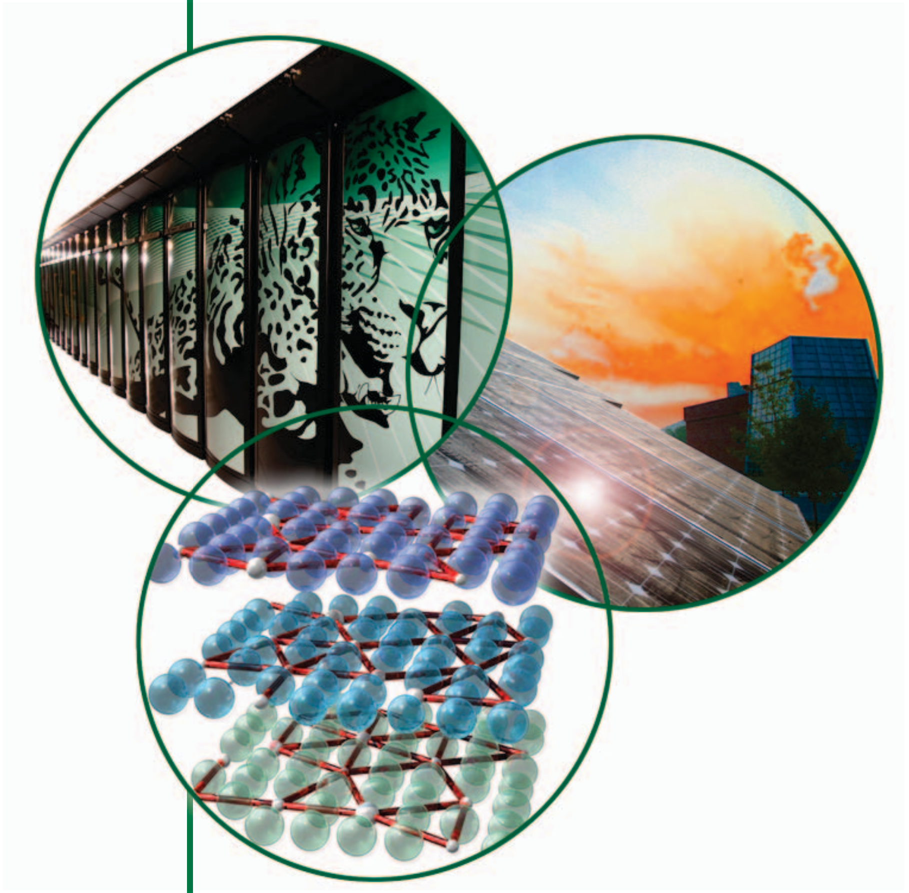
The Effect of Emerging Architectures on Data Science (and other thoughts)

Philip C. Roth

With contributions from Jeffrey S. Vetter and Jeremy S. Meredith (ORNL) and Allen Malony (U. Oregon)

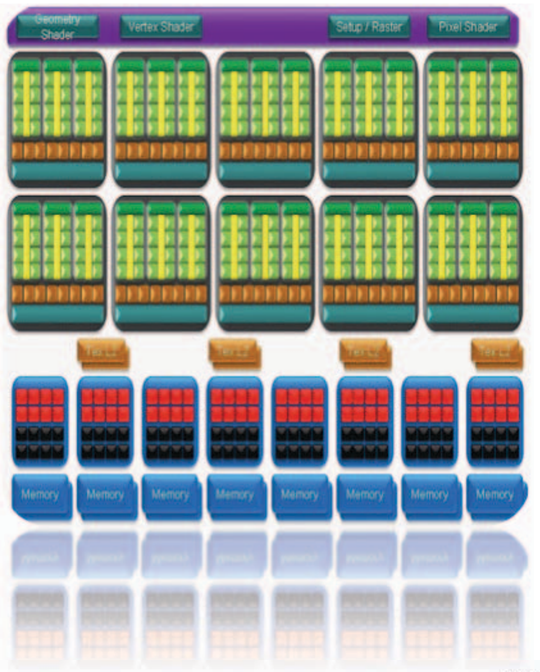
Future Technologies Group
Oak Ridge National Laboratory

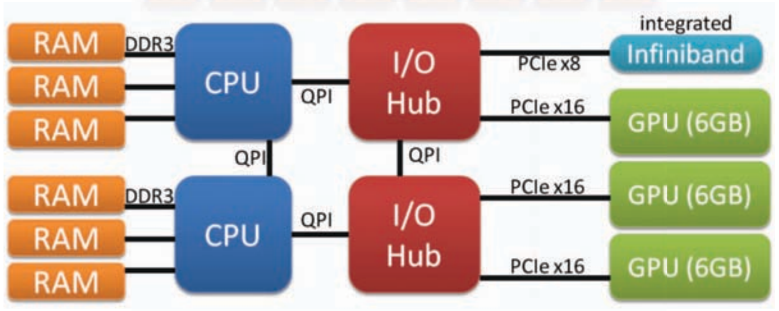
rothpc@ornl.gov



Emerging (Emerged?) Technology I

- Accelerators (DOE Vancouver, NSF Keeneland)
  - Graphics Processing Units (GPUs)
  - Manycore (e.g., Intel's Many Integrated Core)
  - Field Programmable Gate Arrays (FPGAs)
  - Potential for many more threads of execution
    - Great performance, if you can make use of them
    - Many more "events" per walltime unit
  - Tighter coupling between CPU and accelerator (e.g., GPUs/FPGAs in HyperTransport sockets, AMD Fusion)
  - Open architectural questions about relative role of host to accelerator
    - Reduced role of host (CPU) – e.g., NVIDIA project Denver
    - NIC integrated with the GPU?





Emerging (Emerged?) Technology II

- Memory Hierarchy
  - NVRAM (e.g., flash, Phase Change Memory) (DOE Blackcomb project)
  - Solid State Drives (SSDs)
  - Higher performance (but smaller capacity) storage, close to the processor
  - Open questions in programmability (as memory or as disk? Memory mapped?)
  - Potential uses:
    - Burst buffers (e.g., for checkpoints, event traces)
    - Out-of-core algorithms
- Networking
  - High bandwidth traditional networking technologies (e.g., FDR InfiniBand), smart (programmable) NICS
  - Photonics – High throughput, low latency increases effectiveness of in situ analysis
- Programming Models
  - CUDA, OpenCL, OpenACC, OpenMP
  - Domain Specific Languages
  - Interesting that no one at the workshop talked about the community darling programming models (MapReduce)

http://keeneland.gatech.edu

Keeneland

Full Scale WIP

[illegible]

[illegible]

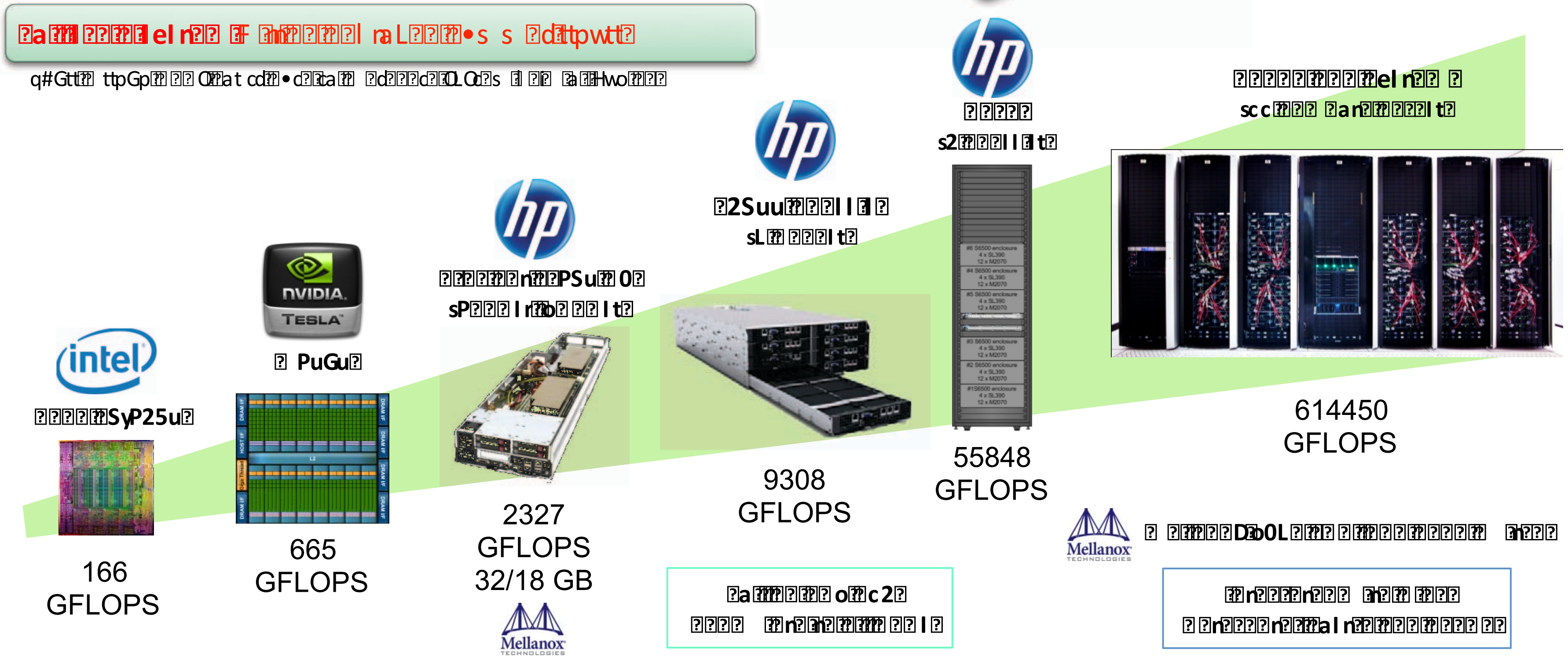
[illegible]

[illegible]

[illegible]

[illegible]

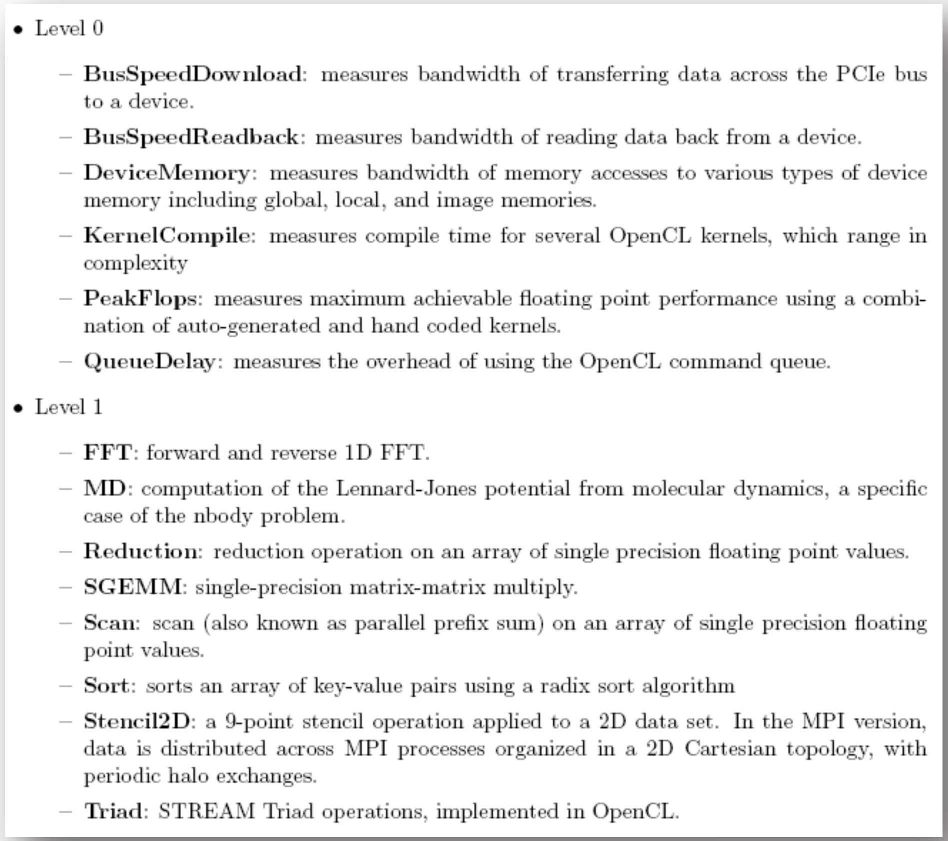
[illegible]



J.S. Vetter, R. Glassbrook *et al.*, *"Keeneland: Bringing heterogeneous GPU computing to the computational science community," IEEE Computing in Science and Engineering*, 13(5):90-5, 2011, http://dx.doi.org/10.1109/MCSE.2011.83.

The Scalable HeterOgeneous Computing (SHOC) Benchmark Suite

- **Benchmark suite with a focus on scientific computing workloads, including common kernels like SGEMM, FFT, Stencils**
- **Parallelized with MPI, with support for multi-GPU and cluster scale comparisons**
- **Implemented in CUDA and OpenCL for a 1:1 performance comparison**
  - **Will be adding OpenACC versions soon**
  - **Have contributions from Intel for MIC**
- **Includes stability tests**



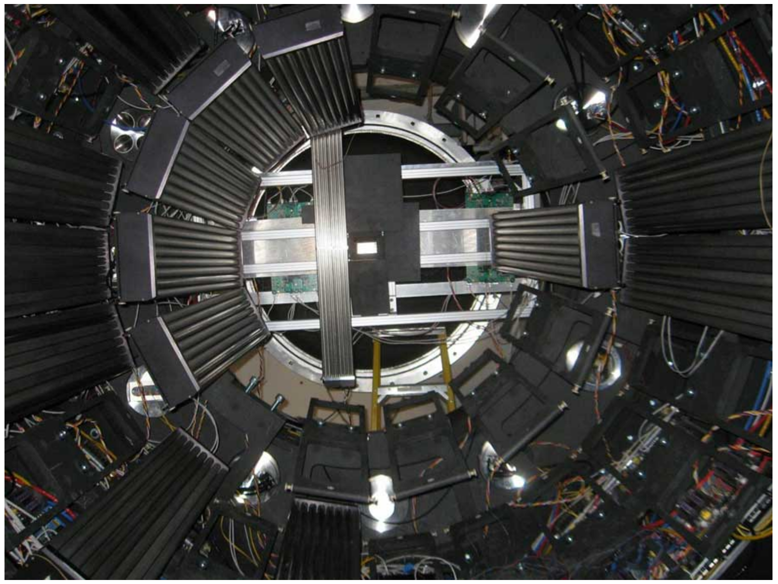
- Level 0
  - **BusSpeedDownload**: measures bandwidth of transferring data across the PCIe bus to a device.
  - **BusSpeedReadback**: measures bandwidth of reading data back from a device.
  - **DeviceMemory**: measures bandwidth of memory accesses to various types of device memory including global, local, and image memories.
  - **KernelCompile**: measures compile time for several OpenCL kernels, which range in complexity
  - **PeakFlops**: measures maximum achievable floating point performance using a combination of auto-generated and hand coded kernels.
  - **QueueDelay**: measures the overhead of using the OpenCL command queue.
- Level 1
  - **FFT**: forward and reverse 1D FFT.
  - **MD**: computation of the Lennard-Jones potential from molecular dynamics, a specific case of the nbody problem.
  - **Reduction**: reduction operation on an array of single precision floating point values.
  - **SGEMM**: single-precision matrix-matrix multiply.
  - **Scan**: scan (also known as parallel prefix sum) on an array of single precision floating point values.
  - **Sort**: sorts an array of key-value pairs using a radix sort algorithm
  - **Stencil2D**: a 9-point stencil operation applied to a 2D data set. In the MPI version, data is distributed across MPI processes organized in a 2D Cartesian topology, with periodic halo exchanges.
  - **Triad**: STREAM Triad operations, implemented in OpenCL.

[illegible]

Software available at https://github.com/spaffy/shoc/wiki

Data Intensive *Computer* Science

- Usual focus is on data problems from science domain
  - Computational science (simulations)
  - Scientific instruments (e.g., particle detectors)
- System administration and monitoring tools can cause data-related problems too



NOMAD detector
Image courtesy Dr. Jörg Neuefeind, ORNL

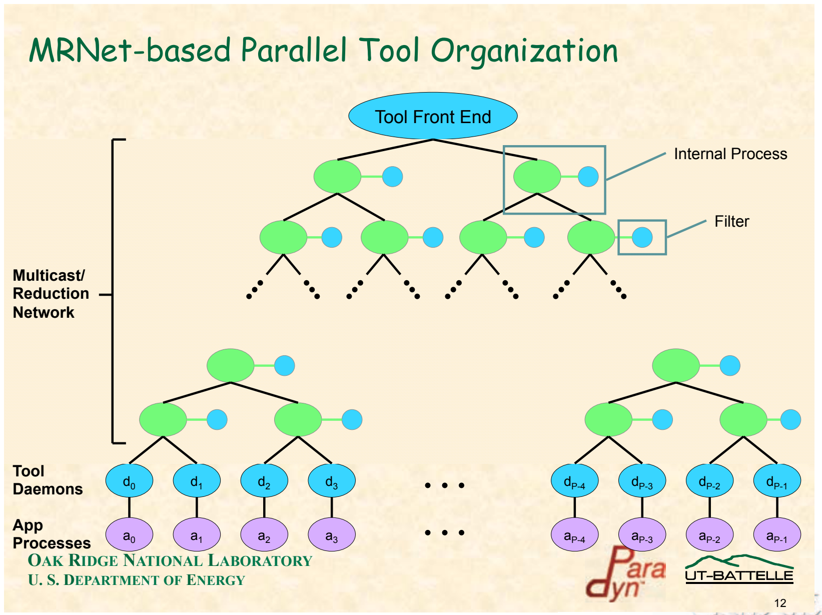
- Event tracing is notorious for causing data *collection*, *management*, and *analysis* problems
  - Similar to that particle detector…
  - …except that we often want to analyze the data *online* so we can make some change
  - Emerging architecture (e.g., GPUs) can greatly exacerbate the problem

Data Intensive *Computer* Science: Example

- Example from my past: Paradyn parallel performance tool (RIP), Bart Miller, U. Wisconsin
- Tool used calipers (inserted using Dynamic Instrumentation) to generate performance data
  - On-line analysis feeds decisions about what instrumentation to insert/remove as program runs
- Tool daemons sampled that data and sent to tool's front end for analysis
- Performance data volume could be large, due to:
  - High sampling rate
  - Large number of active metrics
  - Large number of monitored processes

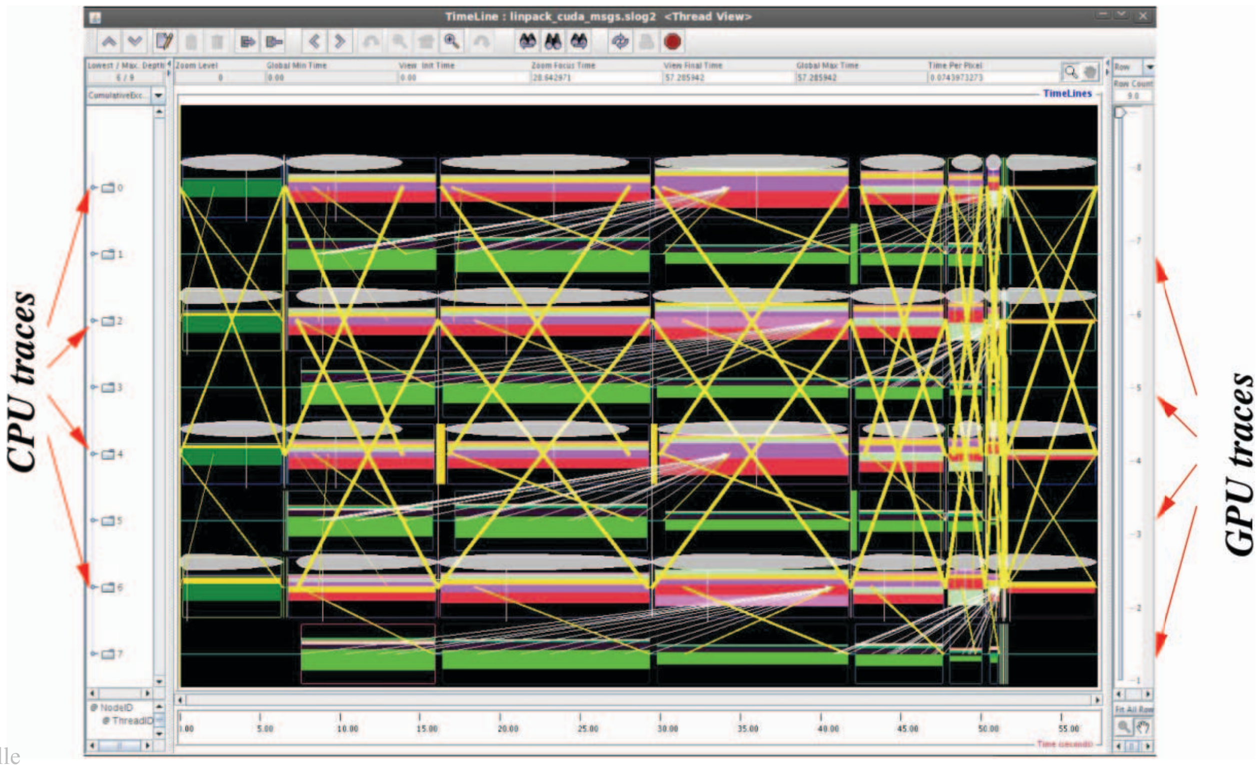
Data Intensive *Computer* Science: Example

- Our approach: develop and use a multicast/reduction network (MRNet) to reduce data within an overlay network before it reaches the tool's front end
- Interesting analogous to in situ analysis
  - Where to run MRNet internal processes?
  - What filters are needed? How to synchronize streaming data in filters?
- Today: Hadoop (in a separate analysis cluster)? GPU accelerated reduction filters?
- Another example: Tiwari et al, "Quantifying the Potential for Program Analysis Peripherals," PACT 09 – shows benefit of using FPGAs to accelerate valgrind-based analysis tools


MRNet-based Parallel Tool Organization
Tool Front End
Internal Process
Filter
Multicast/ Reduction Network
Tool Daemons
App Processes
d_0 d_1 d_2 d_3 . . . d_{P-4} d_{P-3} d_{P-2} d_{P-1}
a_0 a_1 a_2 a_3 . . . a_{P-4} a_{P-3} a_{P-2} a_{P-1}
Oak Ridge National Laboratory
U. S. Department of Energy
Paradyn
UT-Battelle
12

Visualizing Performance Data

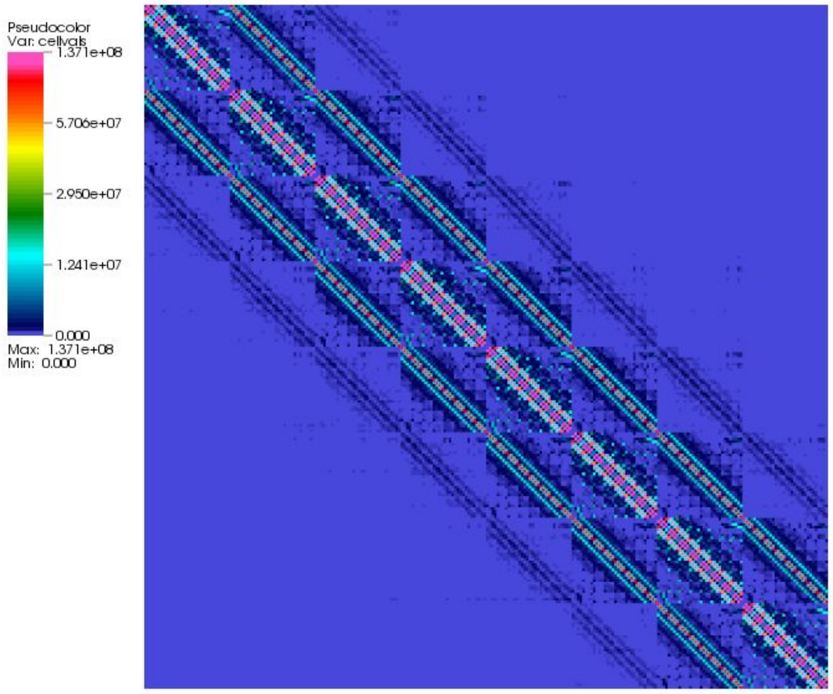
- **Machines have long been large enough to motivate research into scalable performance data visualization**
- **Architectures with GPUs, manycore exacerbate the problem (greater data volume)**



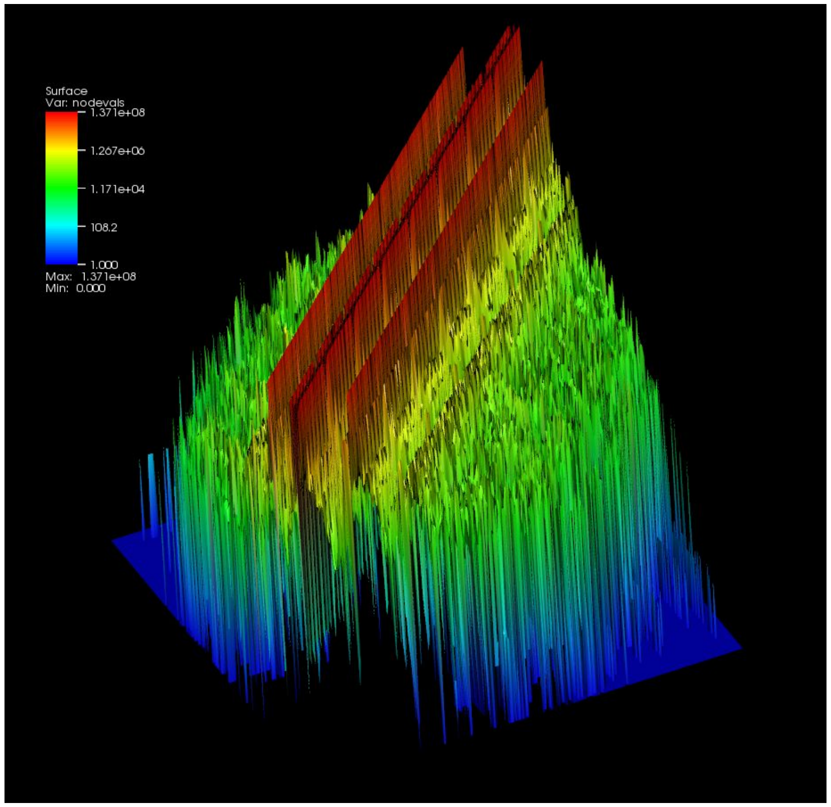
TAU images courtesy Allen Malony, U. Oregon, http://tau.uoregon.edu

Visualizing Performance Data

- **How well do traditional science visualization techniques apply to performance data?**



VisIt images courtesy Jeremy Meredith, ORNL



Summary

- Emerging technology in compute, memory hierarchy, and interconnect
  - Promise of increased performance, larger opportunities for online analysis like in situ visualization
  - But: can make performance data analysis and visualization much more difficult

- Don't forget: there are data problems in computer science domain too
  - Do same techniques apply?

- For more information:
  - rothpc@ornl.gov
  - http://ft.ornl.gov

  - Keeneland (NSF Track 2D): http://keeneland.gatech.edu
  - Vancouver (DOE X-Stack): http://ft.ornl.gov/trac/vancouver
  - Institute for Sustained Performance, Energy, and Resilience (SUPER, DOE SciDAC-3): http://super-scidac.org