BLUE WATERS

SUSTAINED PETASCALE COMPUTING

Parallel Computing 2020: Preparing for the Post-Moore Era

Marc Snir


GREAT LAKES CONSORTIUM
FOR PETASCALE COMPUTATION

THE (CMOS) WORLD IS ENDING NEXT DECADE

So says the International Technology Roadmap for Semiconductors (ITRS)

End of CMOS?

IN THE LONG TERM (~2017 THROUGH 2024)

While power consumption is an urgent challenge, its leakage or static component will become a major industry crisis in the long term, threatening the survival of CMOS technology itself, just as bipolar technology was threatened and eventually disposed of decades ago. [ITRS 2009/2010]

- Unlike the situation at the end of the bipolar era, no technology is waiting in the wings.

Technology Barriers

- New materials
  - *.. such as III-V or germanium thin channels on silicon, or even semiconductor nanowires, carbon nanotubes, graphene or others may be needed.*
- New structures
  - *three-dimensional architecture, such as vertically stackable cell arrays in monolithic integration, with acceptable yield and performance*.
- *…These are huge industry challenges to simply imagine and define*

- **Note**: Predicted feature size in 2024 (7.5 nm) = ~32 silicon atoms (Si-Si lattice distance is 0.235 nm)

Economic Barriers

- ROI challenges
  - *… achieving constant/improved ratio of … cost to throughput might be an insoluble dilemma.*
- Rock's Law: Cost of semiconductor chip fabrication plant doubles every four years
  - Current cost is $7-$9B
  - Intel's yearly revenue is $35B
  - Semiconductor industry grows < 20% annually
  - Opportunities for consolidation are limited
- Will take longer to amortize future technology investments
  - *Progress stops when manufacturing a twice as dense chip is twice as expensive*

Scaling is Plateauing

- Simple scaling (proportional decrease in all parameters) has ended years ago
- Single thread performance is not improving
- Rate of increase in chip density is slowing down in the next few years, for technological and economic reasons
- *Silicon will plateau at x10-x100 current performance*
  - No alternative technology is ready for prime time

IT as Scaling Slows

- End of Moore's Law is not the end of the Computer Industry
  - It needs not be the end of IT growth

- Mass market (mobile, home): Increasing emphasis on function (or fashion)
- Big iron: Increasing emphasis on *compute efficiency*: Get more results from a given energy and transistor budget.

Compute Efficiency

- Progressively more efficient use of a fixed set of resources (similar to fuel efficiency)
  - More computations per joule
  - More computations per transistor
- A clear understanding of where performance is wasted and continuous progress to reduce "waste"
  - A clear distinction between "overheads" – *computational friction* -- and the essential work

(We are still very far from any fundamental limit)

HPC – The Canary in the Mine

- HPC is already heavily constrained by low compute efficiency
  - High power consumption, high levels of parallelism
- *Exascale research is not only research for the next turn of the crank in supercomputing, but research on how to sustain performance growth in face of semiconductor technology slow-down*
  - Essential for continued progress in science, national competitiveness and national security

PETASCALE IN A YEAR

Blue Waters

Blue Waters

System Attribute	Blue Waters
Vendor	IBM
Processor	IBM Power7
Peak Performance (PF)	~10
Sustained Performance (PF)	~1
Number of Cores/Chip	8
Number of Cores	>300,000
Amount of Memory (PB)	~1
Amount of Disk Storage (PB)	~18
Amount of Archival Storage (PB)	>500
External Bandwidth (Gbps)	100-400
Water Cooled	>10 MW

Exascale in 2015 with 20MW [Kogge's Report]

- "Aggressive scaling of Blue Gene Technology" (32nm)
  - 67 MW
  - 223K nodes, 160M cores
  - 3.6 PB memory (1 Byte/1000 flops capacity, 1 Word/50 flops bandwidth)
  - 40 mins MTTI
- *A more detailed and realistic study by Kogge indicates power consumption is ~500 MW*

Kogge -- Spectrum

- *"[A] practical exaflops-scale supercomputer … might not be possible anytime in the foreseeable future"*
- *"Building exascale computers ... would require engineers to rethink entirely how they construct number crunchers…"*
- *"Don't expect to see an [exascale] supercomputer any time soon. But don't give up hope, either."*

Exascale Research: Some Fundamental Questions

- Power Complexity
- Communication-optimal computations
- Low entropy computations
- Jitter-resilient computation
- Steady-state computations
- Friction-less architecture
- Self-organizing computations
- Resiliency

Power Complexity

- There is a huge gap between theories on the (quantum) physical constraints of computation and the practice of current computing devices
- Can we develop power complexity models of computations that are relevant to computer engineers?

Communication-Efficient Algorithms: Theory

- Communication in time (registers, memory) and space (buses, links) is, by far, the major source of energy consumption
- *Need to stop counting operations and start counting communications*
- Need a theory of communication-efficient algorithms (beyond FFT and dense linear algebra)
  - Communication-efficient PDE solvers (understand relation between properties of PDE and communication needs)
- Need to measure correctly inherent communication costs at the algorithm level
  - Temporal/spatial/processor locality: second order statistics on data & control dependencies

Communication-Efficient Computations: Practice

- Need better benchmarks to sample multivariate distributions (apply Optimal Sampling Theory?)
- Need communication-focused programming models & environments
  - User can analyze and control cost of communications incurred during program execution (volume, locality)
- Need productivity environments for performance-oriented programmers

Low-Entropy Communication

- Communication can be much cheaper if "known in advance"
  - Memory access overheads, latency hiding, reduced arbitration cost, bulk transfers (e.g., optical switches)
  - … Bulk mail vs. express mail
- Current HW/SW architectures take little advantage of such knowledge
  - Need architecture/software/algorithm research
- CS is lacking a good algorithmic theory of entropy
  - Need theory, benchmarks, metrics

Jitter-Resilient Computation

- Expect increased variance in the compute speed of different components in a large machine
  - Power management
  - Error correction
  - Asynchronous system activities
  - Variance in application
- Need variance-tolerant applications
  - Bad: frequent barriers, frequent reductions
  - Good: 2-phase collectives, double-buffering
- Need theory and metrics
- Need new variance-tolerant algorithms
- Need automatic transformations for increased variance tolerance

Steady-State Computation

- Each subsystem of a large system (CPU, memory, interconnect, disk) has low average utilization during a long computation
- Each subsystem is the performance bottleneck during part of the computation
- *Utilization is not steady-state – hence need to over-provision each subsystem.*
- Proposed solution A: power management, to reduce subsystem consumption when not on critical path
  - Hard (in theory and in practice)
- Proposed solution B: Techniques for steady-state computation
  - E.g., communication/computation overlap
- Need research in Software (programming models, compilers, run-time), and architecture

Friction-less Software Layering

- Current HW/SW architectures have developed multiple, rigid levels of abstraction (ISA, VM, APIs, languages…)
  - Facilitates SW development but energy is lost at layer matching
- Flexible specialization enables to regain lost performance
  - Inlining, on-line compilation, code morphing
  - Similar techniques are needed for OS layers

Self-Organizing Computations

- Hardware continuously changes (failures, power management)
- Algorithms have more dynamic behavior (multigrid, multiscale – adapt to evolution of simulated system)
- *Mapping of computation to HW needs to be continuously adjusted*
- Too hard to do in a centralized manner -> Need distributed, hill climbing algorithms

Resiliency

- HW for fault correction (and possibly fault detection) may be too expensive (consumes too much power)
  - and is source of jitter
- Current global checkpoint/restart algorithms cannot cope with MTBF of few hours or less
- Need SW (language, compiler, runtime) support for error compartmentalization
  - Catch error before it propagates
- May need fault-tolerant algorithms
  - Need new complexity theory

Summary

- The end of Moore's era will change in fundamental ways the IT industry and CS research
  - A much stronger emphasis on compute efficiency
  - A more systematic and rigorous study of sources of inefficiencies
- The quest for exascale at reasonable power budget is the first move into this new domain