Computational Science
The Third Pillar of Science?

Emil Slușanschi
CERN School of Computing 2019
Cluj-Napoca, Romania
Outline

• Scientific Research Paradigm
• The Evolution of Computing
• The Software Challenge
• The Roofline Model
• Going forward
Scientific Research Paradigm
The Third Pillar of Science

Experimental science

Theoretical science

Computational science

Suggest theory
Test theory

Generate data
Model real process
Suggest experiments
Analyze data
Control apparatus

Provide equations
Interpret results

Suggest experiments
Interpret theory
Perform accurate calculations
Perform large-scale calculations
Future Science and Engineering Breakthroughs Hinge on Computing

- Computational Geoscience
- Computational Medicine
- Computational Physics
- Computational Finance
- Computational Chemistry
- Computational Modeling
- Computational Biology
- 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

20\textsuperscript{th} Century

- Small mask patterns and short light waves
- Electronic microscope and Crystallography with computational image processing
- Anatomic imaging with computational image processing
- Teleconference

21\textsuperscript{st} Century

- Computational optical proximity correction
- Computational microscope with initial conditions from Crystallography
- Metabolic imaging sees disease before visible anatomic change
- 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”
  - 1\textsuperscript{st} principle-based simulation at meaningful scale
  - New, 3\textsuperscript{rd} 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 \textit{hundreds})

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

Hundred-core systems with MatLab-style programming, 10-100 GFLOPS, \$K, machines widely available (programmers numbered in \textit{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”

• “640KB [of main memory] ought to be enough for anybody.”
  - Bill Gates, Chairman of Microsoft / IBM System Limitation, 1981.
Gordon Moore (Intel cofounder) predicted in 1965 that transistor density in semiconductors will **double each 18 months**

Moore’s law is confirmed. Until now…

What about going forward?
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

Transistor Pathway

Si/Ge Gate All Around (GAA) Vertical or Horizontal
- Improved electrostatics
- Precision etch and CMP
- Sealed metals
- High Aspect Ratio ALD

Bi-V FinFET
- Improved mobility
- Epi structure
- III-V gate interface
- New material CMP

Vertical
TFET
- Improved SS
- Epi structure
- Multi-pass CMP
- Precision etch & CMP

Currently in production

FinFET

Fully depleted silicon-on-insulator

Lateral nanowire 2019

Vertical nanowire 2021

Monolithic 3D 2024
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 $c = 3 \times 10^8$ 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 $r < c / 10^{13} \approx 3e^{-6}$ 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 $10$GB/cm$^3$
  - Factor $10^{20}$ below what is required!

- Bottom line: we can’t do this with current technology
Pentium Processors Evolution

Q: What can you observe? Why?

Pentium I

Pentium II

Pentium III

Pentium IV

Chip area breakdown
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

Core0  Core1

Cache

Bloomfield

Gulftown

Beckton

Q: What can you observe? Why?
Why is Architecture Exciting Today?

Stuttering

- Transistors per chip, '000
- Clock speed (max), MHz
- Thermal design power*, W

CPU Speed Flat

**CPU Clock Speed +15%/year**

Log scale

Sources: Intel; press reports; Bob Colwell; Linley Group; IHS Consulting; The Economist

*Maximum safe power consumption

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) \(\sim 20\%\) (eventually disappearing)
  - Better circuits/pipelines \(\sim 15\%\) (trending lower)
  - Better organization/architecture \(\sim 15\%\)
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.

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
Roofline Performance Examples

- Attainable GFLOPs/sec Min = (Peak Memory BW × Arithmetic Intensity, Peak Floating Point Performance)
Roofline Performance Examples
Optimization vs. Roofline Model

- Peak BW: 57.6 GB/s
- W/o SFU: 336 Gflop/s
- W/o FMA: 168 Gflop/s

Attainable Gflop/s (32b)

Operational Intensity (flops/byte)
Bottlenecks & Limitations
Cache-Aware Roofline Model
Performance Limitations

- Peak Flop/s
- L1 GB/s
- L2 GB/s
- L3 GB/s
- DRAM GB/s
- Find the minimum of all memory subsystems
- Actual performance

Find the minimum of all memory subsystems.
Going Forward
What’s next?

- All Large Core
- Mixed Large and Small Core
- Many Small Cores
- Many Floating-Point Cores

+ 3D Stacked Memory

Different Classes of Chips
- Home
- Games / Graphics
- Business
- Scientific

The question is not whether this will happen but whether we are ready.
A Likely Trajectory: Collision or Convergence?

Future Processor(s) Design

CPU

- multi-threading
- multi-core
- many-core

parallelism

programmability

GPU

- fixed function
- partially programmable
- fully programmable
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