22 novembre 2010

BLUE WATERS

SUSTAINED PETASCALE COMPUTING

Exascale Computing: The Last Rehearsal Before the Post-Moore Era

Marc Snir


GREAT LAKES CONSORTIUM
FOR PETASCALE COMPUTATION

THE WORLD IS ENDING

THE (CMOS) WORLD IS ENDING

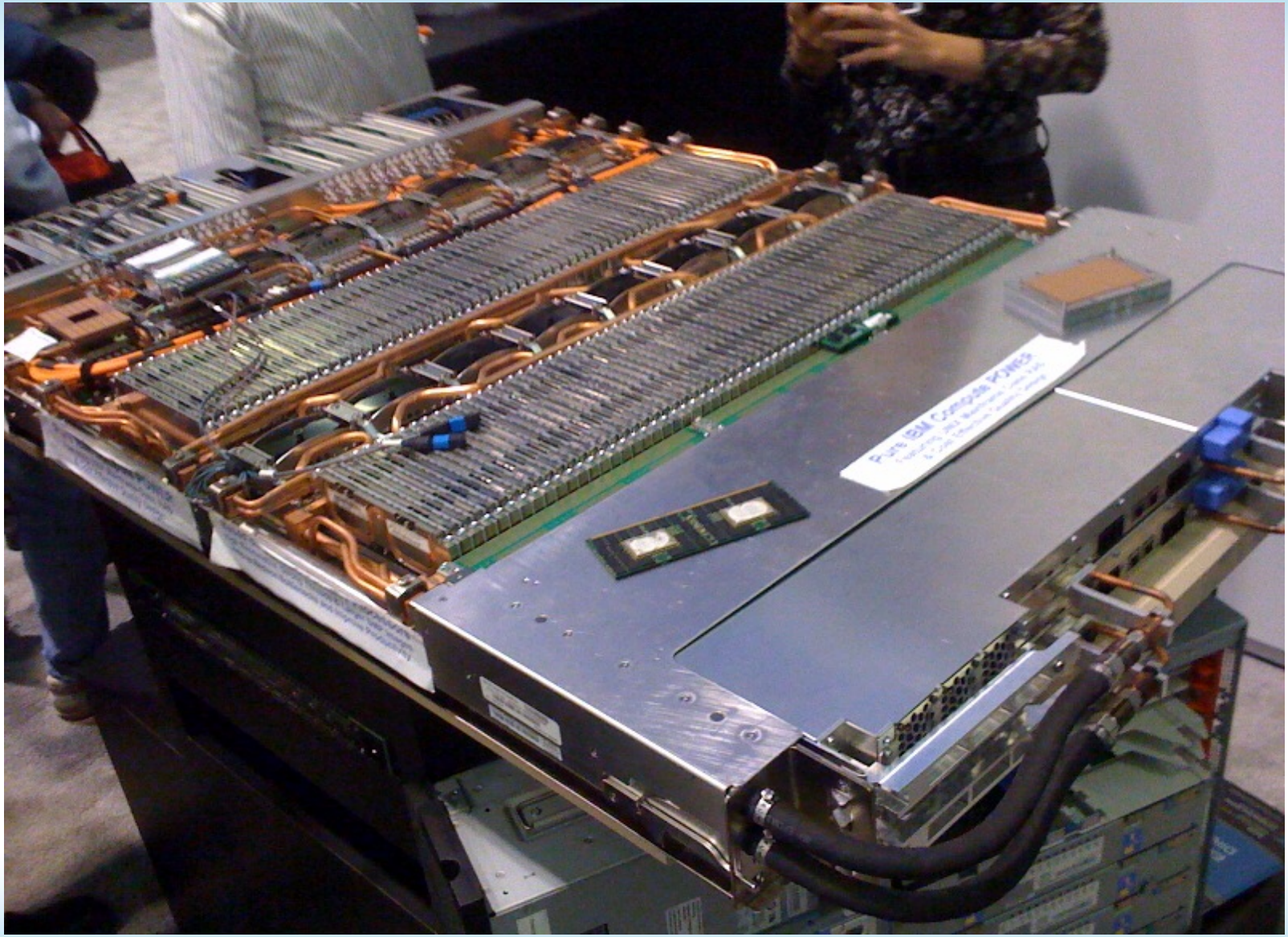
So says the International Technology Roadmap for Semiconductors (ITRS)

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]

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



Sidebar: Cooling in the Bipolar Era

- All large bipolar systems were water cooled
- High-performance supercomputers used immersive cooling (fluorinert, liquid nitrogen)
- A Cray 1 used 115 KW power
- Same evolution (high power density, aggressive cooling) happening now with CMOS

"POST-CONVENTIONAL CMOS"

- 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.
- ROI challenges
  - … achieving constant/improved ratio of … cost to throughput might be an insoluble dilemma.
- …These are huge industry challenges to simply imagine and define
-
- Note: feature size in 2021 (13 nm) = ~55 silicon atoms (Si-Si lattice distance is 0.235 nm)

The Post-Moore Era

- **Scaling is ending**
  - Voltage scaling ended in 2004 (leakage current)
  - Scaling rate will slow down in the next few years
  - Feature scaling will end in 202x (not enough atoms)
    - 13 nm ≈ 55 silicon atoms
  - Scaling may end much earlier, because of technological or economic barriers
    - Continued scaling in the next decade will need a sequence of (small) miracles (new materials, new structures, new manufacturing technologies)
    - Scaling ends, when doubling performance means doubling cost

Rock's Law

- Cost of semiconductor chip fabrication plant doubles every four years
- Current cost is $7-$9B
- Intel's yearly revenue is $35B

- Memory cost has not decreased in the last few years

IT After the End of Scaling (1)

- IT industry changes in fundamental ways
  - Market is driven at the top by function and fashion – not performance
  - All increases in hardware performance are driven by multicore parallelism in the next few years
  - It's a slow slog for improved compute efficiency, afterwards
  - Computer Architecture becomes a true engineering discipline (the end of superficiality)

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 understanding of inherent limitations of technology

HPC – The Canary in the Mine

- **HPC is already heavily constrained by low compute efficiency**
  - High power consumption is at the limit of current machine rooms: a future exascale system may require > 500MW.
  - Low thread performance entails high levels of parallelism: a future exascale system may need ~ 1B threads.
- **Higher compute efficiency is essential for exascale computing**
- **Current Petascale systems are ideal test-bed for research to increase compute efficiency**

Exascale in 2018 at 30 MWatts (?)

- It's impossible [Kogge's report]
  - Conventional designs plateau at 100 PF (peak) – all energy is used to move data
  - Aggressive design is at 70 MW and is very hard to use
    - 600M instruction/cycle
    - 0.0036 Byte/flop
    - No ECC, no redundancy
    - No caching (addressable workpad)
    - HW failure every 35 minutes
    - …
- The report did not account for all energy consumption
  - Actual number is ~500 MW
- Waiting 3-4 years does not solve the problem
  - ***Exascale in CMOS requires revolutionary advances in compute efficiency***

Increasing Compute Efficiency (Software)

- Resiliency
- Communication-optimal computations
- Low entropy computations
- Steady-state computations
- Friction-less software layering
- Self-organizing computations

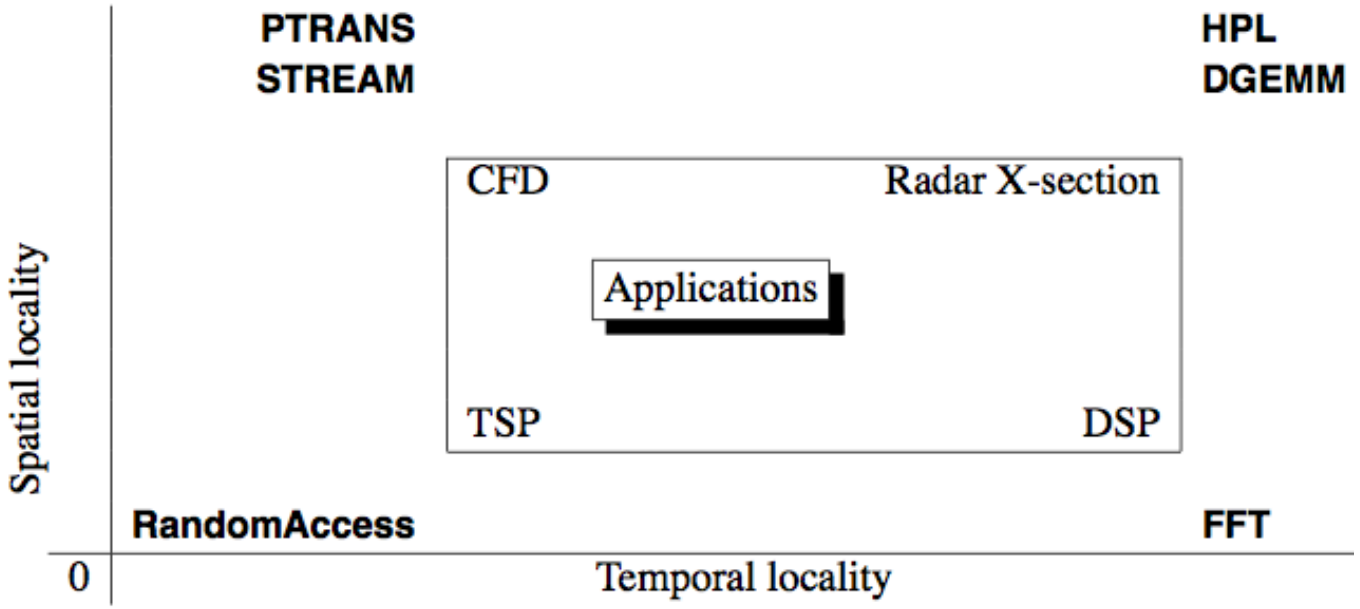
Resiliency

- HW for fault correction (and possibly fault detection) is too expensive
  - and is source of jitter
- Current global checkpoint/restart algorithms cannot cope with MTBF of few hours or less
- SW (language, compiler, runtime) support for error compartmentalization
- Fault-tolerant algorithms
- Research community has limited sources of information on types of failure and failure rates
  - Industry keeps such information confidential
  - Transient HW errors usually cause SW failures – root cause analysis is hard

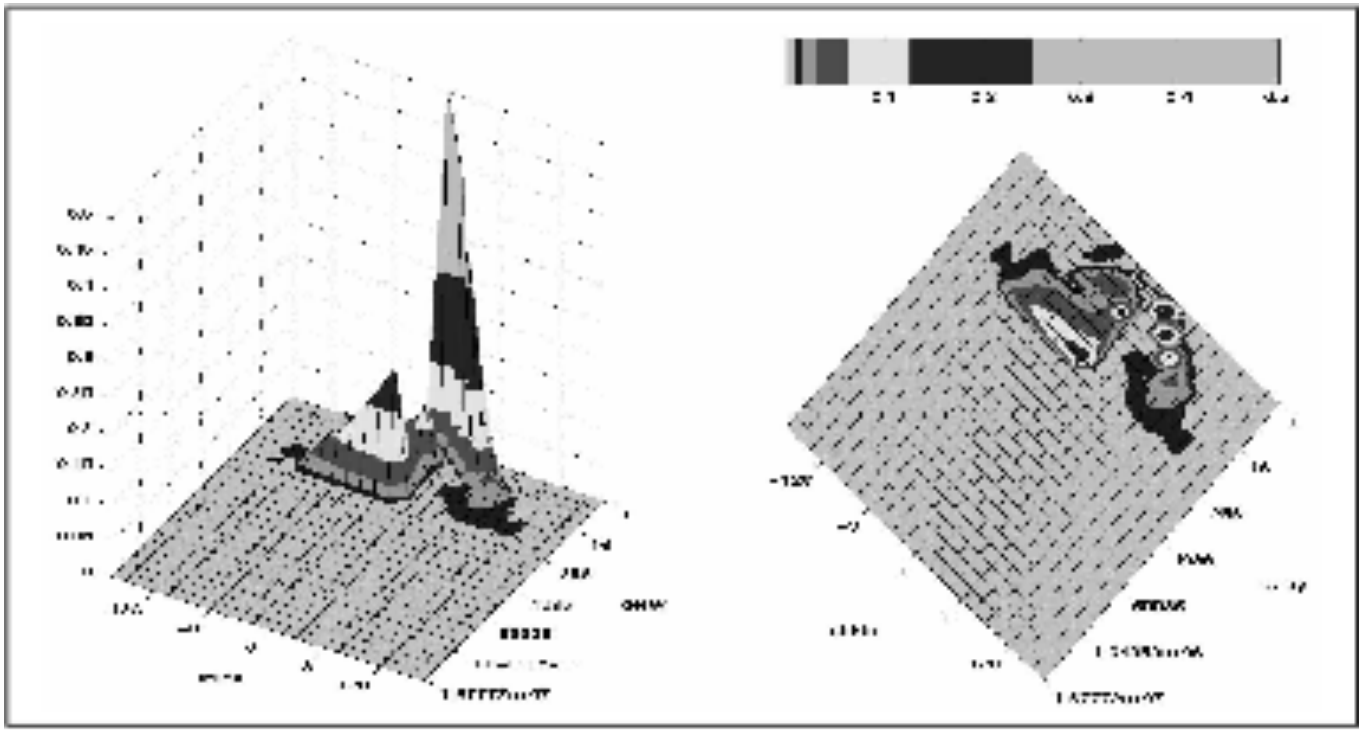
Communication-Efficient Algorithms

- Communication in time (memory) and space is, by far, the major source of energy consumption
- Our understanding of inherent communication needs of algorithms and communication-efficient algorithm design is very limited (FFT, dense linear algebra)
- Current characterization of communication patterns is deficient (dimensionality of space is not understood)
- Need:
  - Better theory of communication complexity
  - Better benchmarks
  - Better metrics
  - Communication-focused language and perf. analysis

Naïve 2D View



Spatio-Temporal Locality Surface



Gzip

Sorenson-Flanagan

Parallelism introduces (at least) one extra dimension

Low-Entropy Communication

- Communication can be much cheaper if "known in advance"
  - Latency hiding, reduced arbitration cost, bulk transfers… bulk mail vs. express mail
- Current HW/SW architectures take little advantage of such knowledge
- CS is lacking a good algorithmic theory of entropy
- Need theory, benchmarks, metrics

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 matching enables to regain lost performance
  - Inlining, on-line compilation, code morphing (Transmeta)
  - Similar techniques are needed for OS layers
- Increased customization becomes possible in the post-Moore era.

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

Summary

- We shall live in interesting times

QUESTIONS?