Computational Science
The Third Pillar of Science?

Emil Slușanschi

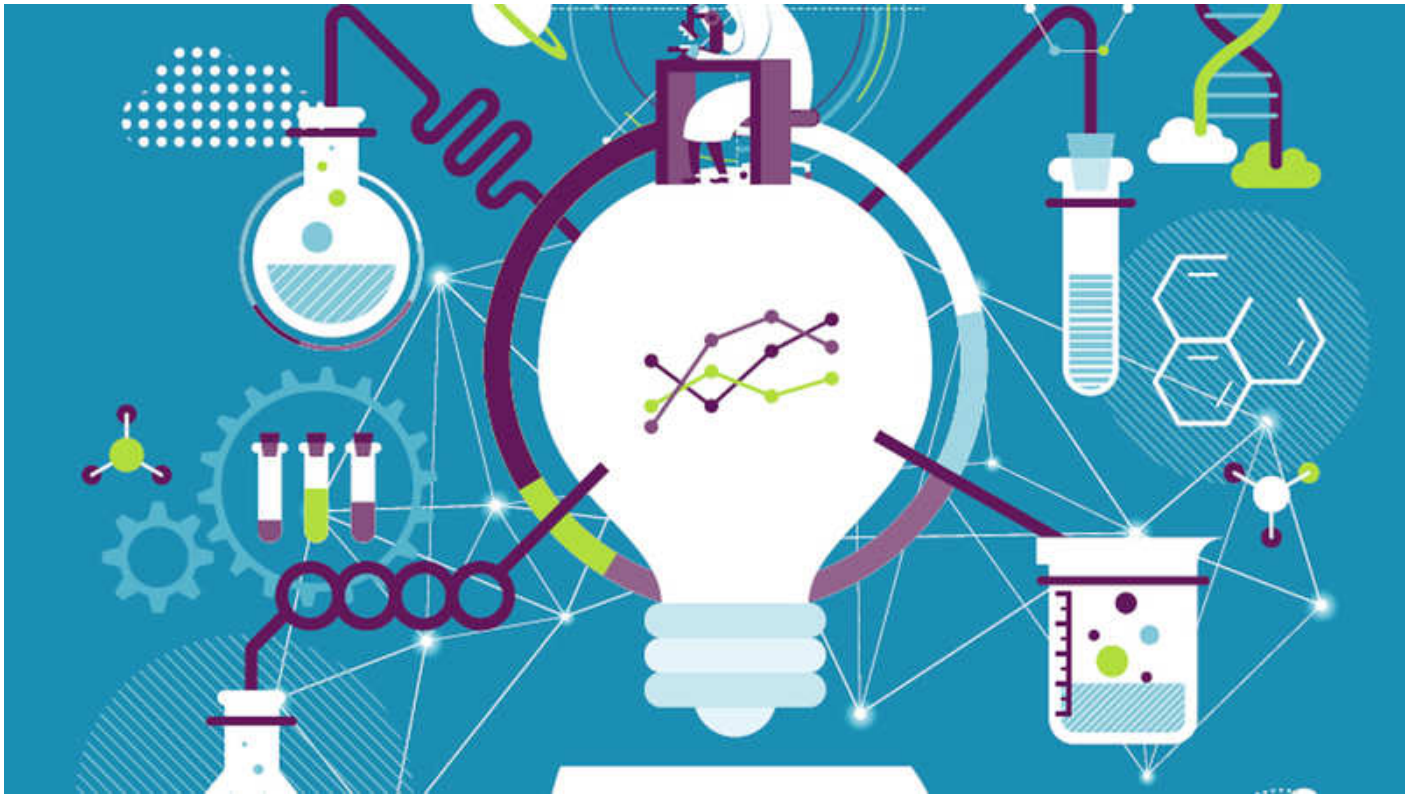
CERN School of Computing 2019

Cluj-Napoca, Romania

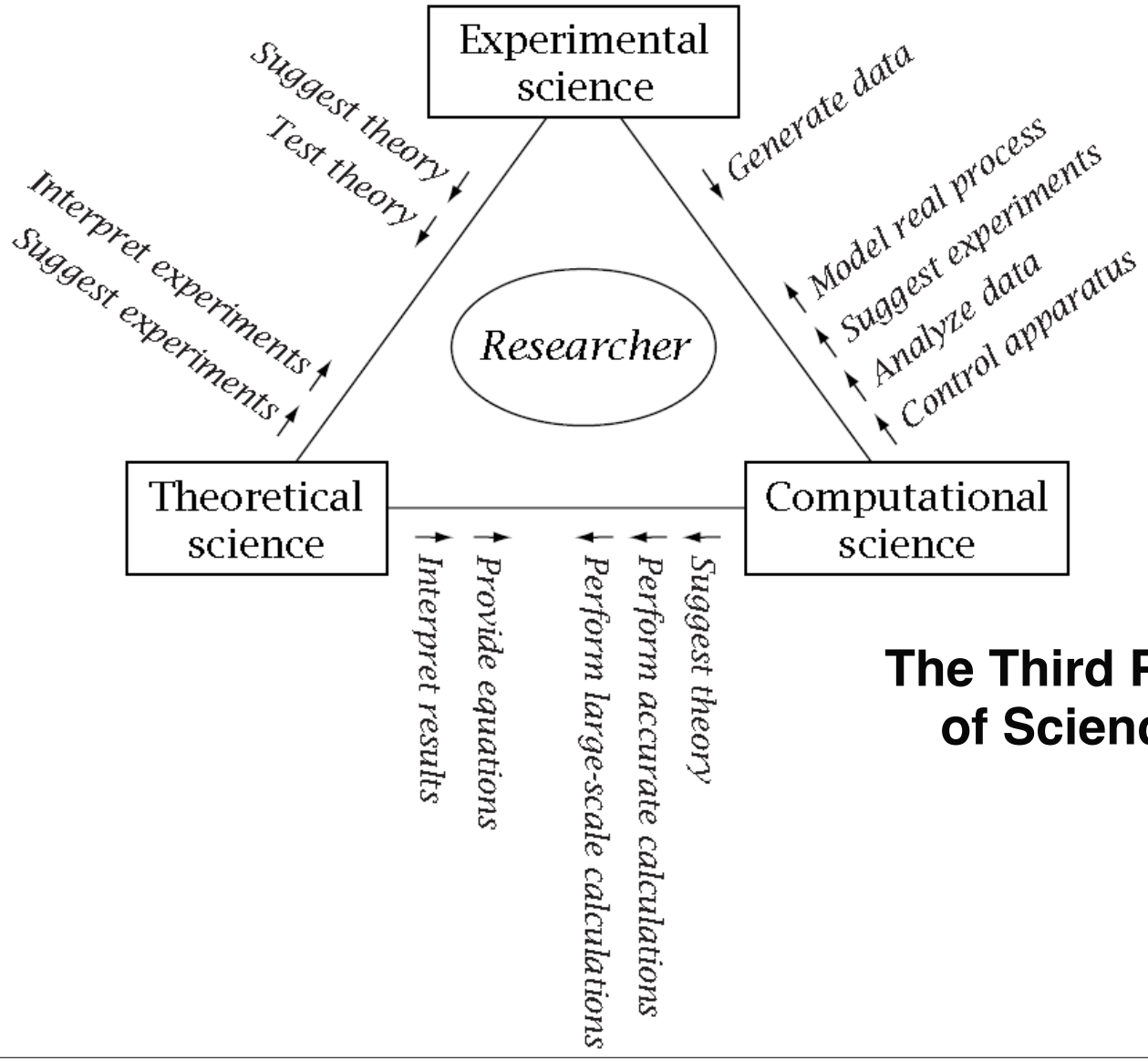
University Politehnica of Bucharest

Outline

- Scientific Research Paradigm
- The Evolution of Computing
- The Software Challenge
- The Roofline Model
- Going forward

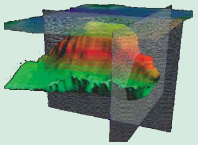


Scientific Research Paradigm

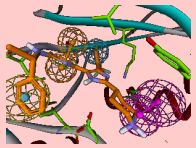


The Third Pillar of Science

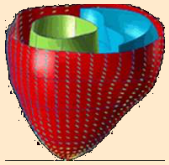
Future Science and Engineering Breakthroughs Hinge on **Computing**



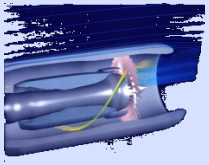
Computational Geoscience



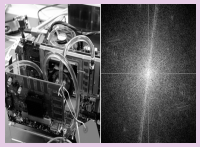
Computational Chemistry



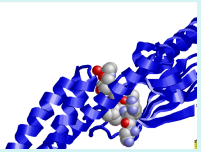
Computational Medicine



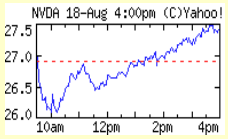
Computational Modeling



Computational Physics



Computational Biology



Computational Finance

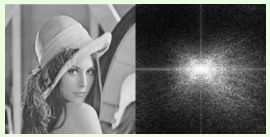


Image Processing

Major paradigm shift

- In the 20th Century, we were able to understand, design, and manufacture what we could **measure**
  - Physical instruments and computing systems allowed us to see farther, capture more, communicate better, understand natural processes, control artificial processes…
- In the 21st Century, we are able to understand, design, create what we can **compute**
  - Computational models are allowing us to see even farther, going back and forth in time, relate better, test hypothesis that cannot be verified any other way, create safe artificial processes

Examples of Paradigm Shift

20th Century	21st Century
• Small mask patterns and short light waves	• Computational optical proximity correction
• Electronic microscope and Crystallography with computational image processing	• Computational microscope with initial conditions from Crystallography
• Anatomic imaging with computational image processing	• Metabolic imaging sees disease before visible anatomic change
• Teleconference	• Tele-emersion

Faster is not “just Faster”

- 2-3X faster is “just faster”
  - Do a little more, wait a little less
  - Doesn’t change how you work
- 5-10x faster is “significant”
  - Worth upgrading
  - Worth re-writing (parts of) the application
- 100x+ faster is “fundamentally different”
  - Worth considering a new platform
  - Worth re-architecting the application
  - Makes new applications possible
  - Drives “time to discovery” and creates fundamental changes in Science

How much computing power is enough?

- Each jump in computing power motivates new ways of computing
  - Many apps have approximations or omissions that arose from limitations in computing power
  - Every 100x jump in performance allows app developers to innovate
  - Example: graphics, medical imaging, physics simulation, etc.

Users & application developers did not take computing seriously until they saw real results.

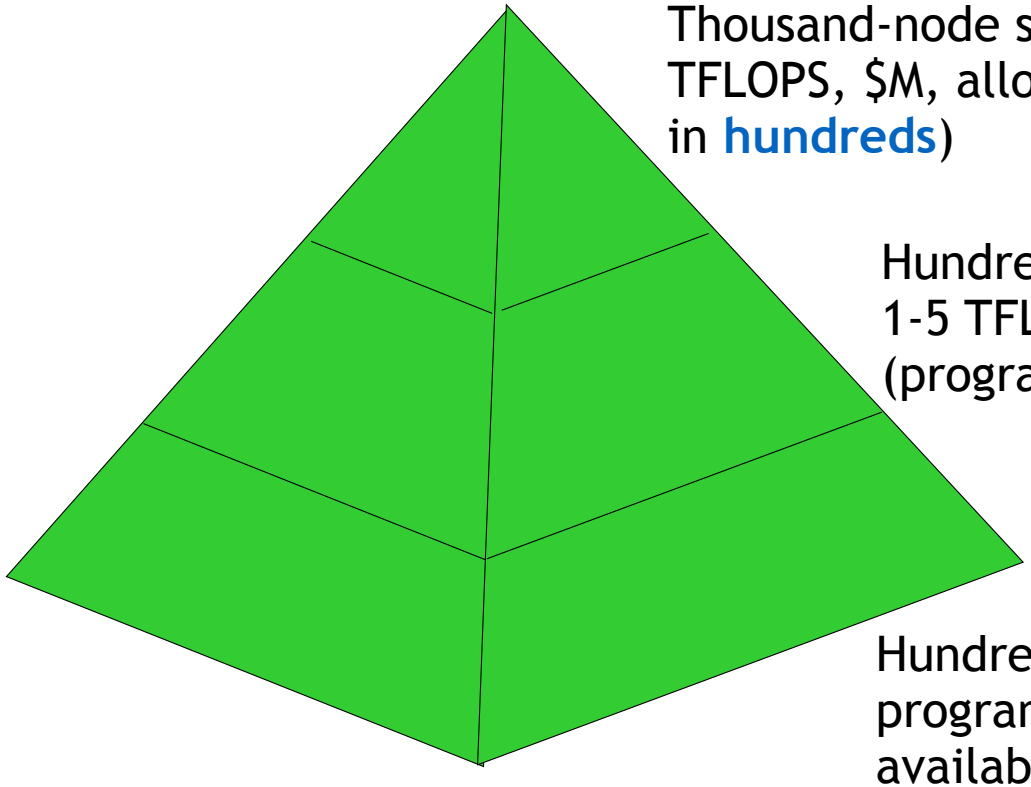
Why didn't this happen earlier?

- Computational experimentation is just reaching critical mass
  - Simulate large enough systems
  - Simulate long enough system time
  - Simulate enough details
- Computational instrumentation is also just reaching critical mass
  - Reaching high enough accuracy
  - Cover enough observations

A Great Opportunity for Many

- New massively parallel computing is enabling
  - Drastic reduction in “time to discovery”
  - 1st principle-based simulation at meaningful scale
  - New, 3rd paradigm for research: computational experimentation
- The “democratization” of power to discover
  - $2,000/Teraflops SPFP in personal computers today
  - $5,000,000/Petaflops DPFP in clusters in 2-3 years
  - HW cost will no longer be the main barrier for big science
- This is once-in-a-lifetime opportunity for many!

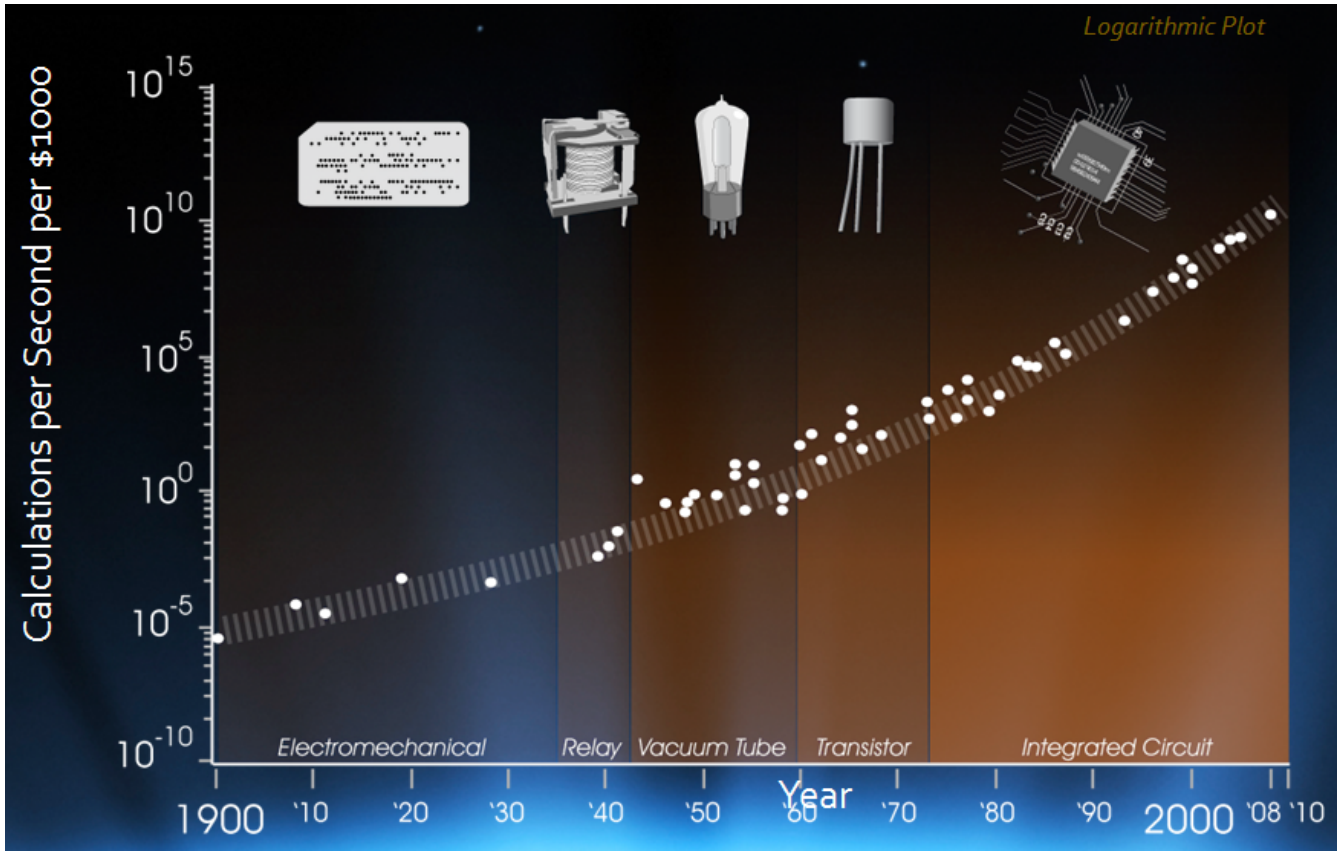
The Pyramid of Parallel Programming



Thousand-node systems with MPI-style programming, >100 TFLOPS, $M, allocated machine time (programmers numbered in **hundreds**)

Hundred-core systems with CUDA-style programming, 1-5 TFLOPS, $K, machines widely availability (programmers numbered in **10s of thousands**)

Hundred-core systems with MatLab-style programming, 10-100 GFLOPS, $K, machines widely available (programmers numbered in **millions**)

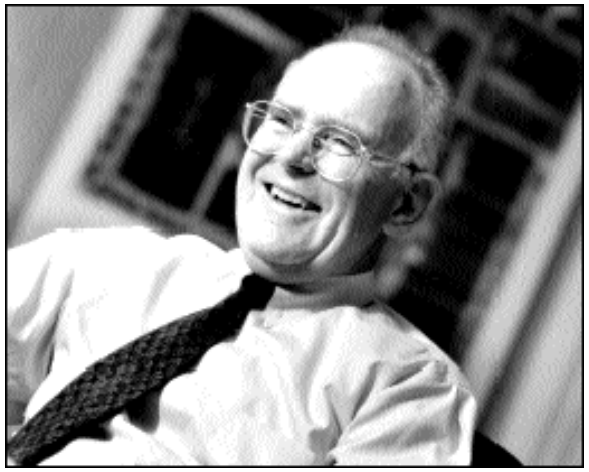


The Evolution of Computing

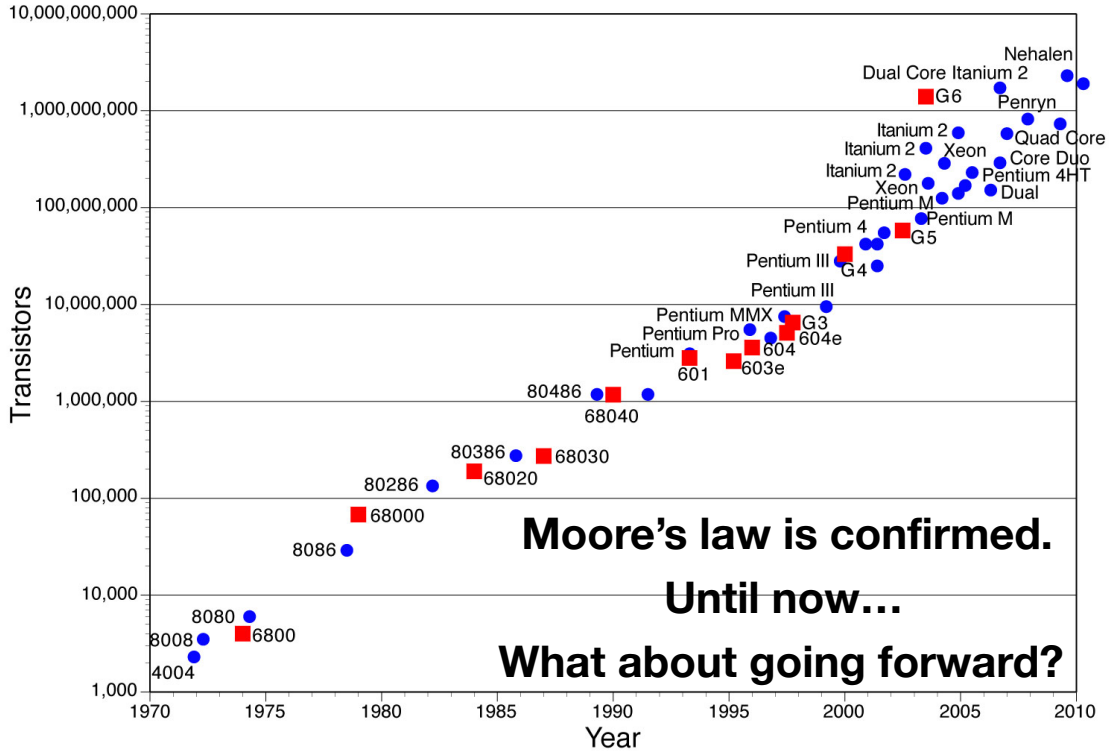
Words of Wisdom

- *"I think there is a world market for maybe five computers."*
  - Thomas Watson, chairman of IBM, 1943.
- *"There is no reason for any individual to have a computer in their home"*
  - Ken Olson, President of Digital Equipment Corporation, 1977.
- *"640KB [of main memory] ought to be enough for anybody."*
  - Bill Gates, Chairman of Microsoft / IBM System Limitation,1981.

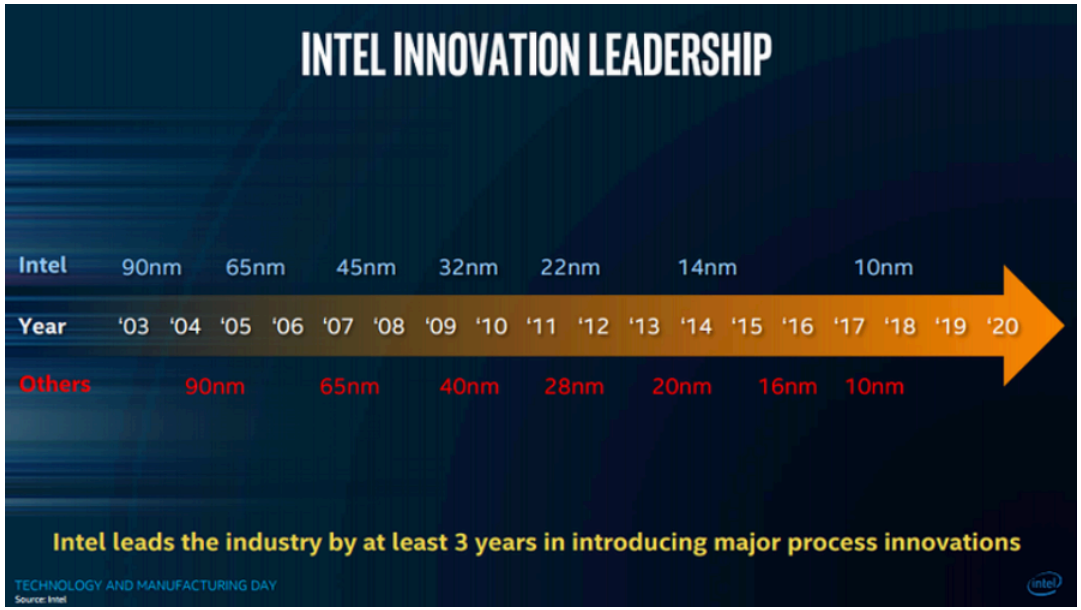
Processor Evolution



Gordon Moore (Intel cofounder) predicted in 1965 that transistor density in semiconductors will **double** each **18 months**

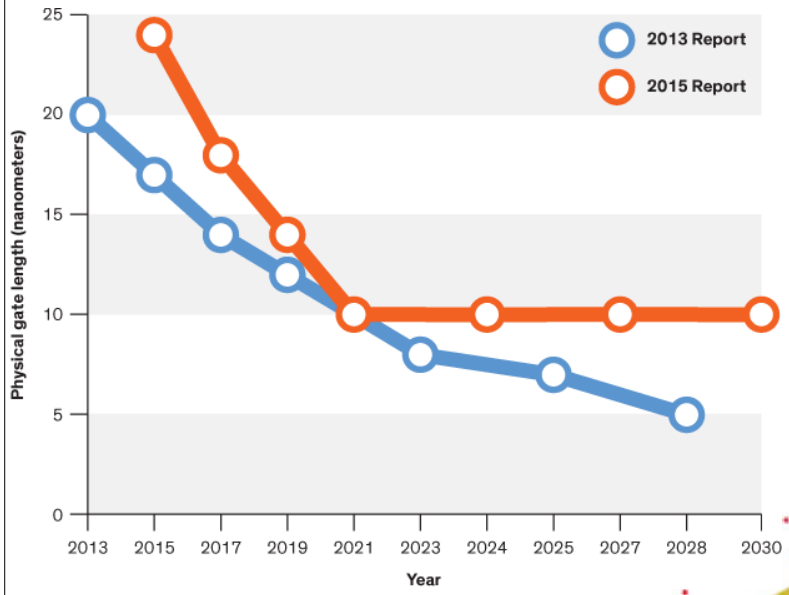


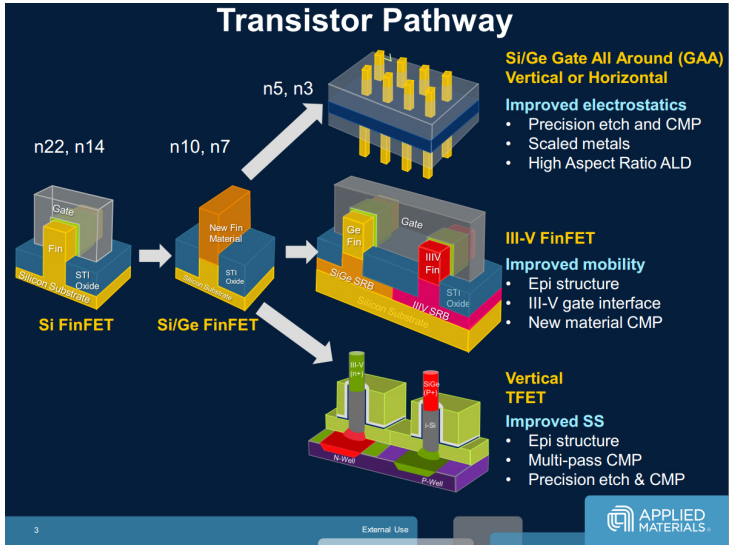
Intel Roadmap vs. Moore's Law

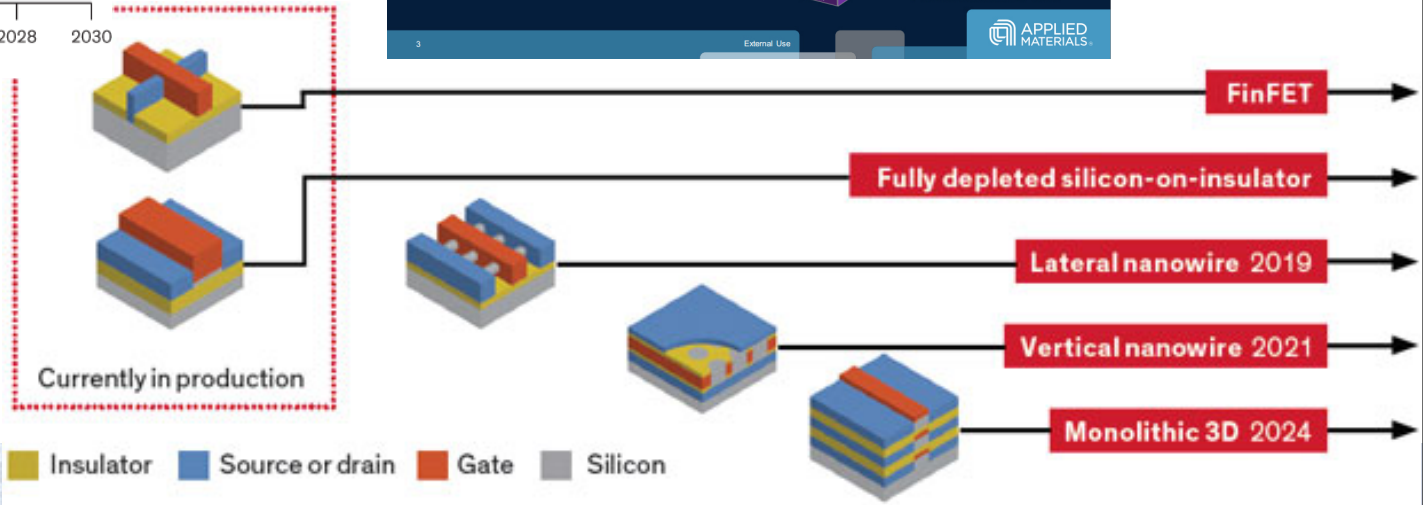


- Up until 1965: 2x / year
- 1965 - 201x: 2x / 2 years
- 201x - 202x: 2x / 3-4 years
- The 2x intervals keep getting longer…

ITRS Roadmap

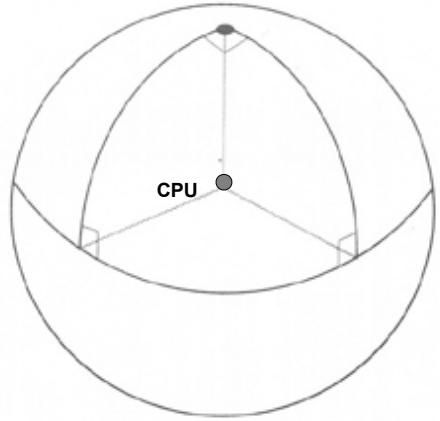






What about a 10 TFlops Processor?

- Can we build a serial CPU
  - Offering 10 TFlops?
  - Operating on 10 TByte of memory?
- Reprezentative for today's needs
- Processor clock should be 10,000 GHz @ 1 instruction / cycle
- Assume data travels at the speed of light c = $3e^8$ m/s
- Assume the processor is an ideal sphere

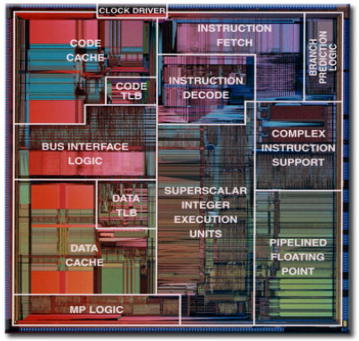


What about a 10 TFlops Processor?

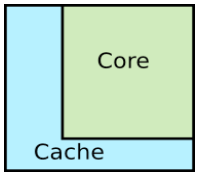
- Data should travel a (non-zero) distance from memory to CPU
  - Distance from memory to CPU should be $r < c / 10^{13} \sim 3e^{-6}$ m
- Each instruction requires at least 8 bytes of memory: 10^{13} bytes of memory in a volume of $4/3\pi r^3 = 3.7e^{-17}$ m^3
  - Each word of memory can occupy at most $3.7e^{-30} m^3 = 3.7$ $Angstrom^3$
  - A tiny molecule of a few atoms…
- Current memory density is about 10GB/cm^3
  - Factor 10^{20} below what is required!
- Bottom line: we can't do this with current technology

Pentium Processors Evolution

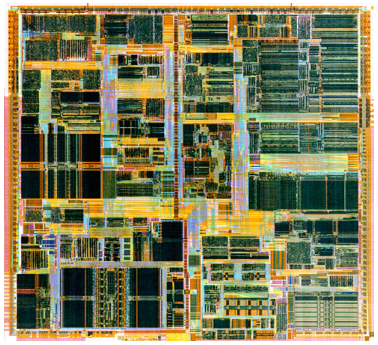
Pentium I

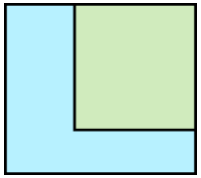


Chip area breakdown

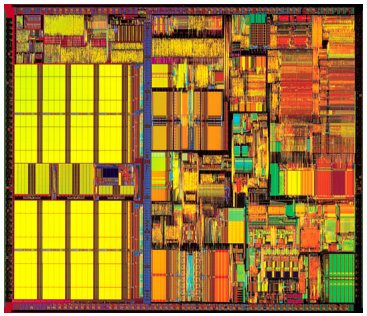


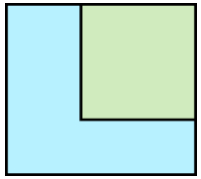
Pentium II



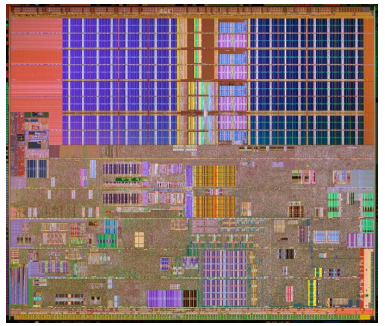


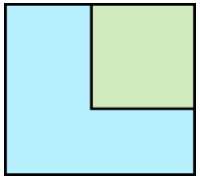
Pentium III





Pentium IV

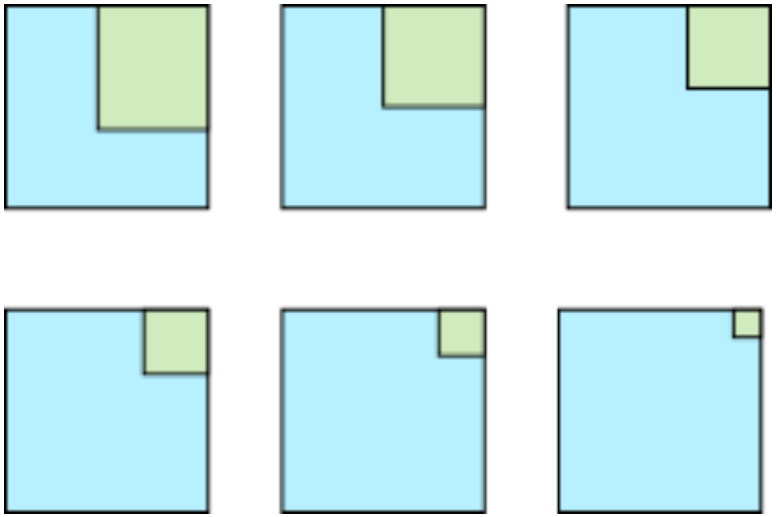




Q: What can you observe? Why?

Extrapolation of Single Core CPUs

If we would have extrapolated the trend, in a few generations, Processors would have looked like this:

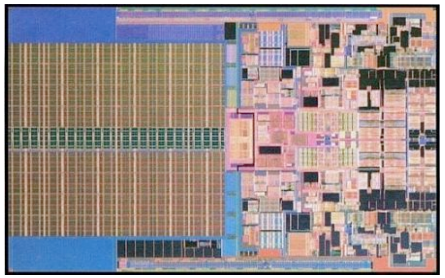


Of course, we know it did not happen.

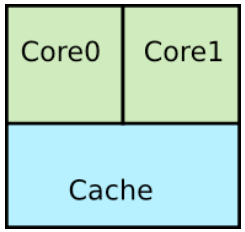
Q: What happened instead? Why?

Multi-Core CPUs Evolution

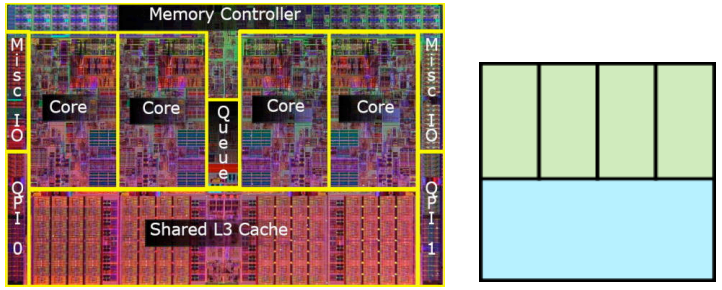
Penny



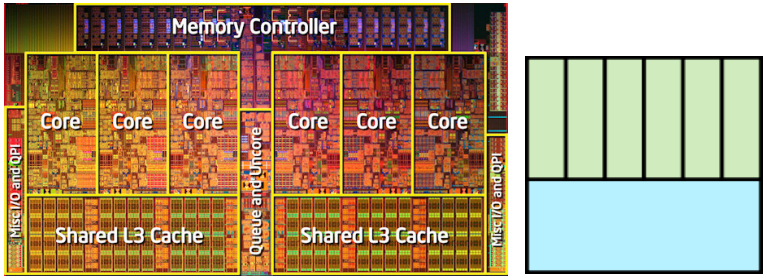
Chip area breakdown



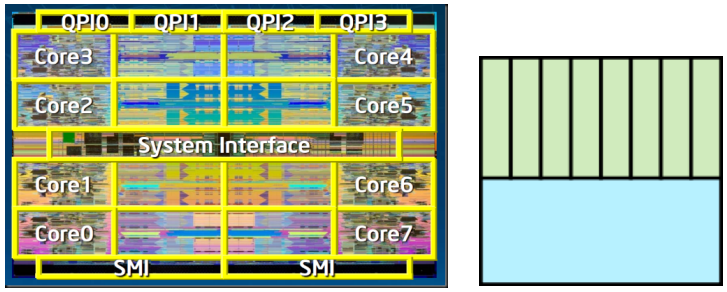
Bloomfield



Gulftown

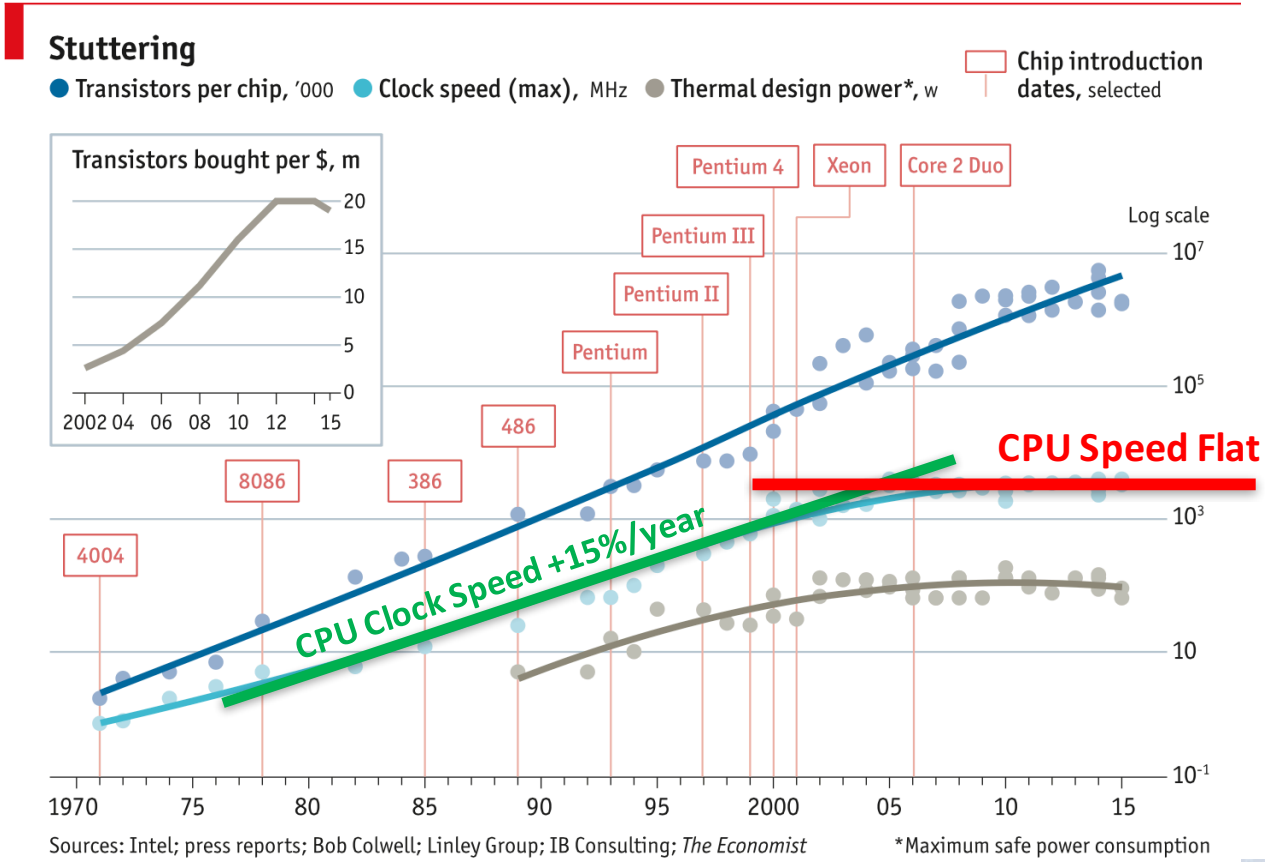


Beckton

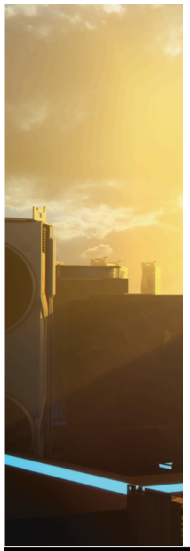


Q: What can you observe? Why?

Why is Architecture Exciting Today?



turing lecture



DOI:10.1145/3282307

Innovations like domain-specific hardware, enhanced security, open instruction sets, and agile chip development will lead the way.

BY JOHN L. HENNESSY AND DAVID A. PATTERSON

A New Golden Age for Computer Architecture

WE BEGAN OUR Turing Lecture June 4, 2018[11] with a review of computer architecture since the 1960s. In addition to that review, here, we highlight current challenges and identify future opportunities, projecting another golden age for the field of computer architecture in the next decade, much like the 1980s when we did the research that led to our award, delivering gains in cost, energy, and security, as well as performance.

"Those who cannot remember the past are condemned to repeat it." —George Santayana, 1905

Software talks to hardware through a vocabulary called an instruction set architecture (ISA). By the early 1960s, IBM had four incompatible lines of computers, each with its own ISA, software stack, I/O system, and market niche—targeting small business, large business, scientific, and real time, respectively. IBM

ACM Communications, February 2019

Important Trends

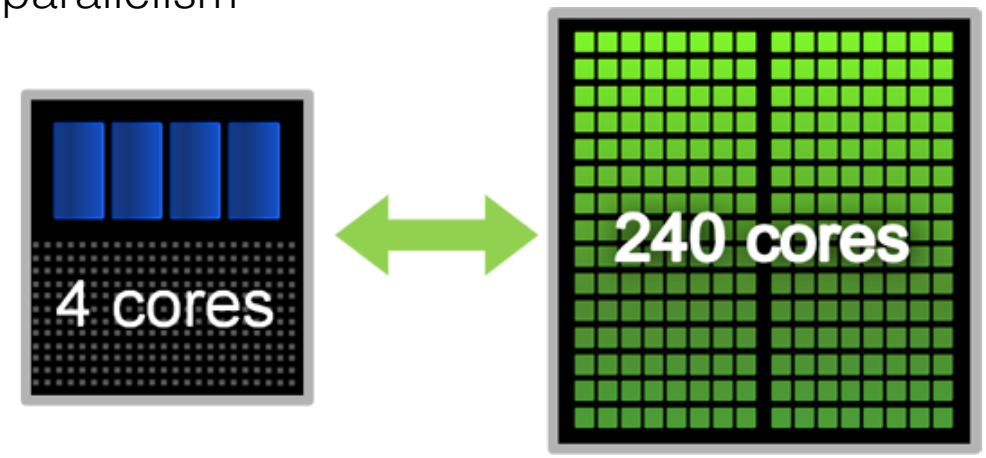
- Running out of ideas to improve single thread performance
- Power wall makes it harder to add complex features
- Power wall makes it harder to increase frequency
- Historical contributions to performance:
  - Better processes (faster devices) ~**20%** (eventually disappearing)
  - Better circuits/pipelines ~**15%** (trending lower)
  - Better organization/architecture ~**15%**

Practical Example

- +**300x** speedup for matrix vector multiplication
  - Data level parallelism: **3.8x**
  - Loop unrolling and out-of-order execution: **2.3x**
  - Cache blocking: **2.5x**
  - Thread level parallelism: **14x**
- Further, one can use accelerators to get an additional **100x**

CPUs & GPUs

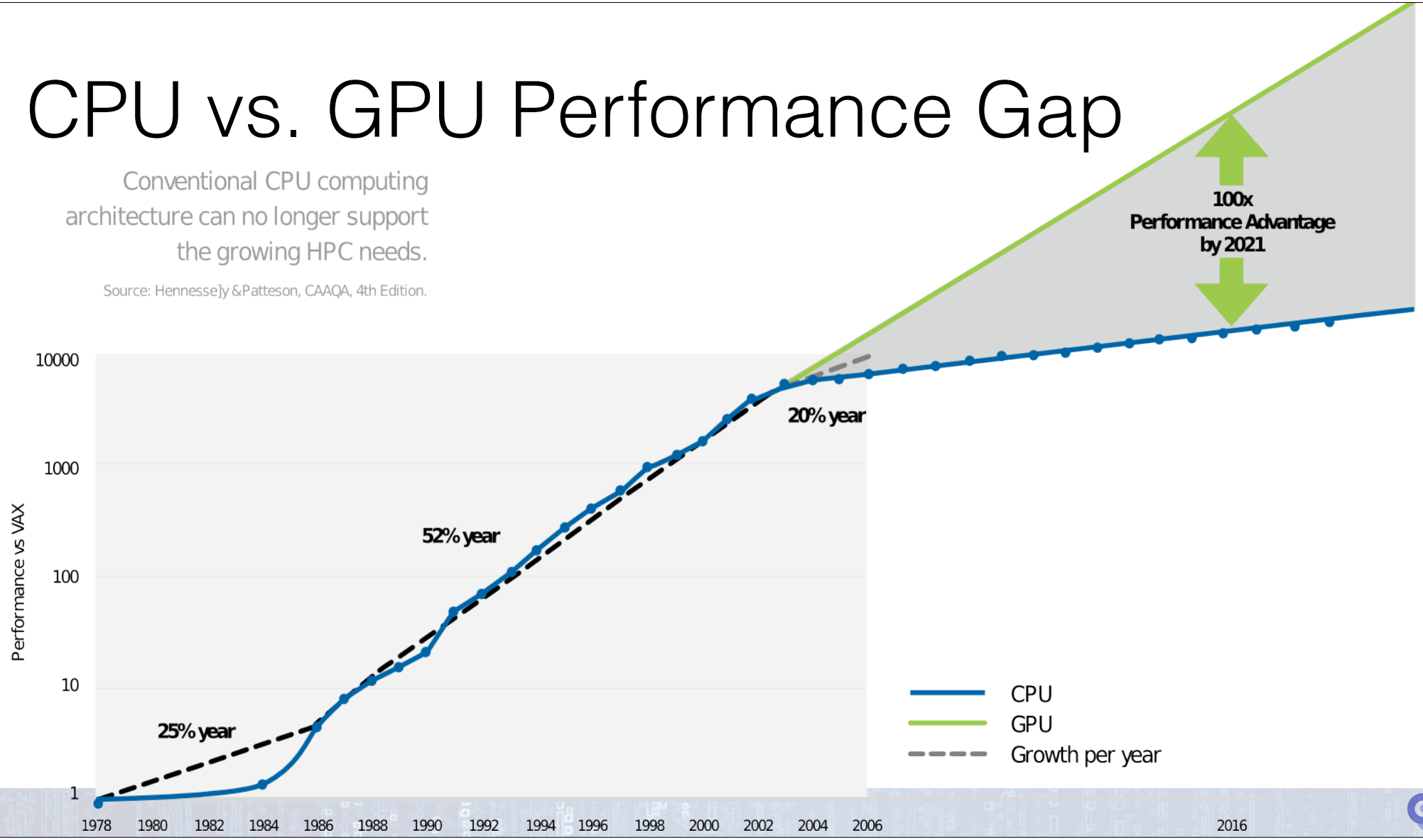
- Typically GPUs and CPUs coexist in a heterogeneous setting
- “Less” computationally intensive part runs on CPU
  - Coarse-grained parallelism
- More intensive parts run on GPU
  - Fine-grained parallelism



CPU vs. GPU Performance Gap

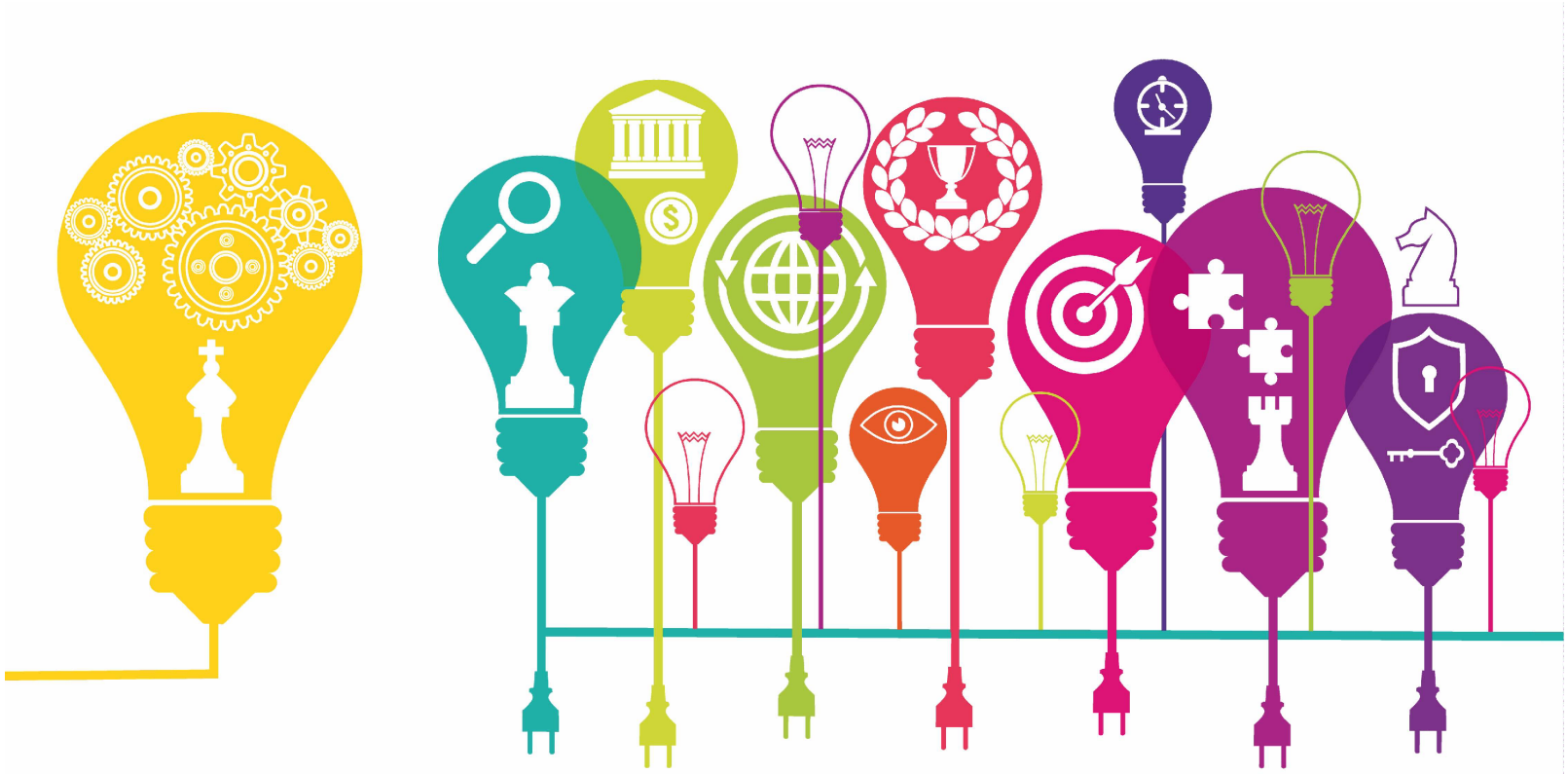
Conventional CPU computing architecture can no longer support the growing HPC needs.

Source: Hennessey &Patteson, CAAQA, 4th Edition.



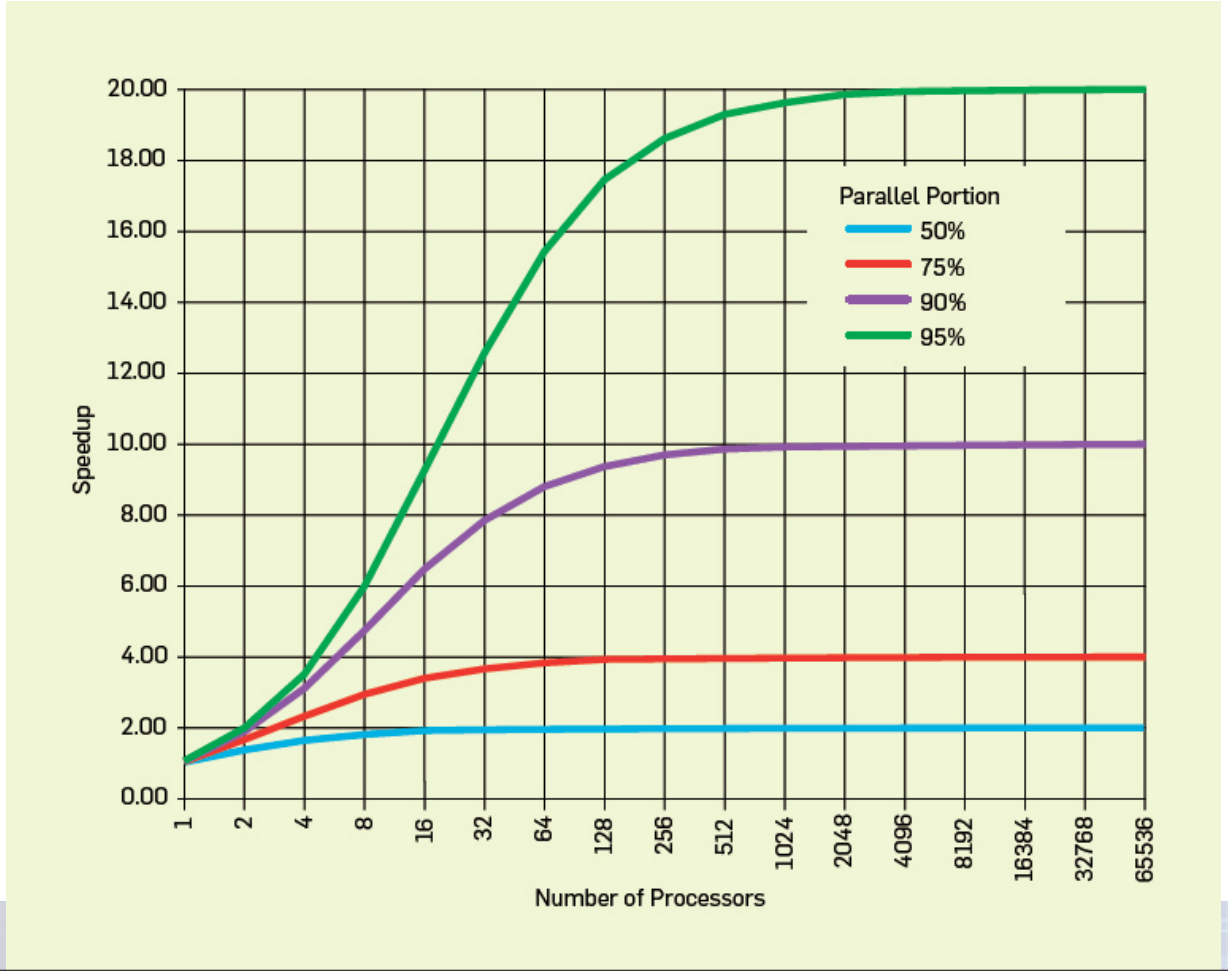
Throughput Oriented Architectures

- Fine-grained interleaved threading (~2x comp density)
- SIMD/SIMT (>10x comp density)
- Simple core (~2x comp density)



The Software Challenge

Amdahl's Law



"Keep Them Honest"

Opportunities

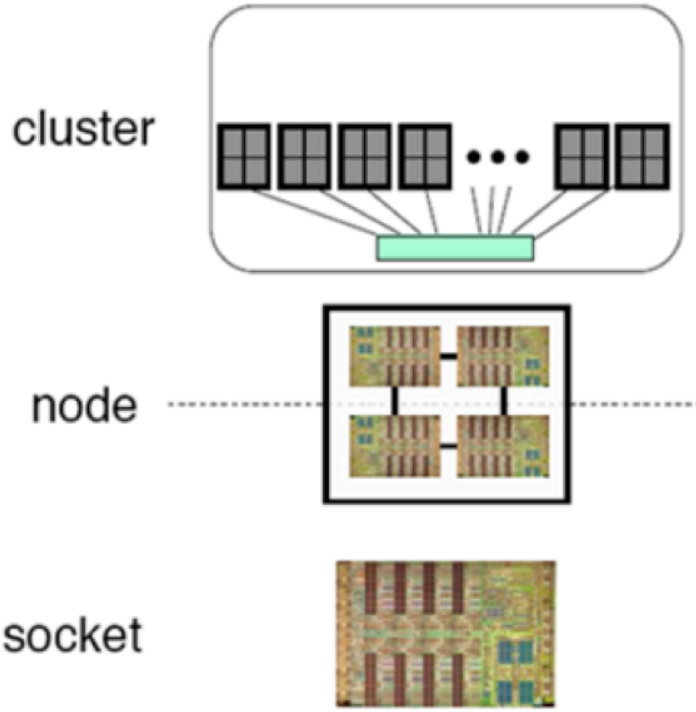
- Computer architecture today
  - General purpose: optimized for control
  - Accelerators: optimized for speed
  - Few players: Intel, AMD, ARM, NVidia…
- Future directions?
  - Applications have different requirements
    - Battery life / energy
    - Security & privacy
- Best solutions depend on application
  - Many players: HW & SW
  - **Innovation matters more than Moore's Law**

What Does This Mean to a Programmer?

- Today, one can expect at most a 20% annual improvement; even lower if the program is not multi-threaded
  - A program needs many threads
  - The threads need efficient synchronization and communication
  - Data placement in the memory hierarchy is important
  - Accelerators should be used when possible

A Likely Future Scenario

System: cluster + many core node



Programming model: MPI+?

Message Passing

Not Message Passing

Hybrid & many core technologies will require new approaches: PGAS, auto tuning, ?

Why MPI (still) Persists

- MPI did not (and will not) disappear the foreseeable future
- Today we have more than 25 years of legacy software in MPI
- New systems are not sufficiently different to lead to a radical new (distributed) programming model

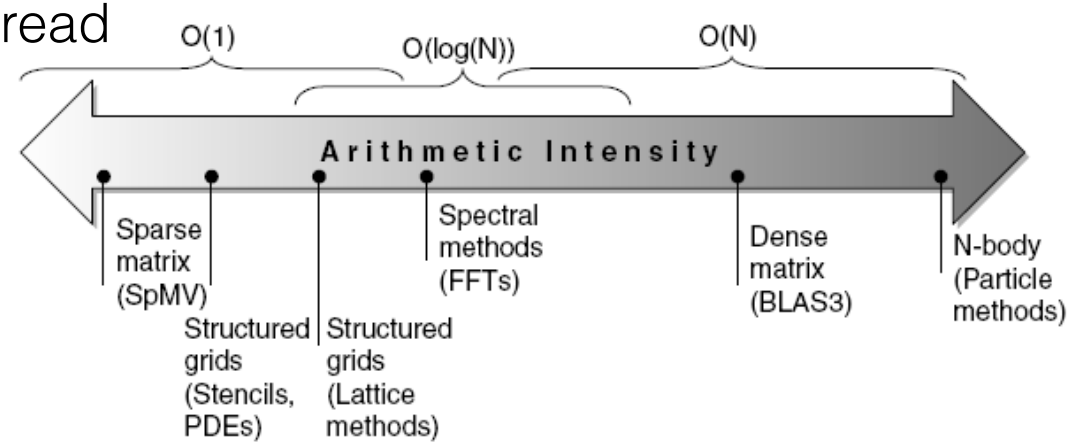
What will be the “?” in MPI+?

- Likely candidates include
  - PGAS languages
  - Autotuning
  - CUDA, OpenCL, OpenACC
  - A wildcard from the commercial space
  - Domain-specific paradigms

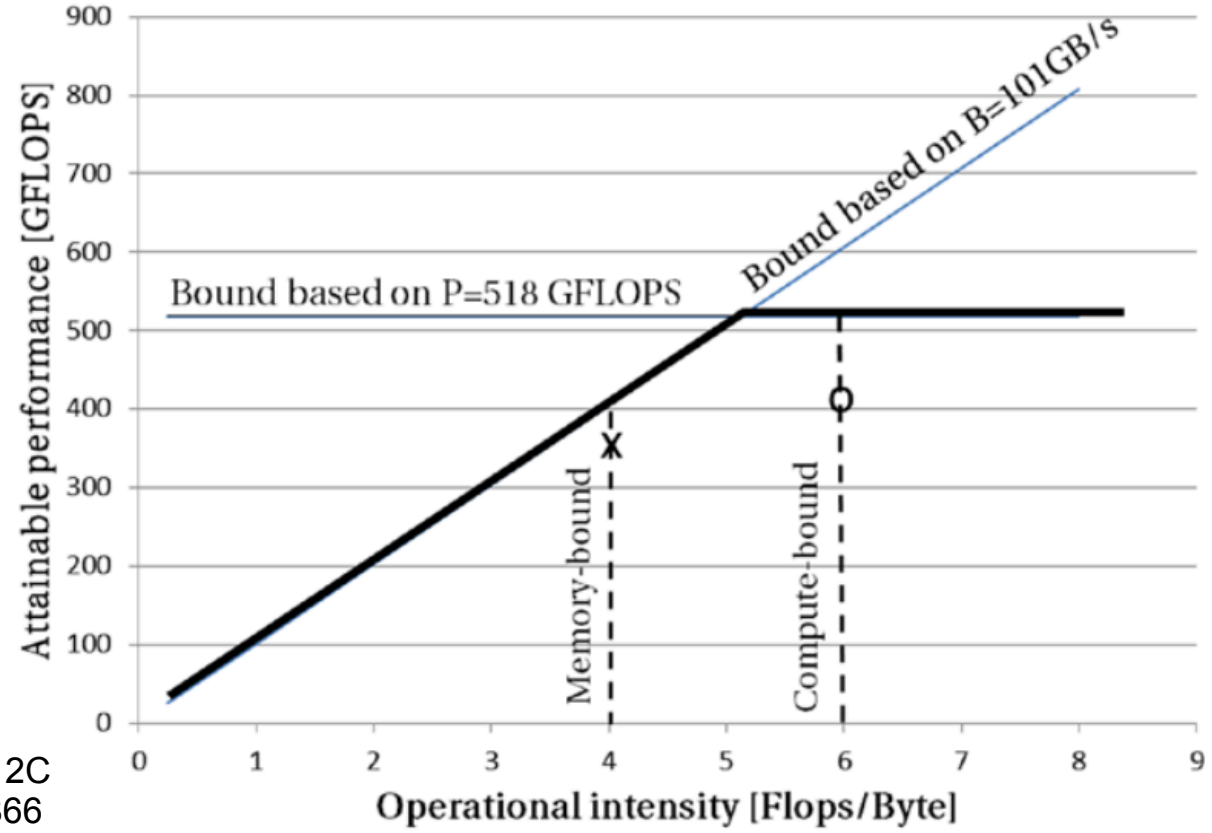
The Roofline Model

The Roofline Performance Model

- Basic Idea
  - Plot peak floating-point throughput as a function of arithmetic intensity
  - Ties together floating-point performance and memory performance for a target machine
- Arithmetic Intensity
  - Floating-point operations per byte read



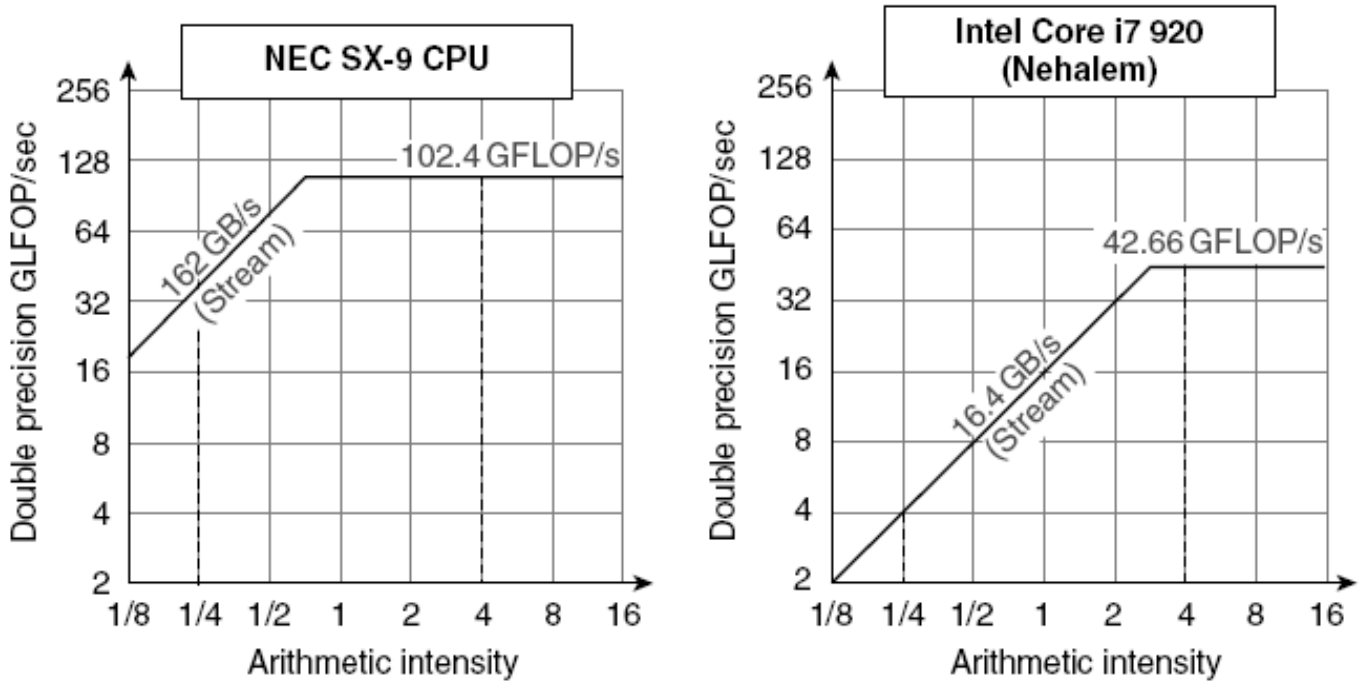
Roofline Performance Examples



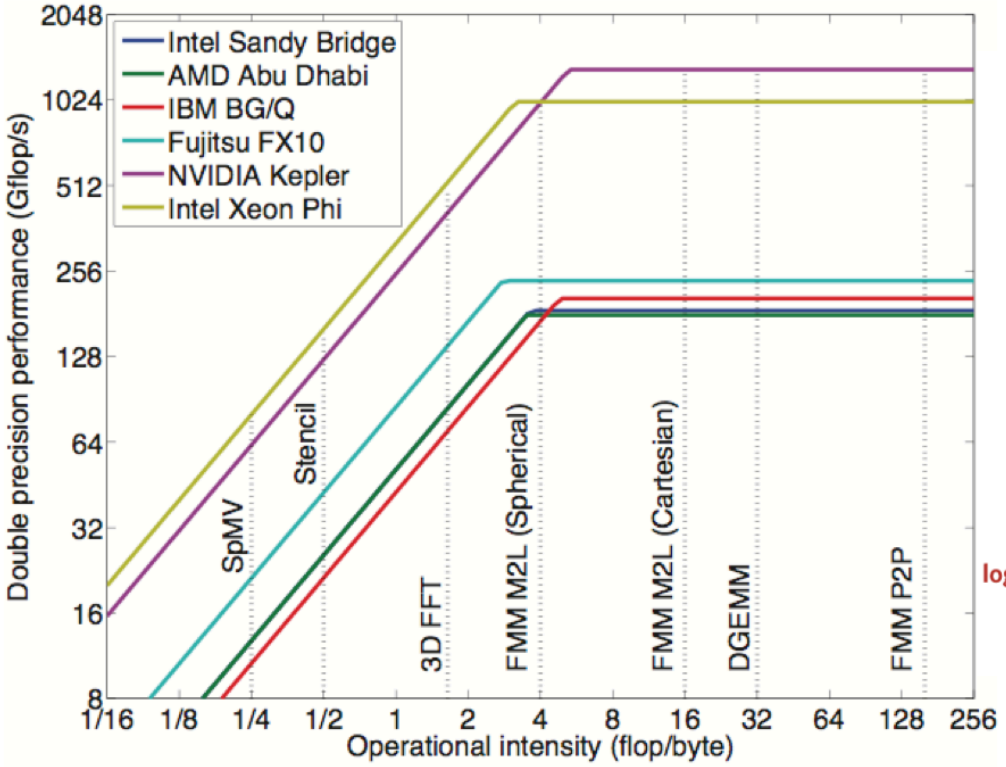
Intel XeonE5-2697v2 12C
2.7GHz with DDR3-1866

Roofline Performance Examples

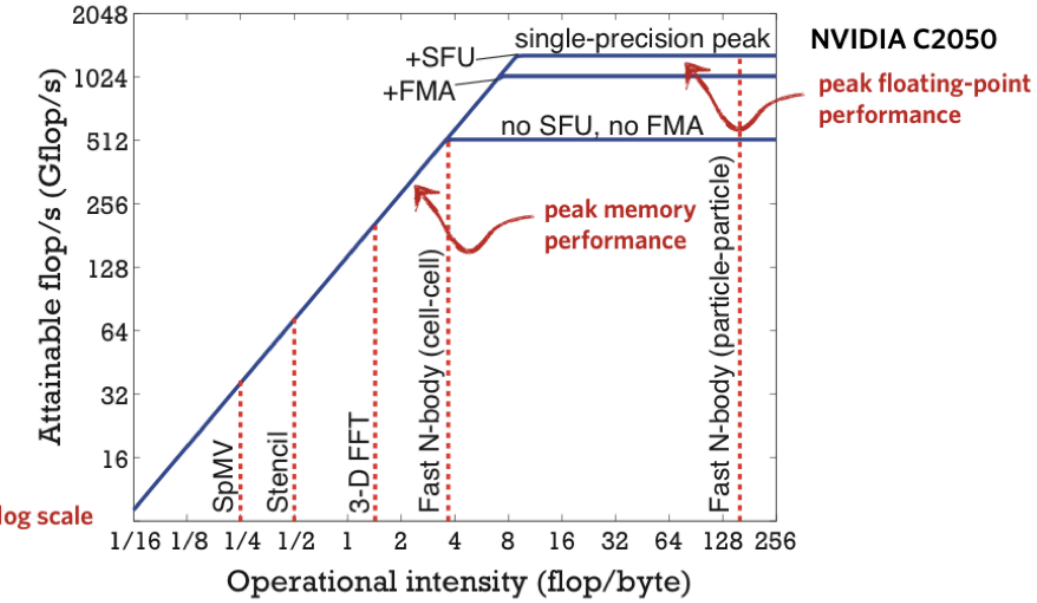
- Attainable GFLOPs/sec Min = (Peak Memory BW × Arithmetic Intensity, Peak Floating Point Performance)



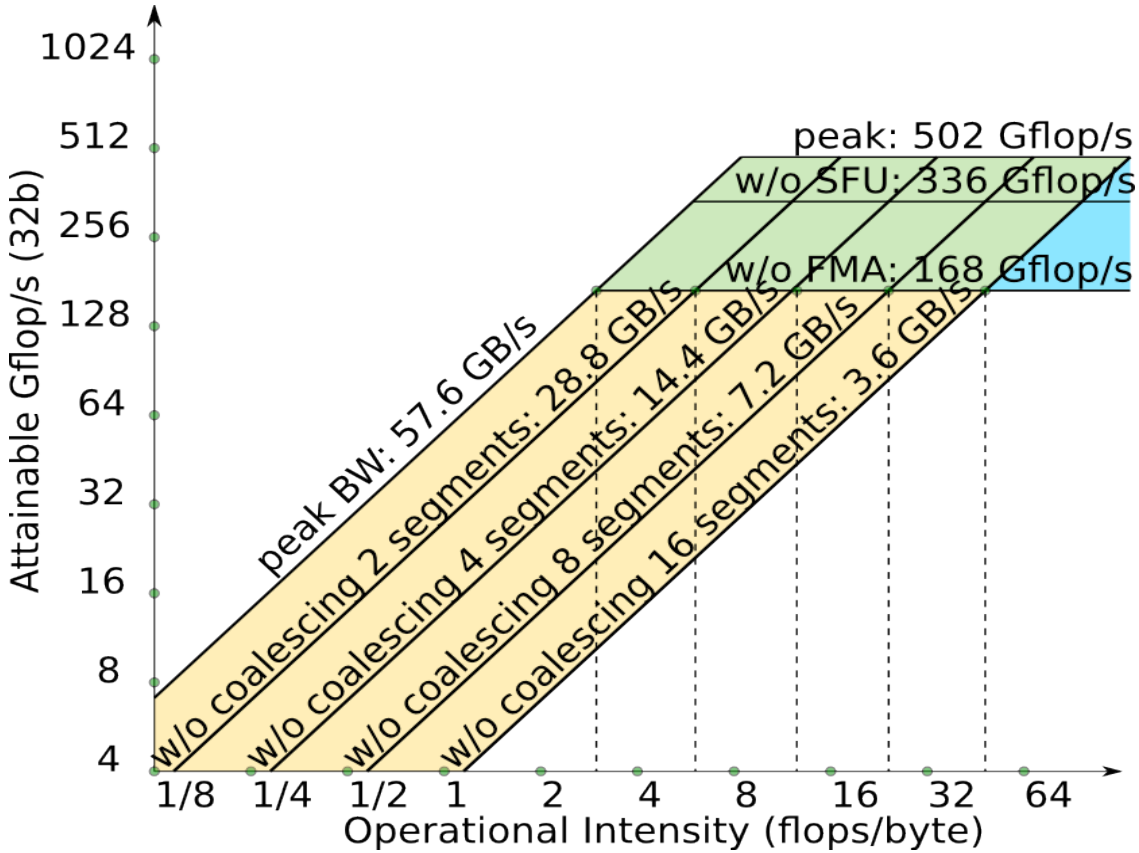
Roofline Performance Examples



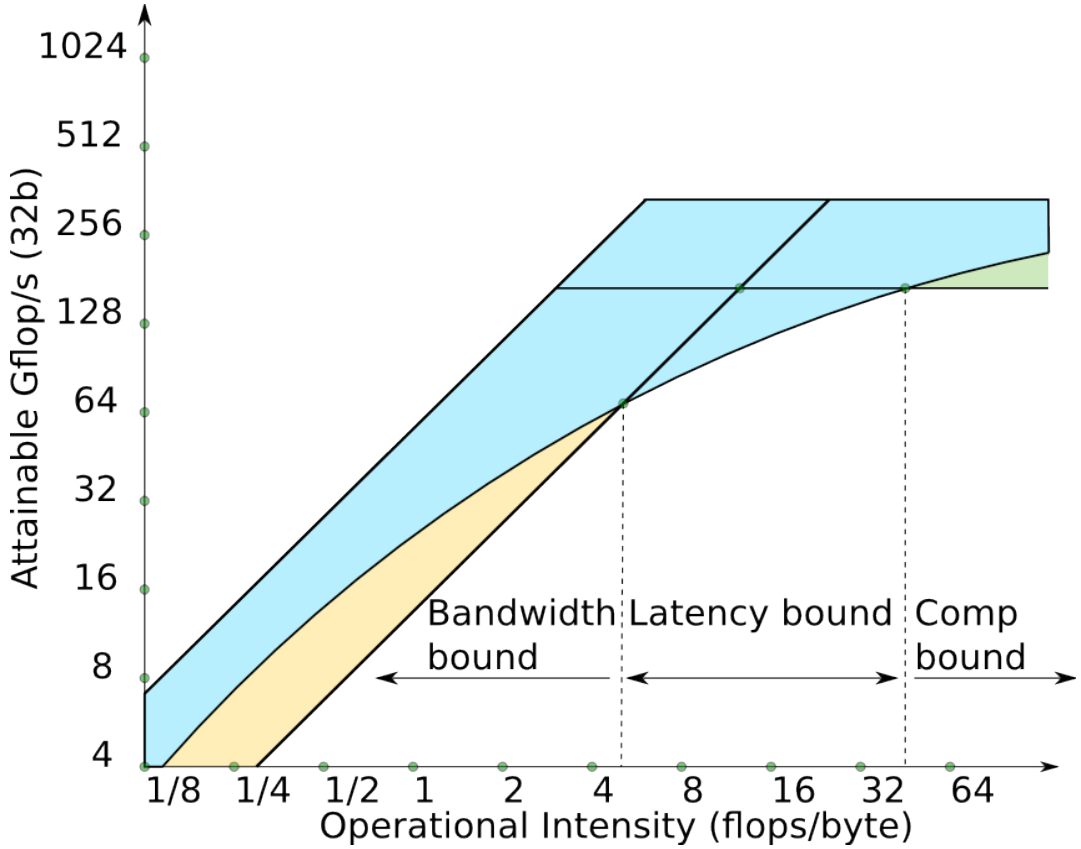
log/log scale



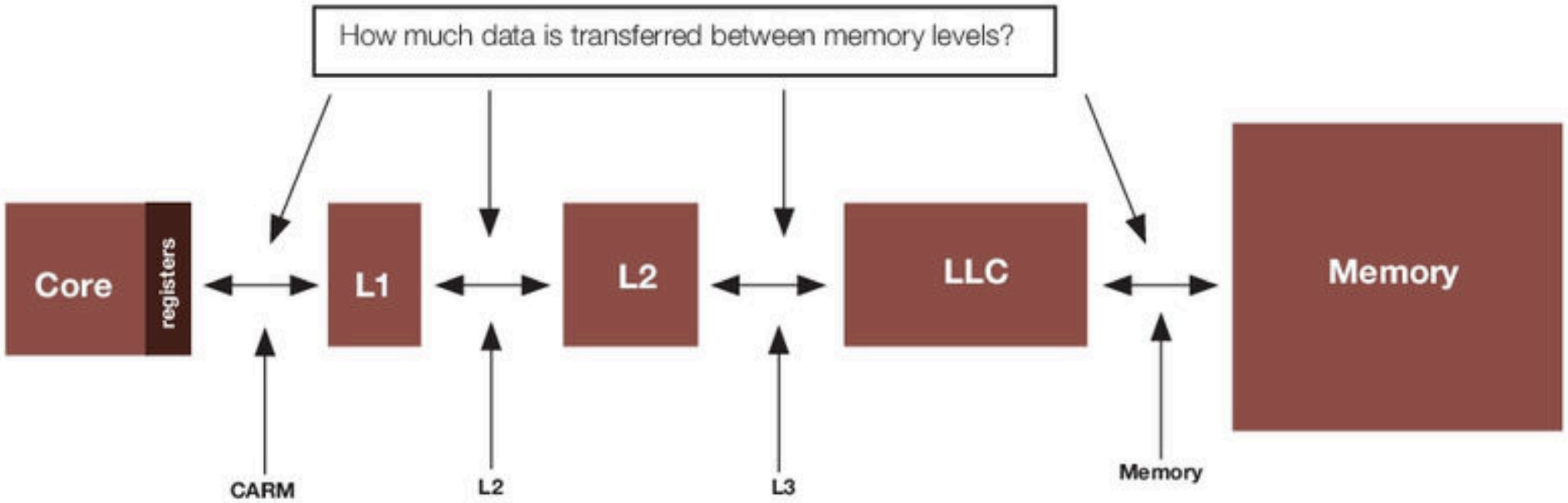
Optimization vs. Roofline Model



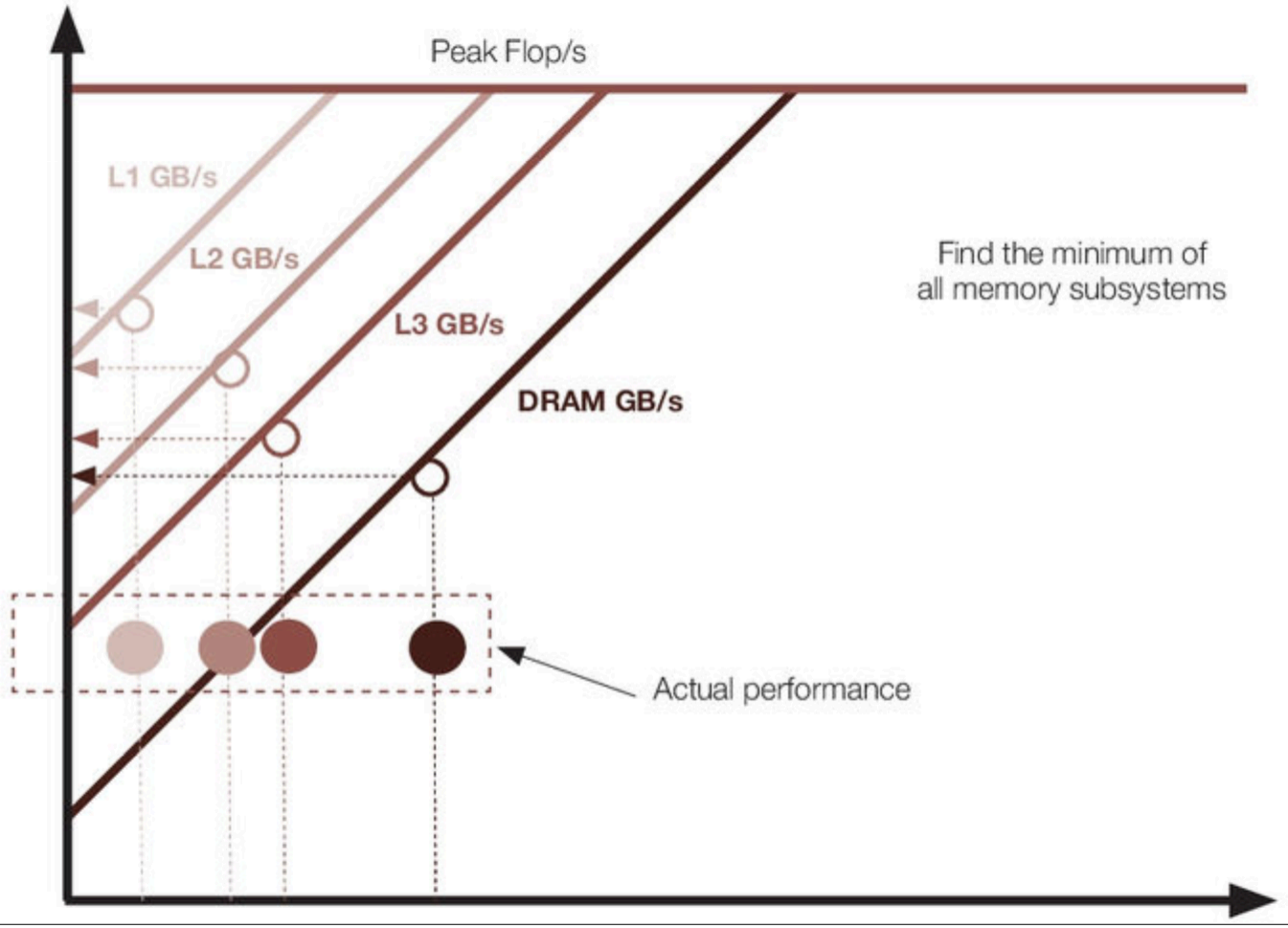
Bottlenecks & Limitations



Cache-Aware Roofline Model

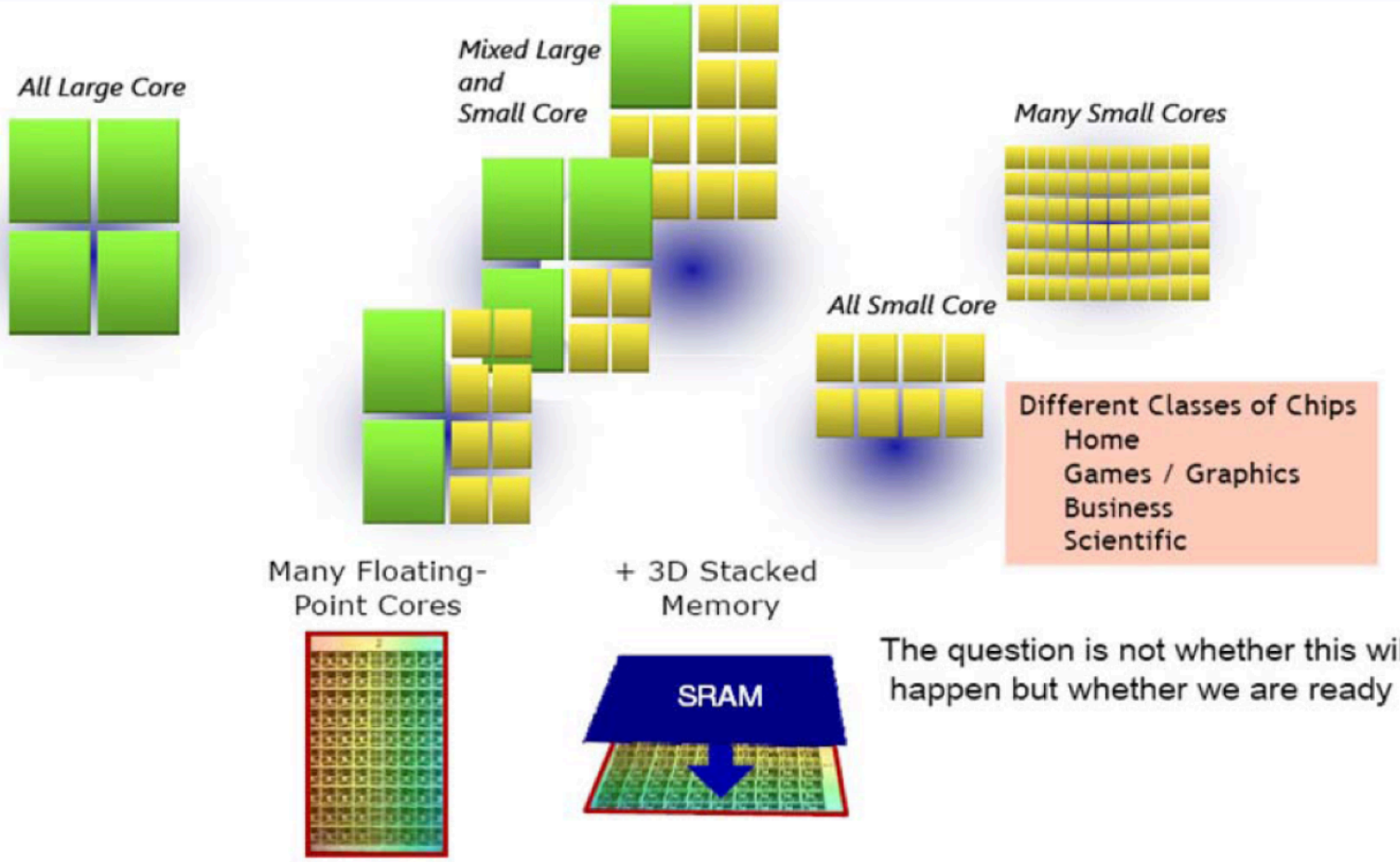


Performance Limitations

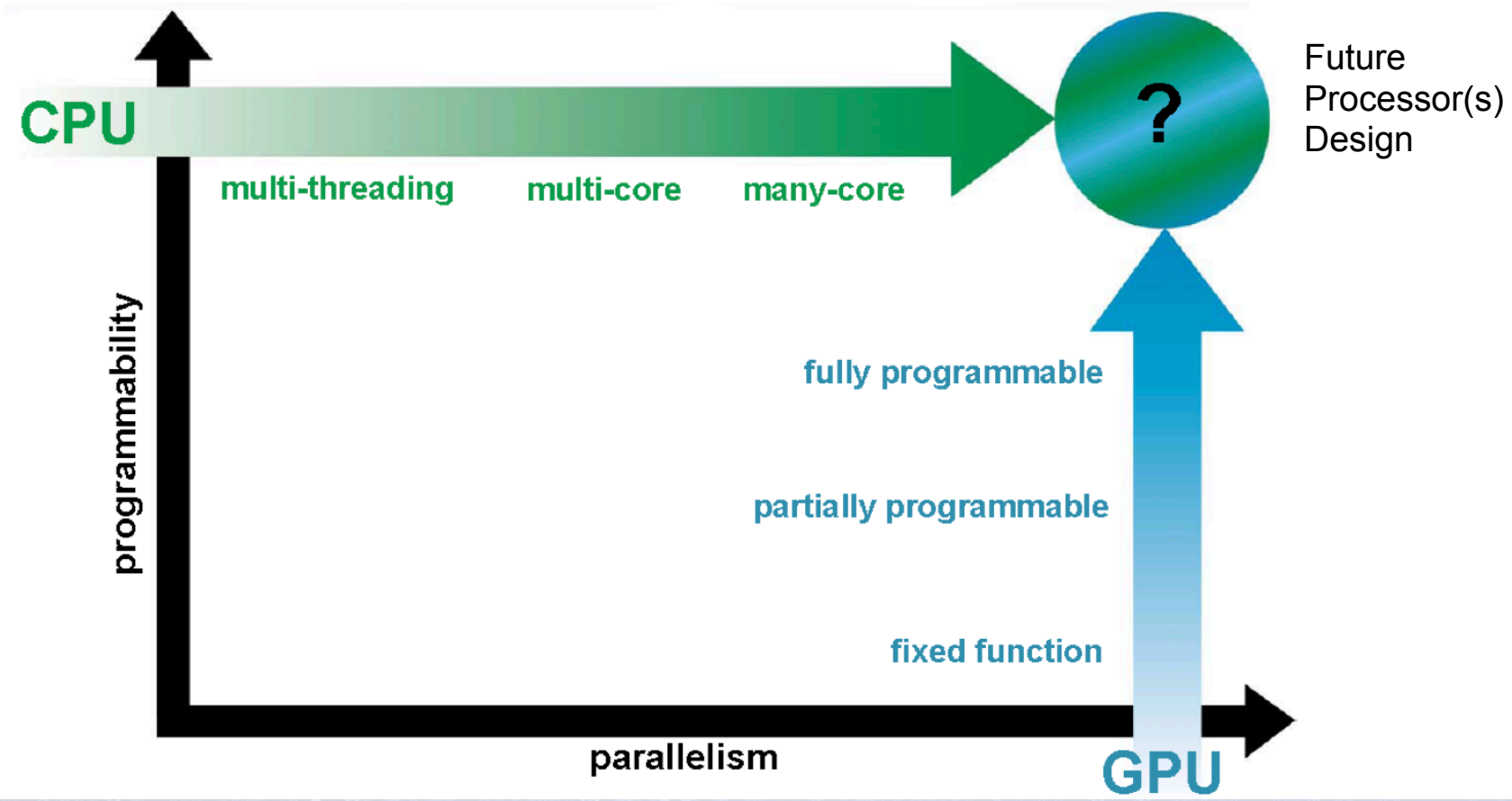


Going Forward

What's next?


All Large Core
Mixed Large and Small Core
Many Small Cores
All Small Core
Different Classes of Chips
Home
Games / Graphics
Business
Scientific
Many Floating-Point Cores
+ 3D Stacked Memory
SRAM
The question is not whether this will happen but whether we are ready

A Likely Trajectory: Collision or Convergence?

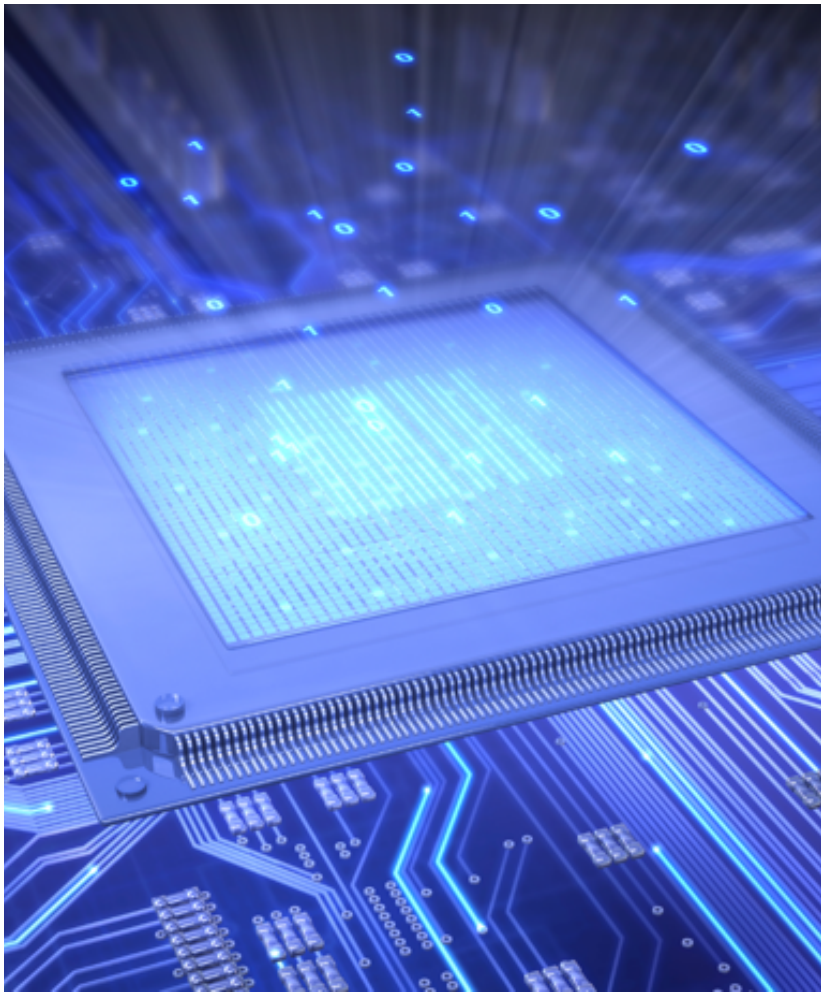


Scientific Computing Community Objective

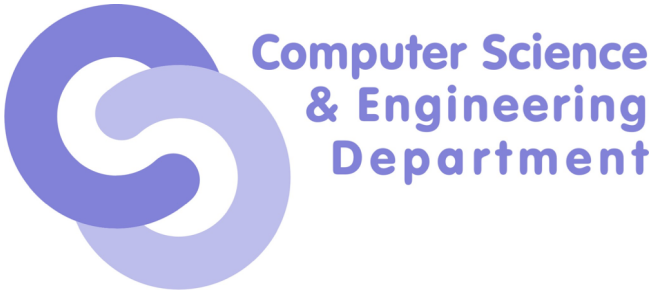
- Building up the ability to translate parallel computing power into science and engineering breakthroughs
  - Identify apps whose computing structures are suitable for SC
  - These applications can be revolutionised by 100X computing power
  - Provide access to expertise needed to tackle these apps
- Develop SW solutions offering better HW efficiency & utilisation

Slides & Support Material Acknowledgements

David Patterson: Computer Architecture is Back 2007
Dongarra & Rattner - ISC 2008
Sathish Vadhiyar - 2009
Ye Mesman Corporaal - 2010
Michael Garland et. al, "Understanding throughput-oriented architectures", CACM 2010
Dan Negrut & Dan Melanz & Andrew Seidl - 2013
David Kirk - NVidia 2009/2014
CS61C - University of Berkeley 2016
CS/ECE 3810 - University of Colorado 2017
Cache-Aware Roofline Model - 2018



Thank you for your attention



emil.slusanschi@cs.pub.ro
http://csc.web.cern.ch