Visualization at the Exa-scale

Juan Hernando Vieites
jhernando@fi.upm.es

ECAR 2012

Why higher compute power?

Capacity vs. capability

- **Capacity**: Refers to problems already solvable but being able to do it faster or with more instances.
- **Capability**: Refers to problems that are only solvable if we have enough compute power.

More specifically, higher compute power enables:

- Higher resolution
- More timesteps
- More variables or detailed Physics
- Ensembles

Epochs of High Performance Computing

Epochs

- **Giga-scale**:
  - ILLIAC IV, 1972
  - GFLOPS + GB of memory + TB of disk
- **Tera-scale**:
  - Intel ASCI Red, 1997
  - TFLOPs + TBs of memory + TBs/PBs of disk
- **Peta-scale**:
  - IBM Roadrunner, 2009
  - PFLOPs + PBs of memory + PBs of disk
- **Exa-scale**:
  - Estimated date of achievement: 2018
  - EFLOPs + PBs of memory + EBs disk

Impact of transitions between Epochs

From giga-scale to tera-scale

- **Highly disruptive.**
- Big architectural changes: from vector processors to MPI.
- New algorithms and software need to be developed.
- Nonetheless, brute force algorithms work OK.

From tera-scale to peta-scale

- **Not very disruptive.**
- Incremental advances over the established paradigm (MPI + Fortran/C/C++).
- CPU vs storage architectural gap widening, IO bandwidth becoming an issue.
- Out-of-core, sub-setting, in-situ, compression used to de-emphasize IO.

Impact of transitions between Epochs

And what about the transition to exa-scale?

Impact of transitions between Epochs

And what about the transition to exa-scale?

It will be more painful than any previous one

The Exa-scale Computing Challenge

Requirements of the exa-scale computer

- It must cost less than $200 M (and $20 M operational costs).
- It must consume less than 20 MW
- It should be ready by 2018

Source: Dongarra & Beckman et al., *The International Exascale Software Project Roadmap*, Int. J. High Perf. Comput. Appl., 2011

Why spend resources to tackle the challenge?

- To solve capability problems.
- And not researching on exa-scale won't make peta-scale easier.

The Exa-scale Computing Challenge

Premises

- Moore's Law still seems to hold:
  "The number of transistors on die doubles every 18 months"
- However, Dennard's Scaling Law does not hold anymore:
  "When scaling down MOSFETs, they behave as voltage controlled with improved performance as long as all important model parameters are scaled consistently"

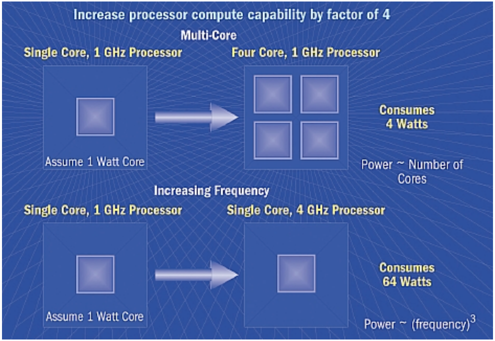
Dennard Scaling is broken because of gate leak currents.

- **Power consumption** has become the biggest concern ...
- ... clock frequencies have stalled ...
- causing single core computing power to plateau since 2004.

Hurdle #1

Power consumption

Power limitations require slower cores and higher paralelism: 4 times faster 64 times more power consuming.

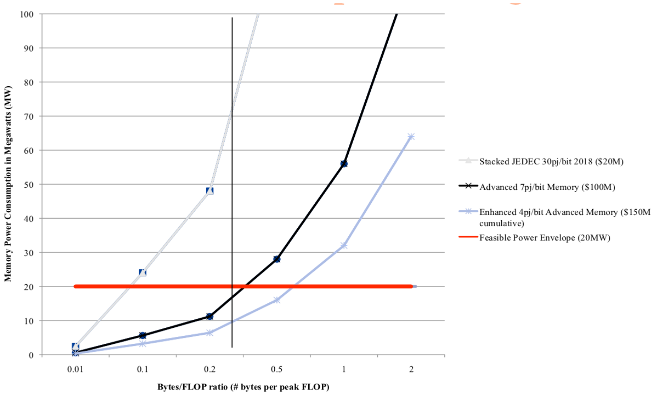


Credit: SciDAC Review 16, February 2010

Hurdle #2

Memory bandwidth

Memory bandwidth eats up the entire power budget

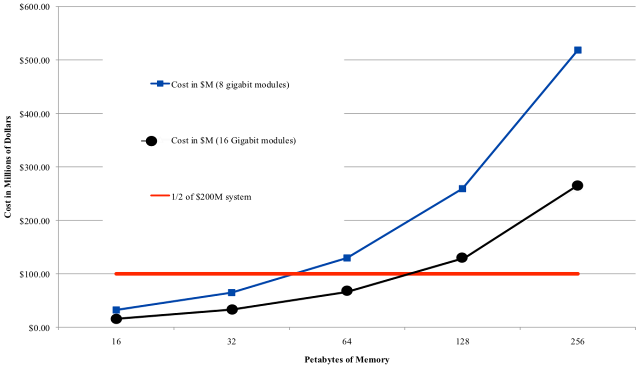


Credit: John Shalf, Lawrence Berkley National Laboratory

Hurdle #3

Memory capacity

Memory capacity eats up the entire fiscal budget

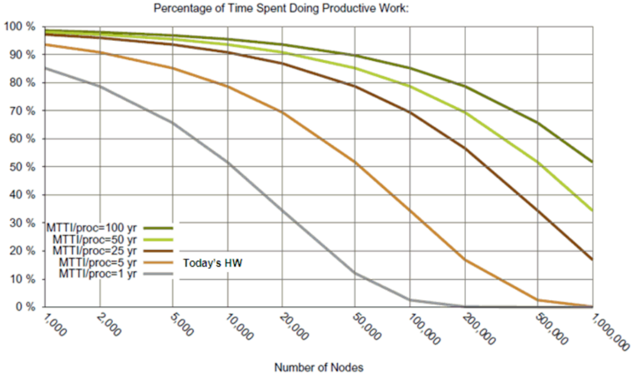


Credit: John Shalf, Lawrence Berkley National Laboratory

Hurdle #4

Reliability

High concurrency causes unreliability



Credit: Rick Stevens, Argonne National Laboratory

Swim lanes

- On the CPU side there are two possible paths, based on some type of accelerator technology
  - BlueGene-like successor
  - GPGPU inspired (what about power consumption?)
- These two approaches are called swim lines.
- Still not clear which one will be used.
- And what about Intel many cores and mobile processors?
- However, the real challenge is not so much how to put more cores, but how to take advantage of them while the memory bandwidth and capacity expand ten-fold less.

An educated speculation is that exa-scale supercomputers will be an heterogeneous system with:

Giga-Hz Kilo-Core Mega-Node

Swim lanes

System Parameter	2011	Swim Lane 1	Swim Lane 2	Factor Change
System Peak	2 Pf/s	1 Ef/s		500
Power	6 MW	≤20 MW		3
System Memory	0.3 PB	32–64 PB		100–200
Total Concurrency	225K	1B×10	1B×100	40,000–400,000
Node Performance	125 GF	1 TF	10 TF	8–80
Node Concurrency	12	1,000	10,000	83–830
Network BW	1.5 GB/s	100 GB/s	1000 GB/s	66–660
System Size (nodes)	18700	1,000,000	100,000	50–500
I/O Capacity	15 PB	300–1000 PB		20–67
I/O BW	0.2 TB/s	20–60 TB/s		10–30

Credit: Rick Stevens, Andrew White, et al. *Architectures and technology for extreme scale computing.* Technical report, ASCR Scientific Grand Challenges Workshop Series, 2009

Data Movement

- The exa-scale machine is being designed for data producers, not consumers.
- Moving the data out of the machine is not feasible.
- Moving the data inside the machine will be expensive.
- Producing data to be sitting around on tape will be useless.
- Alternative strategies for Data Management, Analysis and Visualization are required.

In-situ visualization

- More than an alternative, in-situ visualization will become a necessity.

Considerations

In a tightly coupled scenario:

- Simulation and analysis/visualization have to share resources in an already constrained environment (zero-copy when possible).
- Visualization/analysis code can't jeopardize simulation stability and performance.
  - Visualization must scale at least as well as the simulation.
  - And can't cause failures.

In-situ visualization

Use case coverage

- Remembering the 3 uses cases of visualization:
  - Debugging/confirmation:
  - Discovery/exploration:
  - Dissemination/communication:
- The first two can be covered. For the third one there are additional problems.

In-situ exploration

- Having a human in the loop may be expensive for this resource.
- In-situ could be used to produce reduced data sets for later post-processing.
- But even in that case, some prior knowledge is still required.

Programming an exa-scale machine

What type of language do we need?

- Support for massive parallelism.
- Potentially heterogeneous architectures.
- Facilitate explicit memory management.
- Fault tolerance?
- Until this problem is solved, MPI will have to be combined with other languages (OpenMP, OpenCL)

What type of middleware?

- APIs like VTK won't be suitable as they are right now.
- Can middle-ware hide the complexities?

Conclusions

The exa-scale machine

- Exa-scale computing will be possible by massive parallelism.
- Less relative memory per node and less bandwidth that in peta-scale.
- Data movement will be very expensive, FLOPS will be cheap.

Visualization problems

- Analysis and visualization will have to be coupled to the simulation code.
- Many open challenges: from visualization specific research to programming languages and middlewares.