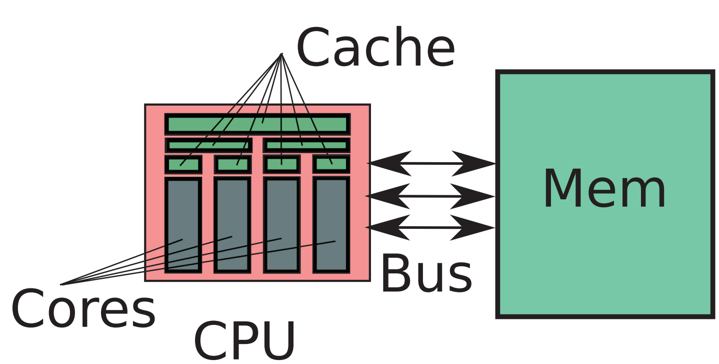
Overlapping computation and communication

- Typically handle by the compiler/hardware
- See also - ILP

UPPSALA
UNIVERSITET

Outline
Intro and motivation
Computational intensity
Hardware model
Accelerators
Top500
Programming
Performance only?
Open issues
Assignment
References

Uniform Memory Access - UMA



D. Lukarski, March, 2014, Uppsala

UPPSALA UNIVERSITET

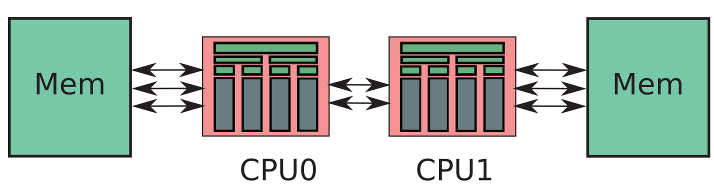
Uniform Memory Access - UMA

- All cores/threads share the same bandwidth
- The memory controller is out of the CPUs

UPPSALA UNIVERSITET

Outline
Intro and motivation
Computational intensity
Hardware model
Accelerators
Top500
Programming
Performance only?
Open issues
Assignment
References

Non-Uniform Memory Access - NUMA



D. Lukarski, March, 2014, Uppsala

Non-Uniform Memory Access - NUMA

- The threads/cores have fast access to their own memory
- The threads/cores have slow access to others memory
- Specific allocation can be made with **numactl**
- **numactl** - terminal or API
- Default is Round-Robin

UPPSALA UNIVERSITET

Outline
Intro and motivation
Computational intensity
Hardware model
Accelerators
Top500
Programming
Performance only?
Open issues
Assignment
References

Simple Hardware Model

When talking about large data – in many cases we can ignore the locality of the hardware (simplify the model).

Computational time $= F + M$

- F =time for flops - floating point operations
- M =time for data transfer

Computational time $= max(F, M)$ with overlapping computation and communication.

Ideally our algorithm will be bound either by the computational bound or by the bandwidth bound of the hardware!

D. Lukarski, March, 2014, Uppsala

UPPSALA UNIVERSITET

Example: Dot product

Assume to have a computer with

- Memory bandwidth 25 GByte/s
- Computational power 40 GFlop/s

And we want to perform a dot product $\alpha = x^T y$

- x, y are of size N, stored in double precision
- Data $= 2N + 1$ and Flop (for the dot product) $= 2N$

What will be the performance of the dot product on this machine? (for N very large)

UPPSALA UNIVERSITET

Outline
Intro and motivation
Computational intensity
Hardware model
Accelerators
Top500
Programming
Performance only?
Open issues
Assignment
References

Example: Dot product

- We need to perform $2N$ operations on $2N + 1$ data
- 25 GByte/s = 3.125 (=25/8) GDouble/s
- 3.125 $\times$ 2 (mult and add) / 2 (two vectors) = 3.125 GFlop/s
- Assuming communication and computation overlapping
- $P = min(40, 3.125) = 3.125$ GFlop/s
- This is 7.5% of the peak performance
- This is 100% of the memory bandwidth

Example: Matrix-Matrix Multiplication

Assume to have a computer with

- Memory bandwidth 25 GByte/s
- Computational power 40 GFlop/s

And we want to perform a dot product $A \times B = C$

- A, B, C are real matrices $N \times N$, stored in double precision
- Data $= 3N^2$ and Flop (for mm mult) $= 2N^3$

What will be the performance of the matrix-matrix multiplication on this machine? (for N very large)

UPPSALA UNIVERSITET

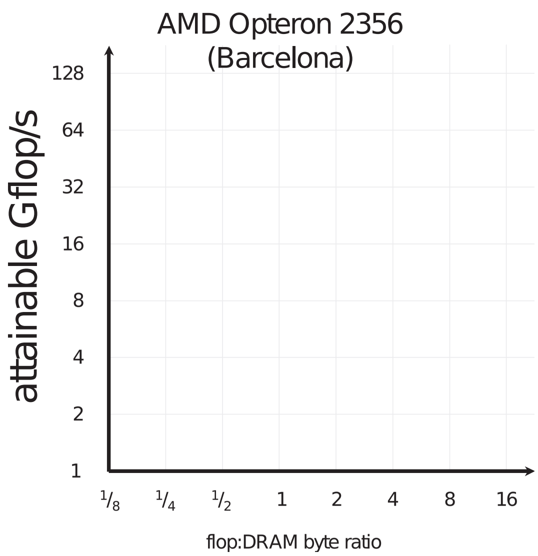
Outline
Intro and motivation
Computational intensity
Hardware model
Accelerators
Top500
Programming
Performance only?
Open issues
Assignment
References

Example: Matrix-Matrix Multiplication

- We need to perform $2N^3$ operations on $3N^2$ data
- Assuming communication and computation overlapping
- $P = min(40, O(N)) = 40$ GFlop/s, for N-very large
- This is 100% of the peak performance
- The bandwidth cannot be estimated based on this model (due to locality)

D. Lukarski, March, 2014, Uppsala

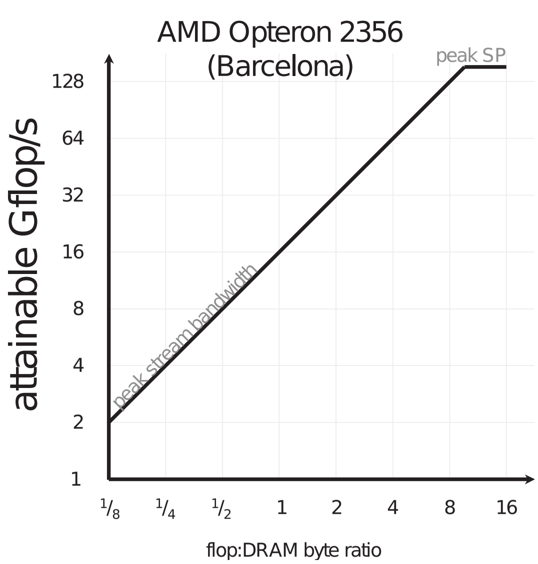
Roofline Model



UPPSALA UNIVERSITET

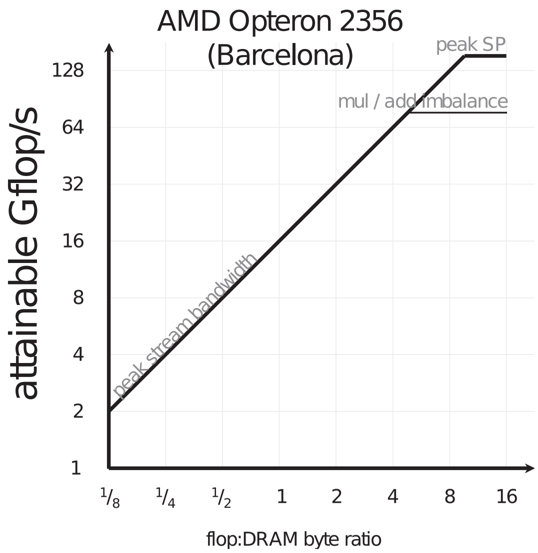
Outline
Intro and motivation
Computational intensity
Hardware model
Accelerators
Top500
Programming
Performance only?
Open issues
Assignment
References

Roofline Model



UPPSALA UNIVERSITET

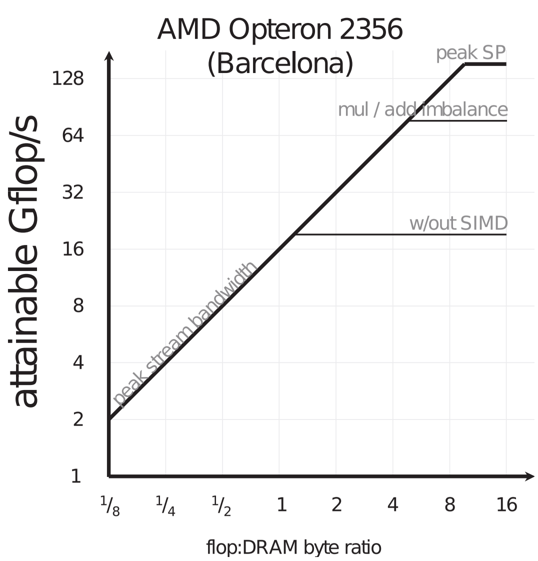
Roofline Model



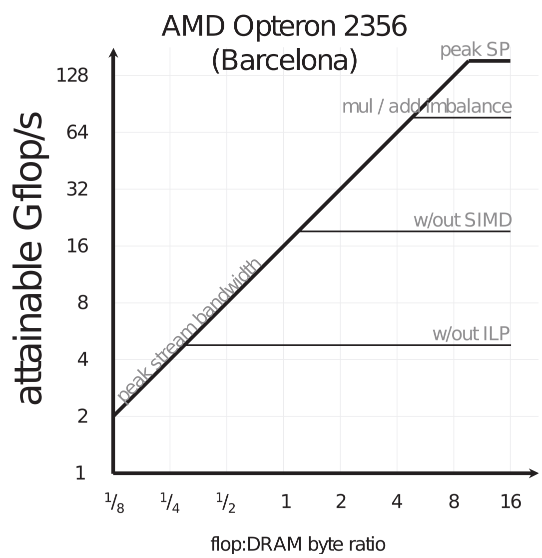
UPPSALA UNIVERSITET

Outline
Intro and motivation
Computational intensity
Hardware model
Accelerators
Top500
Programming
Performance only?
Open issues
Assignment
References

Roofline Model



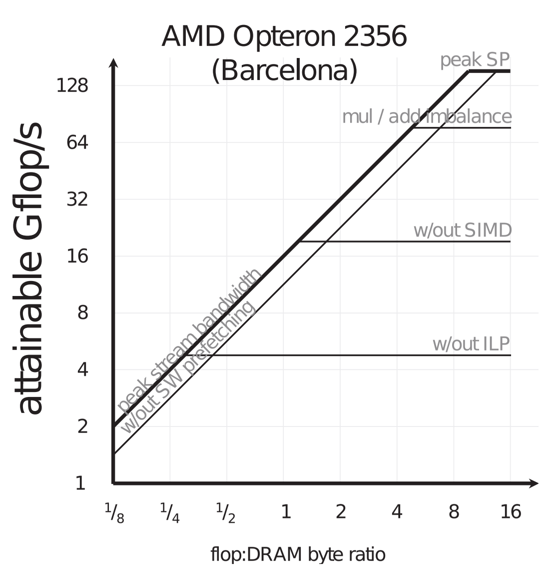
Roofline Model



UPPSALA
UNIVERSITET

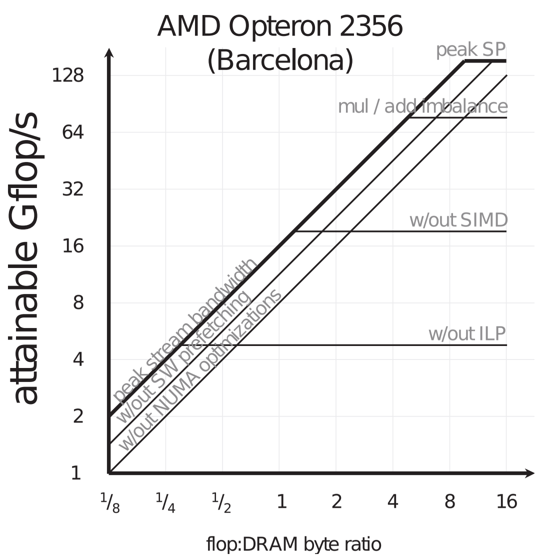
Outline
Intro and motivation
Computational intensity
Hardware model
Accelerators
Top500
Programming
Performance only?
Open issues
Assignment
References

Roofline Model



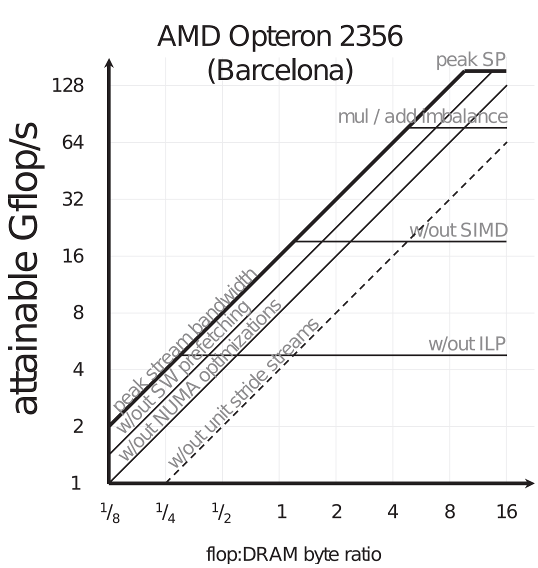
UPPSALA UNIVERSITET

Roofline Model

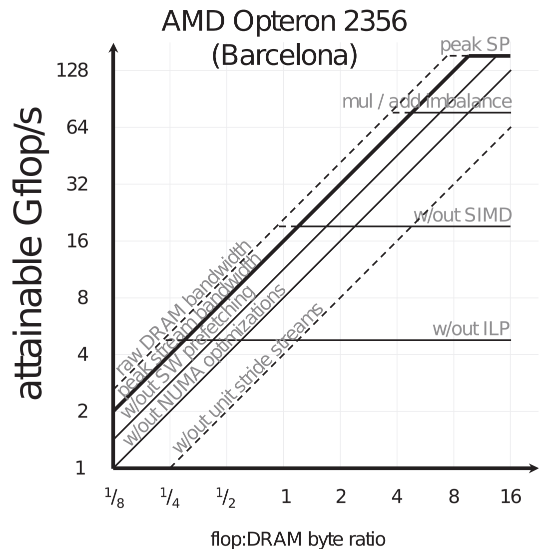


UPPSALA UNIVERSITET

Roofline Model



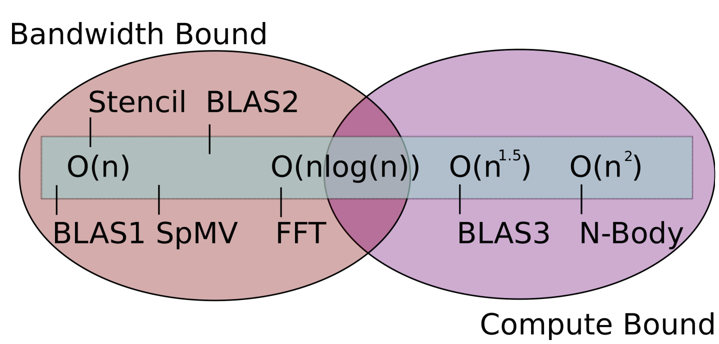
Roofline Model



UPPSALA UNIVERSITET

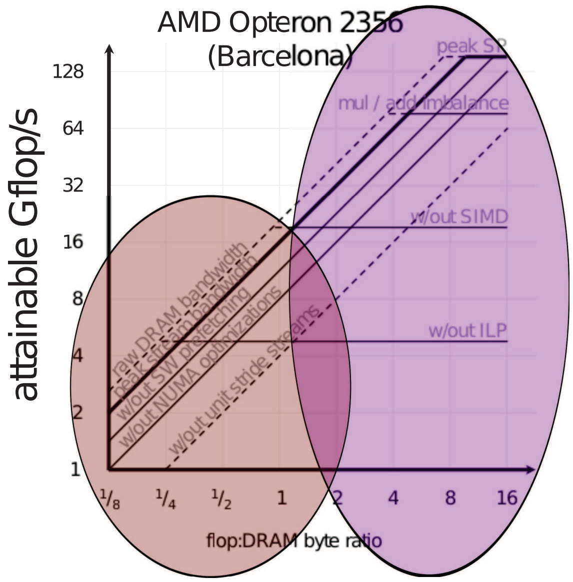
Outline
Intro and motivation
Computational intensity
Hardware model
Accelerators
Top500
Programming
Performance only?
Open issues
Assignment
References

Computational Complexity of Algorithms



D. Lukarski, March, 2014, Uppsala

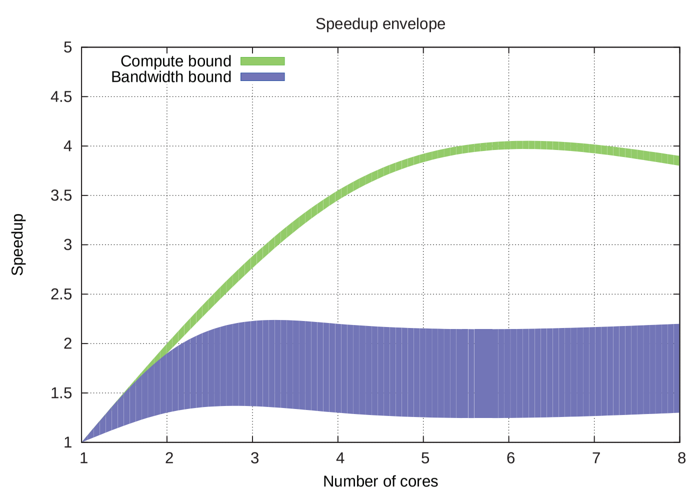
Typical Optimization Scenario



UPPSALA UNIVERSITET

Outline
Intro and motivation
Computational intensity
Hardware model
Accelerators
Top500
Programming
Performance only?
Open issues
Assignment
References

Typical Speed-up Numbers



- i7 SandyBridge (4cores+4HT)

UPPSALA UNIVERSITET

Outline
Intro and motivation
Computational intensity
Hardware model
Accelerators
Top500
Programming
Performance only?
Open issues
Assignment
References

Accelerators

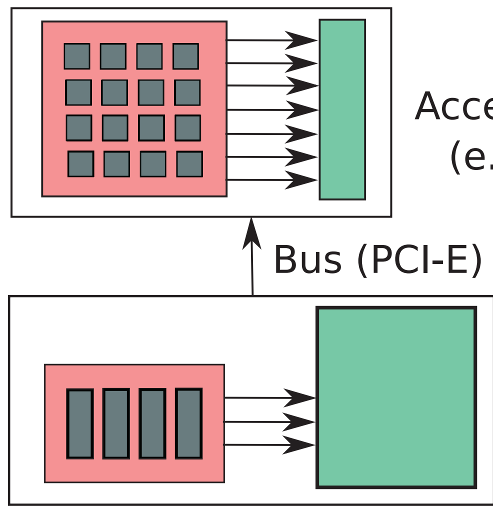
Additional devices which can perform some arithmetic operations faster than the CPU

- Co-processors (1970s-1990s)
- Special accelerator devices – ClearSpeed, Convey
- GP-GPU – General Purpose GPU

D. Lukarski, March, 2014, Uppsala

UPPSALA UNIVERSITET

Accelerators in the Computer



UPPSALA UNIVERSITET

Accelerators in the Computer

Bandwidth

- Accelerator memory - very fast
- Host memory - fast
- PCI-E bus - slow
- Network - slow
- Hard disk - very slow

Capacity

- Hard disk - very large
- Host memory - large
- Accelerator memory - small (2-8GB)

Compute capabilities

- Host CPU - few fat cores (they do everything!)
- Accelerator chip - many, small, specialized (Flop/s)

Example: Dot product

Assume to have a computer with

- Memory bandwidth 25 GByte/s
- Computational power 40 GFlop/s

and accelerator attached to it, with

- Memory bandwidth 100 GByte/s
- Computational power 500 GFlop/s
- PCI-E bus 3 GByte/s

And we want to perform a dot product $\alpha = x^T y$

- x, y are of size N, stored in double precision
- Data $= 2N + 1$ and Flop (for the dot product) $= 2N$

What will be the performance of the dot product on the host/accelerator? If the data is on the host/accelerator. (N=10M)

Example: Dot product

- We need to perform $2N$ operations on $2N + 1$ data
- 100 GByte/s = 12.5 (=100/8) GDouble/s
- 12.5 × 2 (mult and add) / 2 (two vectors) = 12.5 GFlop/s
- Assuming communication and computation overlapping
- $P = min(100, 12.5) = 12.5$ GFlop/s
- 12.5 GDouble/s for 2×10M elements = 0.0016 sec
- CPU: 3.125 GDouble/s for 2×10M elements = 0.0064 sec
- If we need to copy the data
- PCI-E 3 GByte/s = 0.37 GDouble/s, time = 0.027 sec
- In total 0.027 + 0.0016 = 0.04 sec

Example: Matrix-Matrix Multiplication

Assume to have a computer with

- Memory bandwidth 25 GByte/s
- Computational power 40 GFlop/s

and accelerator attached to it, with

- Memory bandwidth 100 GByte/s
- Computational power 500 GFlop/s
- PCI-E bus 3 GByte/s

And we want to perform a dot product $A \times B = C$

- A, B, C are real matrices $N \times N$, stored in double precision
- Data $= 3N^2$ and Flop (for mm mult) $= 2N^3$

What will be the performance of the dot product on host/accelerator? If the data is on the host/accelerator. (N=6000)

Example: Matrix-Matrix Multiplication

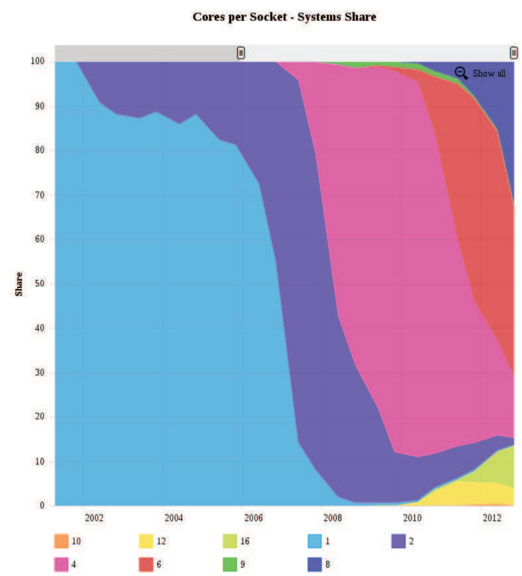
- We need to perform $2N^3$ operations on $3N^2$ data
- Data $= 3\ 8\ NN = 823.9$ MByte
- Flop $= 2N^3 = 402.3$ GFlop
- Assuming communication and computation overlapping
- Time $= 402.3/500 = 0.8$sec
- CPU: 40 Gflop/s $= 10.05$ sec
- If we need to copy the data
- PCI-E 3 GByte/s $= 0.37$ GDouble/s, 2.2 sec
- In total $0.8 + 2.2 = 3$ sec

Best Practice

Keep the data in the accelerator!
Avoid PCI-E communication!

UPPSALA UNIVERSITET

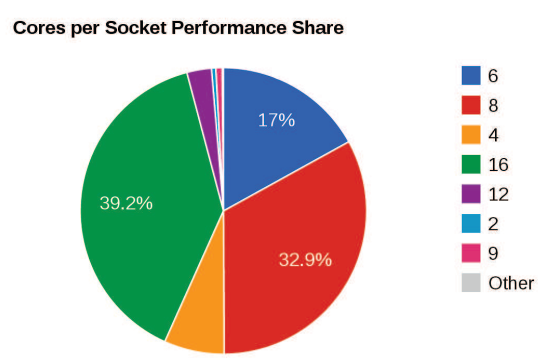
Top 500



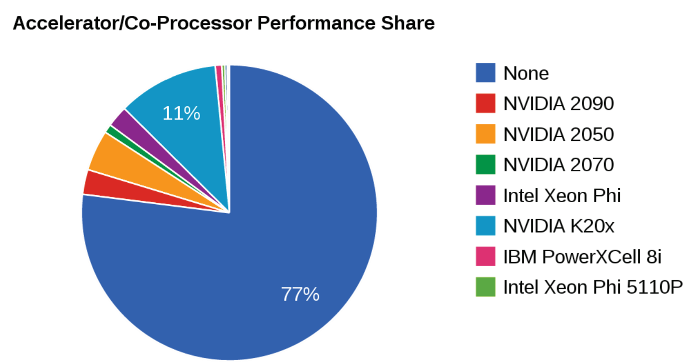
UPPSALA UNIVERSITET

Outline
Intro and motivation
Computational intensity
Hardware model
Accelerators
Top500
Programming
Performance only?
Open issues
Assignment
References

Top 500 / Nov 2012



Top 500 / Nov 2012



UPPSALA UNIVERSITET

Outline
Intro and motivation
Computational intensity
Hardware model
Accelerators
Top500
Programming
Performance only?
Open issues
Assignment
References

D. Lukarski, March, 2014, Uppsala

Programming Tips

You need to

- modify your code
- modify your algorithm
- modify you data structure

Exampe: Dot product

UPPSALA
UNIVERSITET

Outline
Intro and motivation
Computational intensity
Hardware model
Accelerators
Top500
Programming
Performance only?
Open issues
Assignment
References

Programming Languages

pthreads

Good

- Manual launch - optimal
- Manual threads - optimal communication
- Manual threads - optimal synchronization

Bad

- Manual launch - not easy
- Manual threads - hard / deadlocks
- Manual threads - hard / deadlocks

D. Lukarski, March, 2014, Uppsala

Programming Languages

OpenMP

Good

- Automatic - easy to use
- Implicit communication
- Easy synchronization

Bad

- Automatic - most optimal?
- Implicit communication - dataracing
- Synchronization - not very robust
- Not super generic approach
- Not very scalable approach

UPPSALA
UNIVERSITET

Outline
Intro and motivation
Computational intensity
Hardware model
Accelerators
Top500
Programming
Performance only?
Open issues
Assignment
References

Programming Languages

CUDA

Good

- Very scalable model
- Optimal with respect to the hardware

Bad

- No global synchronization mechanism
- NVIDIA GPU only
- Hard to express algorithms in this model

D. Lukarski, March, 2014, Uppsala

UPPSALA UNIVERSITET

Programming Languages

OpenCL

Good

- Very scalable model
- Can run on various hardware

Bad

- No global synchronization mechanism
- Good performance - only with optimized kernels
- Optimized kernels - not portable across different hardware

UPPSALA UNIVERSITET

Performance vs. Productivity

Performance

- Parallel and better programming
- Goal: faster program
- Why: time is money

Productivity

- Easy and simple programming
- Goal: faster results
- Why: time is money

UPPSALA UNIVERSITET

Outline
Intro and motivation
Computational intensity
Hardware model
Accelerators
Top500
Programming
Performance only?
Open issues
Assignment
References

Performance vs. Portability

Performance

- Parallel and better programming
- Goal: faster program
- Why: time is money

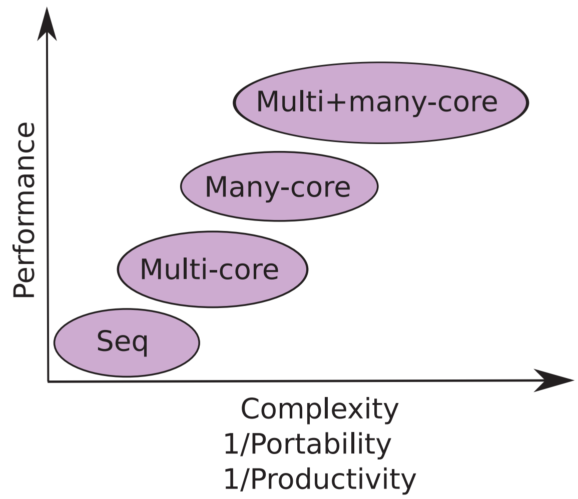
Portability

- High level languages are slow but portable (e.g. MATLAB)
- Goal: the program should run everywhere (old, current, future PCs)
- Why: cannot re-write all software for every new hardware

D. Lukarski, March, 2014, Uppsala

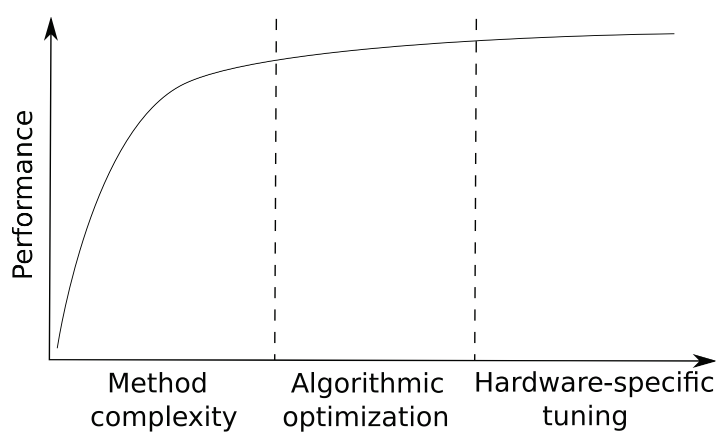
UPPSALA UNIVERSITET

Good Parallel Programming is Hard



UPPSALA UNIVERSITET

Methods are the MOST Imporant



UPPSALA UNIVERSITET

Your Future Work Might be...

- Specific software - Speed it up!
- General languages - Speed it up!
- Green it up!

UPPSALA UNIVERSITET

UPPSALA UNIVERSITET

Your Not Compulsory Assignment 1/2

For a vector update of type $x = x + \alpha y$, where $x, y \in \mathbb{R}^N$ and $\alpha \in \mathbb{R}$

- Compute the computational intensity

If the size is $N = 20M$, stored in double prec. and we have a computer with

- 56 GFlop/s peak performance
- 30 GByte/s peak bandwidth

What is the peak performance? And the total execution time, if we can and cannot overlap computation and communication?

UPPSALA UNIVERSITET

Outline
Intro and motivation
Computational intensity
Hardware model
Accelerators
Top500
Programming
Performance only?
Open issues
Assignment
References

Your Not Compulsory Assignment 2/2

We have an algorithm which performs 1000 dot products and 5000 vector updates (2 input vectors, 1 output vector). What is the speed up of such algorithm if we offload it to an accelerator.
Accelerator:

- 1 TFlop/s peak performance
- 180 GByte/s peak bandwidth
- PCI-E bus 5GByte/s

(the size is again 20M, double)

What is the peak performance (accel. only)? What is the speed up of such algorithm if we store the input output results on the host memory

D. Lukarski, March, 2014, Uppsala

UPPSALA UNIVERSITET

Outline
Intro and motivation
Computational intensity
Hardware model
Accelerators
Top500
Programming
Performance only?
Open issues
Assignment
References

References (from this talk)

- Talk: "Ten Ways to Waste a Parallel Computer" - Kathy Yelick (Berkeley)
- Talk: "The Roofline Model: A pedagogical tool for program analysis and optimization" - Samuel Williams (Berkeley)

D. Lukarski, March, 2014, Uppsala

Resources

- Google is your friend!
- Search for lectures, tutorials, video materials
- Computer Architecture: A Quantitative Approach, 4th Edition - Patterson
- Programming Massively Parallel Processors: A Hands-On Approach - CUDA
- Patterns for Parallel Programming (Software Patterns)