Embracing Uncertainty: Performance Analysis and Prediction for Next-Generation Architectures

Jeffrey S. Vetter

Future Technologies Group

Our Team and Collaborators

- Future Technologies Group
  - Sadaf Alam
  - Richard Barrett
  - Nikhil Bhatia
  - Ken Roche
  - Philip Roth
  - Olaf Storaasli (Sept 1)
  - Jeffrey Vetter
- Collaborators
  - Melissa Smith
  - Pat Worley
  - Tom Dunigan
  - Pratul Agarwal
  - SciDAC PERC Team
  - DARPA HPCS Team
  - U of Oregon Tau Team
  - U of Tenn ICL Team
  - Vendors
  - Many others…

- Performance analysis, evaluation, and modeling of architectures in support of scientific computation
- Research and development of software and algorithms for HPC
- ExCL (Experimental Computing Lab) for examining new technologies
  - FPGAs
  - Array processors
  - Optical processors
  - Multicore processors
  - …

http://www.csm.ornl.gov/ft

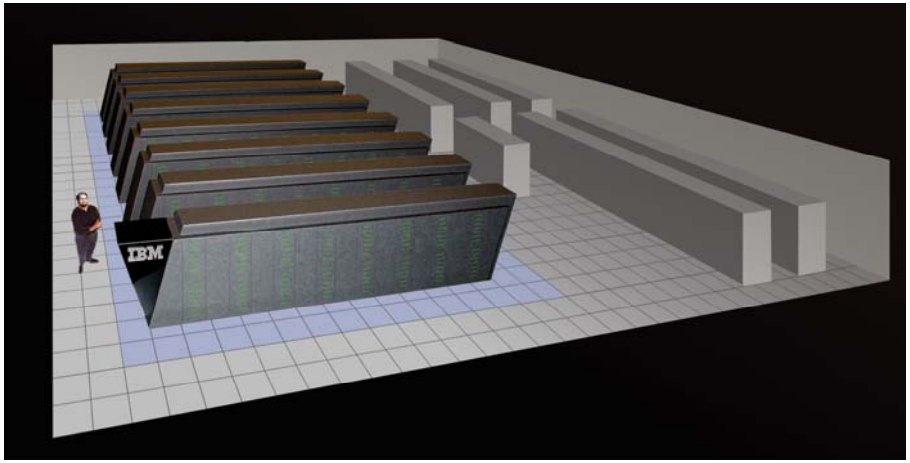
Increasing Uncertainty in Performance Analysis, Evaluation, and Prediction

- The past decade of HPC had processors and interconnects that were ‘relatively’ easy to understand, analyze, and predict

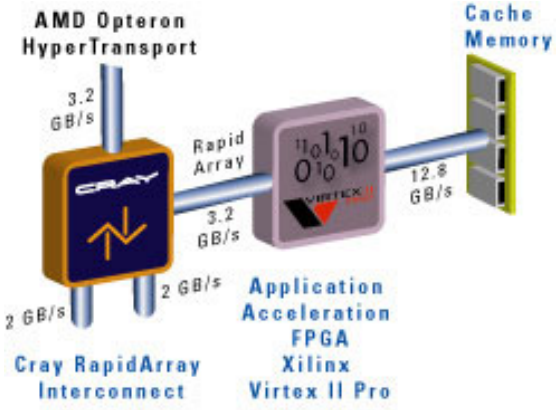
- The next decade of HPC will unveil technologies that make analysis, evaluation, and prediction more challenging
  - Scale
  - Architecture complexity
  - Application complexity

Scale

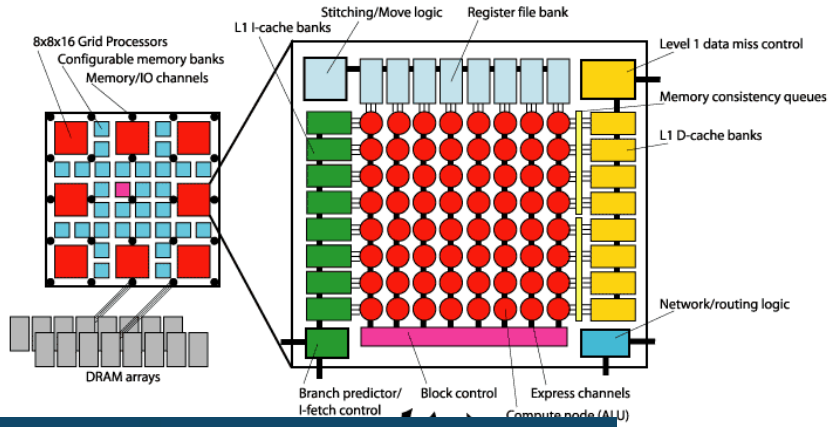
- Past decade has applications that scale from O(1) to O(1000) levels of concurrency
  - Hidden assumptions that algorithms at O(10) are effective at O(1000)
  - MPI specification ensures function portability, not performance portability
- Next decade will needs applications that scale to O(10,000,000) levels of concurrency
  - With appropriate languages, tools, etc
  - Transparent support for reliability

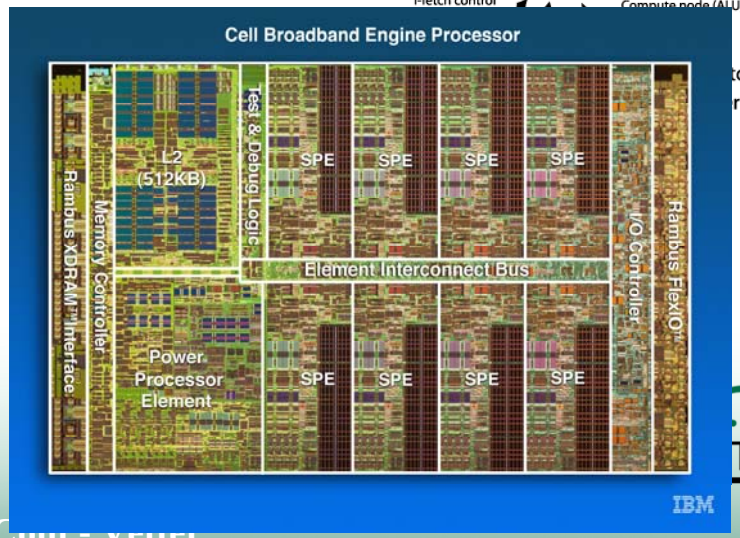


Architectural Complexity



- Architectures are changing radically at the node, core level
  - Multicores, multiple interconnects, etc.
  - Experimental architectures present more challenges
    - Streaming, FPGAs, accelerators
- Performance response of architectures is highly discontinuous
  - Small changes in applications, system software, or architectures can produce 'catastrophic' performance differences





top applications
r applications

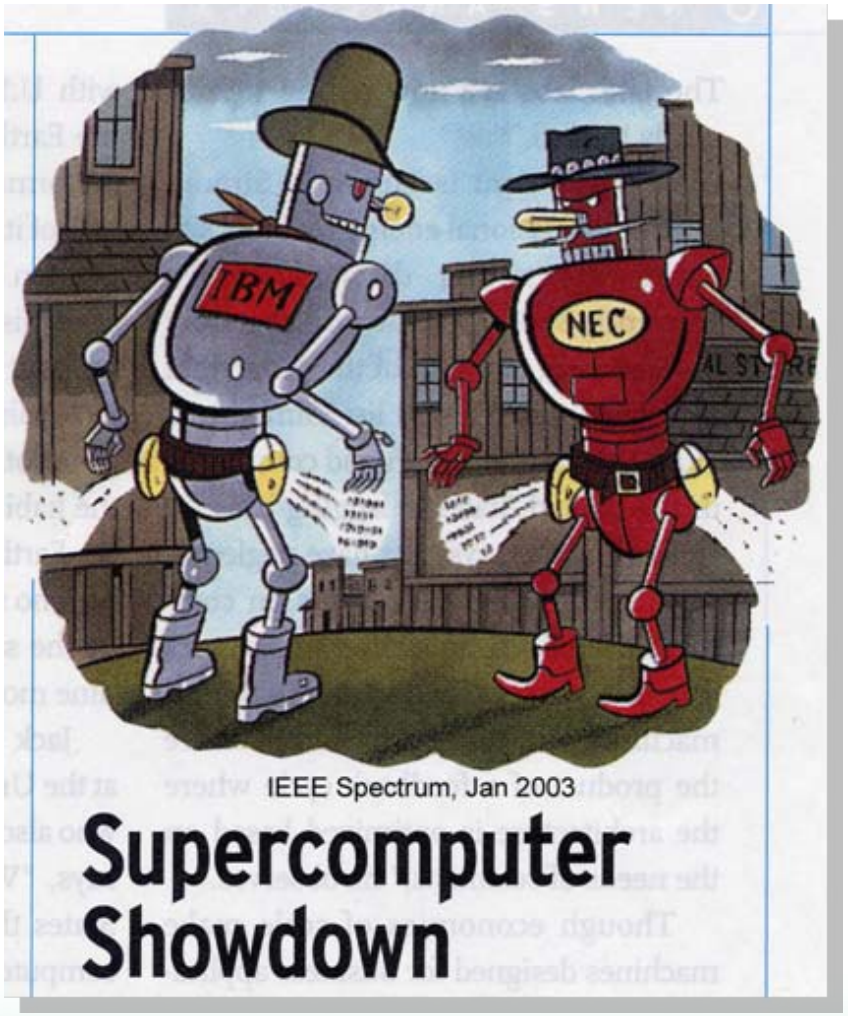
Application Complexity

- Multi-phase, multi-scale applications present challenges in performance uncertainty
  - Multiple languages
  - Multiple phases of physics, chemistry
  - Adaptive meshing, solving, etc
  - What is good performance?
- Applications teams know this best!

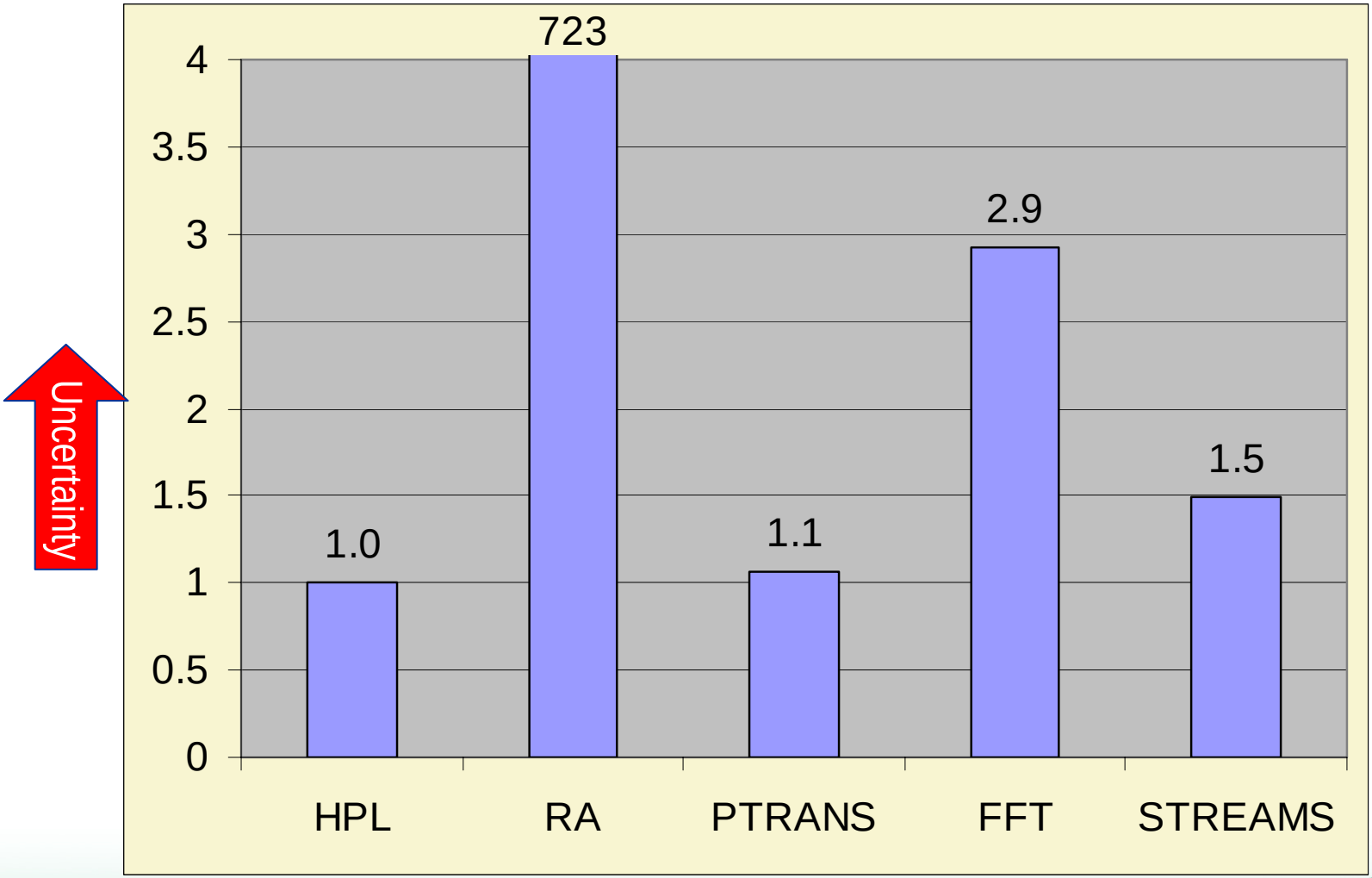
- Science-based metrics that reflect real problems and their characteristics must drive the design and procurement of new systems

An Example of Uncertainty in Performance: HPC Challenge Benchmarks

- HPC Challenge has two versions
- Base
  - Portable: Functional across 'most' platforms
  - Represents legacy codes (or otherwise immutable)
- Optimized
  - Match application characteristics to architectural strengths
- Both versions are valid
- Distance between *Base* and *Optimized* proportional to performance uncertainty?



Improvement on Cray X1 (64 MSPs) Optimized/Baseline



Joint work on HPCC for X1 with Cray, GWU.

Ok, these are all challenging problems. Should we just accept it as is?

Of course not,
We should embrace the uncertainty…

- Know your applications
  - Map application requirements to architectures
  - Use performance analysis and modeling to prioritize optimization, hardware acceleration efforts
- Work closely with applications team to adapt their codes to new technology
  - Integrate performance engineering directly into the software development process
  - Develop scalable techniques for performance analysis
- Evaluate next-generation and experimental architectures for potential acceleration (and feed this back to vendors)

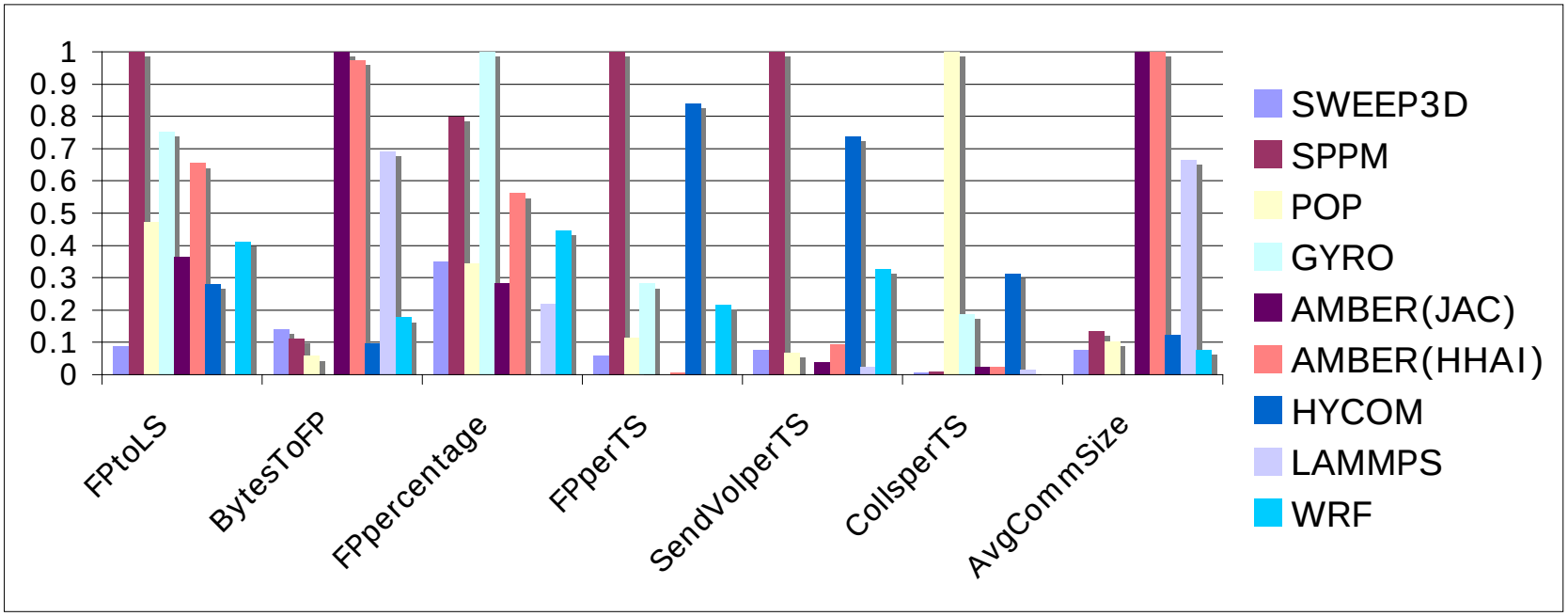
Know your Applications

Empirically Measured Application Characteristics Reveal True Behaviors

- Using Sequoia tracing tool to capture salient communication and computation characteristics of applications
  - MPI command and parameters
  - Computational block summaries
    - Flops, instructions, loads, stores
- Current status
  - POP, GYRO, AMBER(JAC), AMBER(HHAI), HYCOM, LAMMPS, WRF, SWEEP3D, SPPM

Work by Future Tech team.

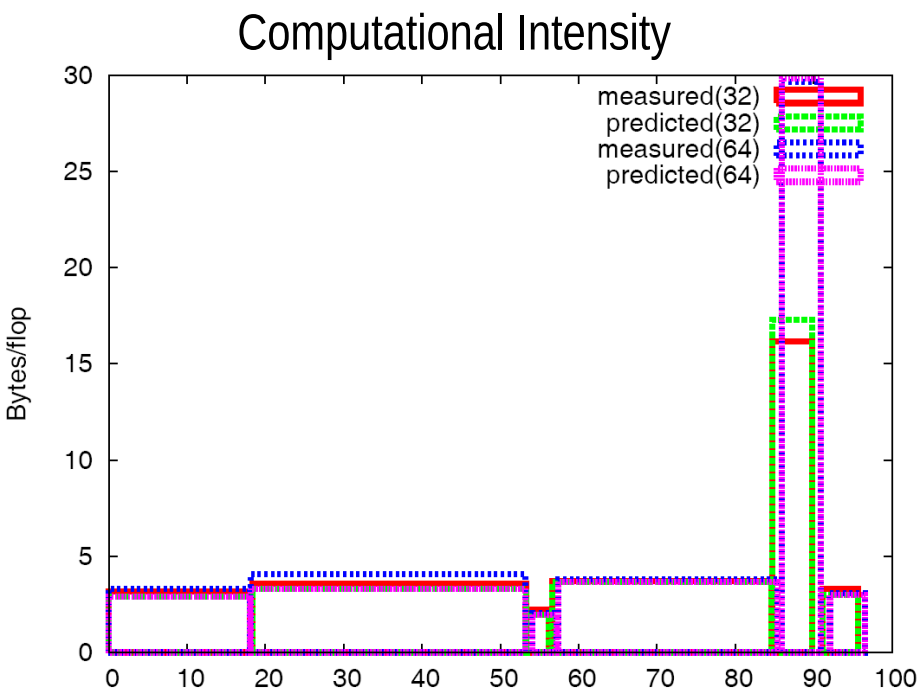
Summarized Empirical Data Show Range of Requirements for Applications

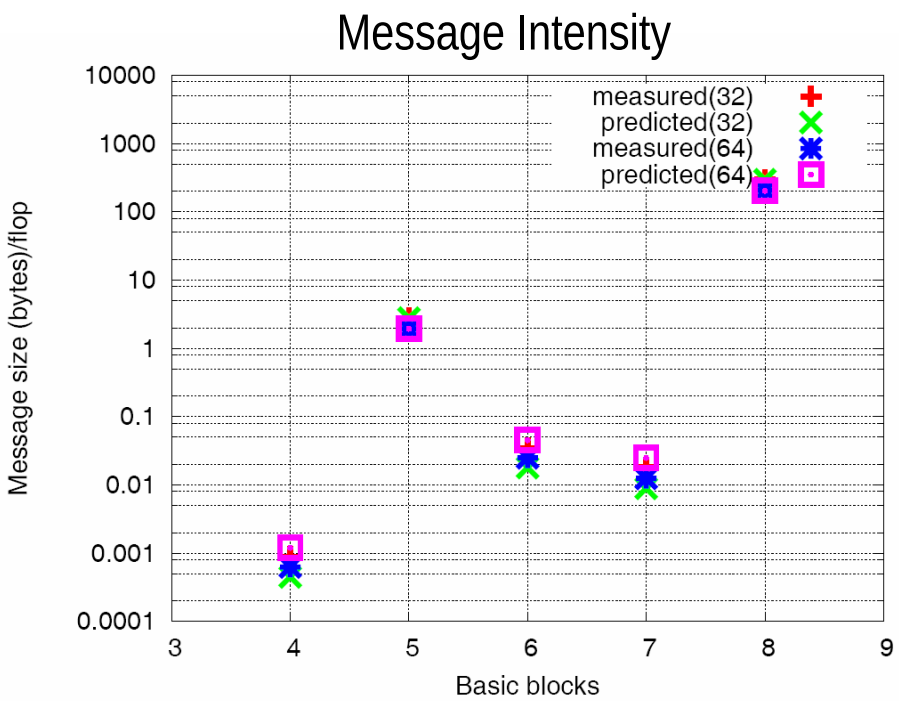


Each metric normalized to range of zero to one.

POP Metric Breakdown by Basic Block

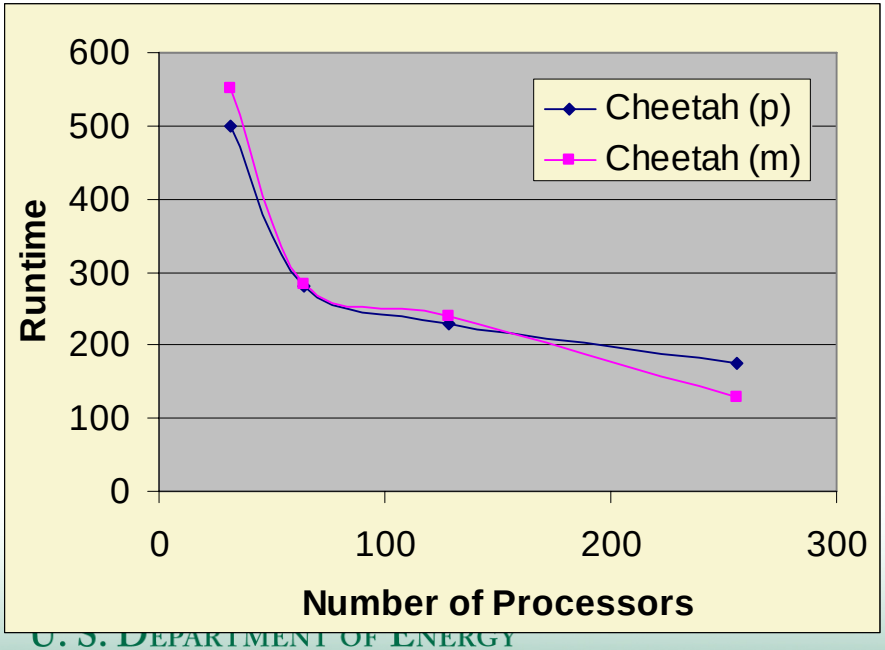
- Developed detailed understanding of POP computation and communication
- Models match measurement closely

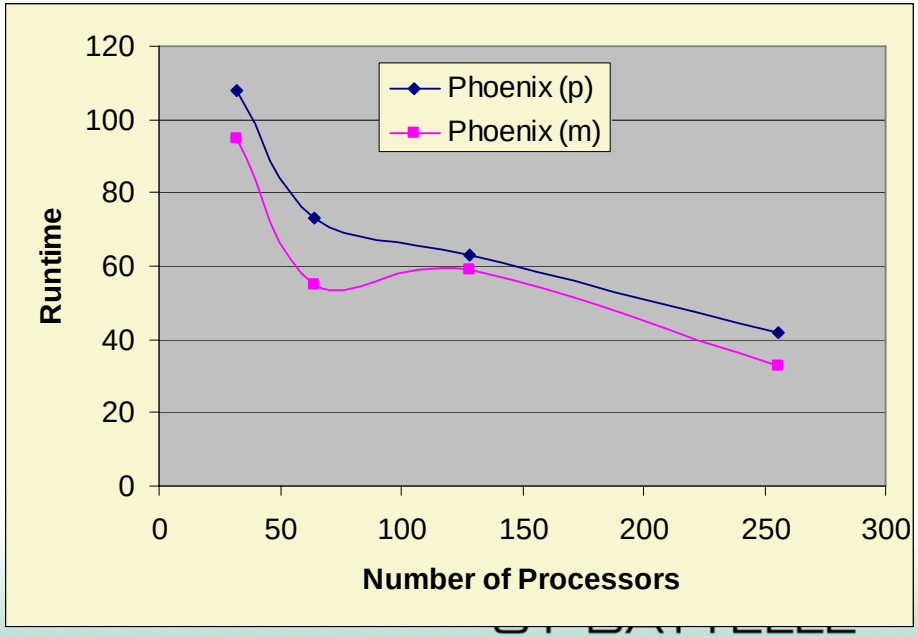




POP Execution Time Predictions

- Use collection of detailed models to create execution time predictions
- Validated POP model on ORNL systems
  - Note that this is a strong scaling model





Models Enable Exploration of Design Space

- With models in hand, we can
  - Explore the parameter design space
  - Predict requirements at larger scale

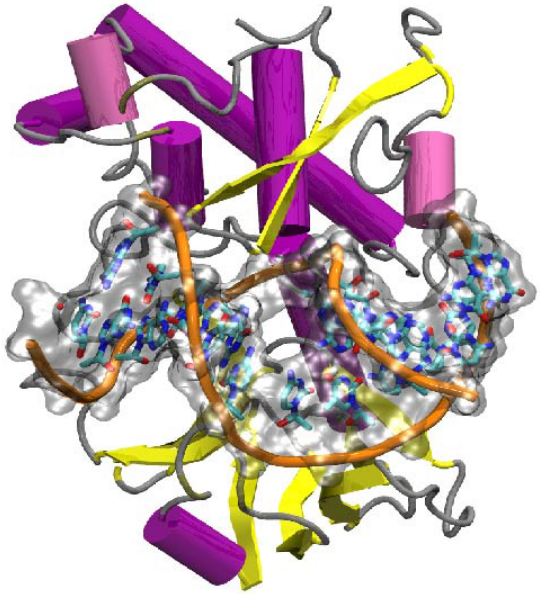
$$
\begin{aligned}
C_{nstep} = \; & C_{barotropic_river} + C_{baroclinic_river} + C_{baroclinic_correct} \\
& C_{set_surface_forcing} + C_{tavg_2d_compute} + C_{dhdt} \\
& C_{boundary_baroclinic} + C_{boundary_barotropic} + 2 * km * C_{state}
\end{aligned}
$$

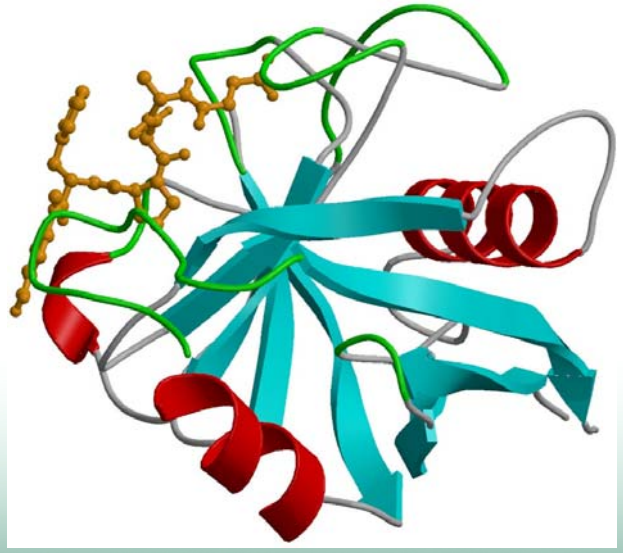
POP		Latency		
		-20%	0%	20%
Bandwidth	20%	0.91	1.04	1.16
	0%	0.88	1	1.14
	-20%	0.85	0.97	1.11

Collaborate with Applications Teams to use New Technology

Computational Biology using Molecular Modeling

- The structure, dynamics and function of biomolecular complexes are inter-related
- Various aspects of biomolecules structure and function span multiple scales of time and length
- Wide community of biologist are interested in the multi-scale modeling of biomolecules
- *Multi-scale modeling of a real system may require 1 peta-flop/s for an entire year!*
- Scaling of existing software packages and algorithms is limited





Joint work between Sadaf Alam and Comp Biologist Pratul Agarwal at ORNL.

Computer Simulations
(Molecular Dynamics)

⇨ Mathematical (potential) function

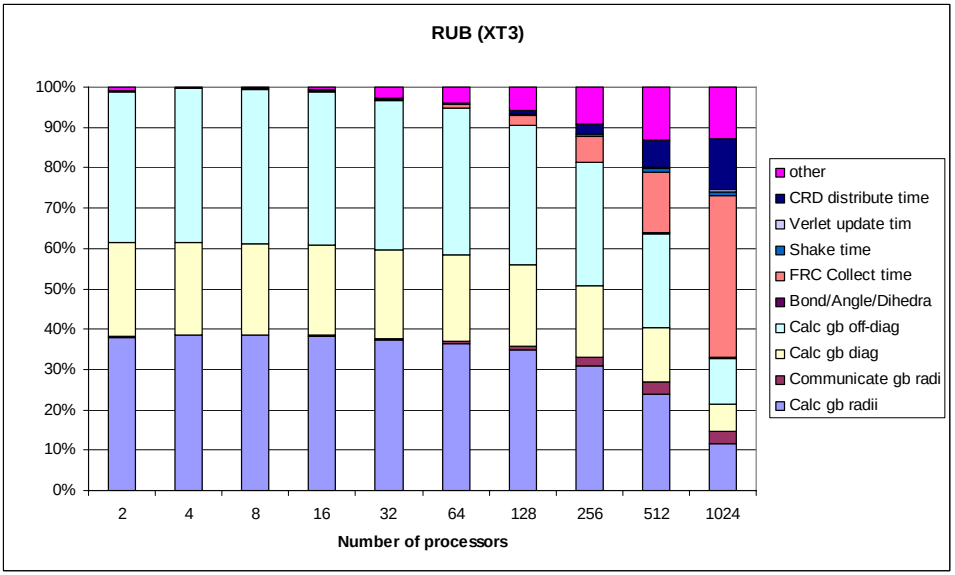
$$V(r^n) = \sum_{bonds} K_l (l - l_{eq})^2 + \sum_{angles} K_\theta (\theta - \theta_{eq})^2 + \sum_{torsions} \frac{V_n}{2}(1 + \cos[n\phi - \gamma]) + \sum_{i=1}^{N} \sum_{j=i+1}^{N} \left(\left[\frac{A_{ij}}{r_{ij}^{12}} - \frac{B_{ij}}{r_{ij}^{6}} \right] + \frac{q_i q_j}{\varepsilon r_{ij}} \right)$$

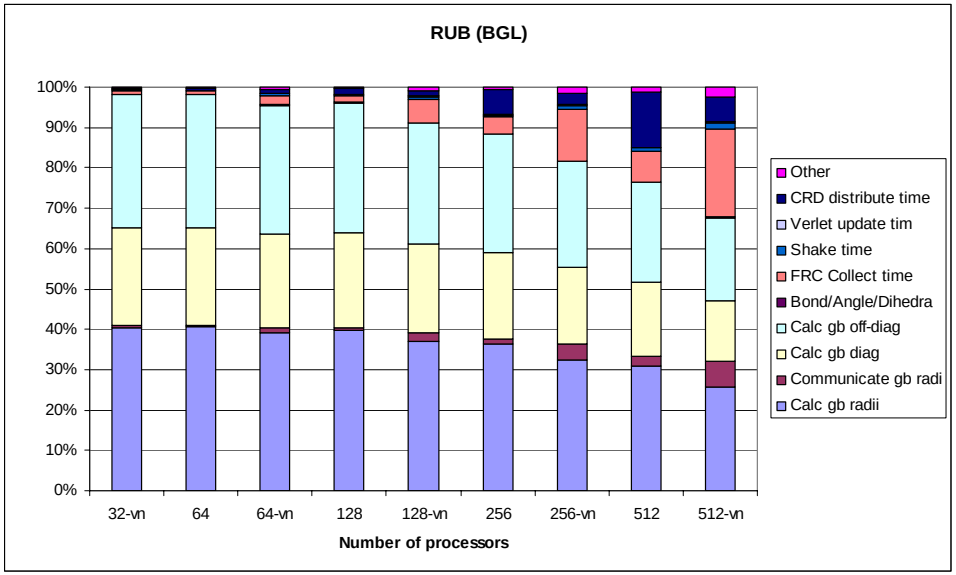
- Bond stretching, angle bending, angle torsion and the non-bond term
- Degree of freedom = 3N-6, where N = number of atoms
- Number of points to sample = M^{3N-6}, M >> 10
- Packages: Amber, GROMACS, GAMESS, LAMMPS, NAMD

AMBER Performance Analysis

- ORNL Computational biologists were using AMBER for their simulations, but its scalability was limited to about 128 processors
- Used several tools to study AMBER's performance
  - MPIP, PAPI, Xprofiler, GPROF
- Modified communication operations to improve scaling
- Identified computational kernels for acceleration with FPGAs

AMBER Profiling on Cray XT3 and IBM BlueGene/L





XT3

Bottlenecks: Distribute, Collect and I/O times

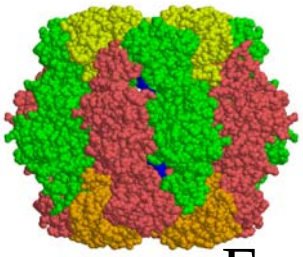
Expected to improve significantly as system matures

BGL

Bottlenecks: Distribute and collect times

Computation and communication times can improve with tool chain

Amber Control Flow for RUB
(RuBisCO with Generalized Born method)



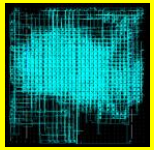
RunMD

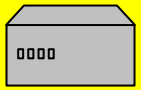
Main method of Sander

Force | Shake | Update | CRD distribute

Non-bond | Bond | FRC Collect

Generalized Born





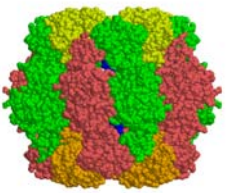
Short_ene: mapped to FPGA
Divided and distributed in MPI model

Cost increases with number of processors

GB Computation | GB Communication

Most expensive on a single node system

RUB Scaling



Gen Born=48811s
FRC Collect=2s
CRD Distribute=0

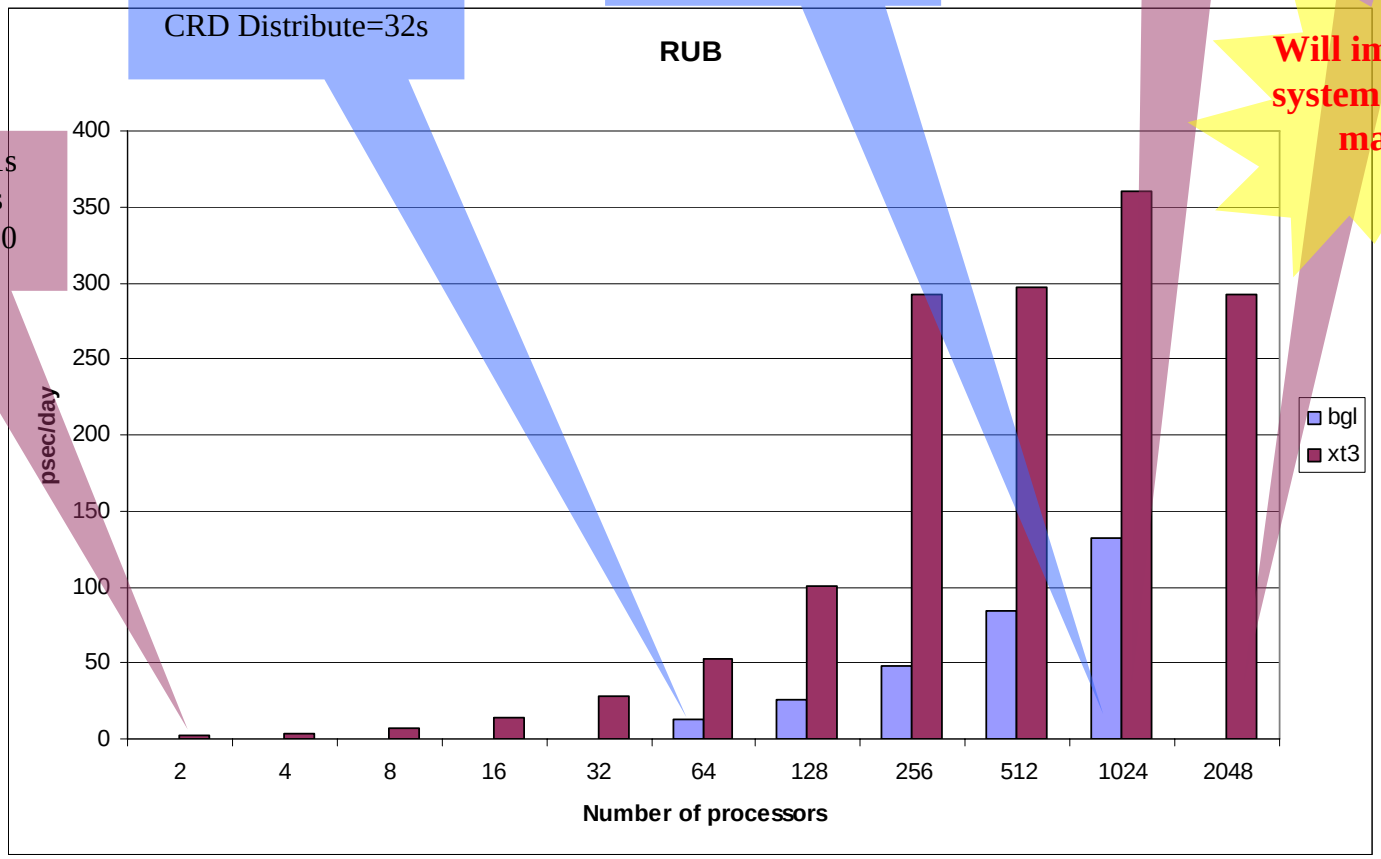
Gen Born=6701s
FRC Collect=52s
CRD Distribute=32s

Gen Born=398s
FRC Collect=142s
CRD Distribute=145s

Gen Born=107s
FRC Collect=61s
CRD Distribute=27s

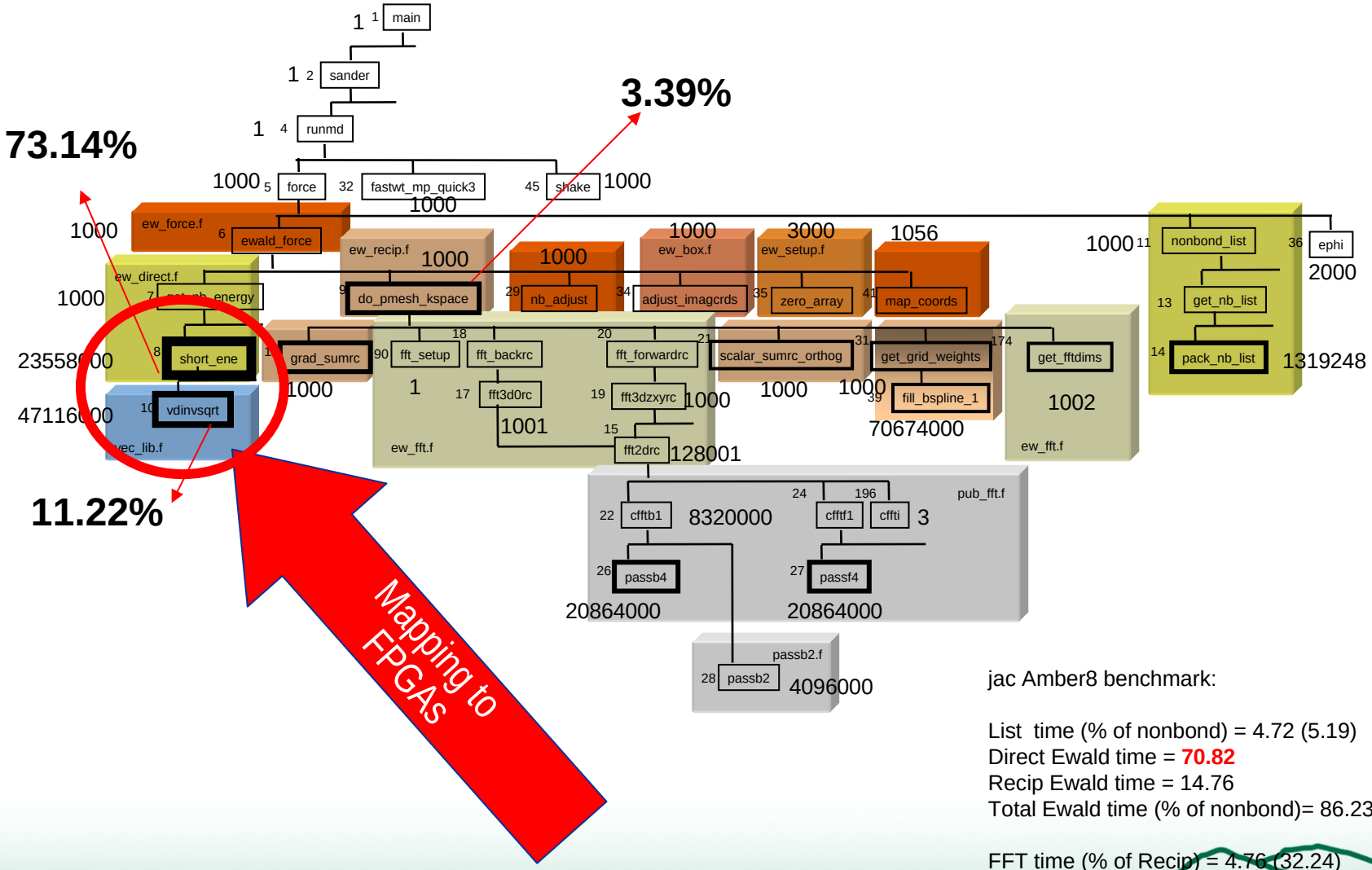
Gen Born=58s
FRC Collect=139s
CRD Distribute=54s

Will improve as system software matures



Rubisco with Generalized Born solvation method (ORNLtest3). Note that on BGL only results from 64, 128, 256, 512 nodes run are shown.

Mapping Amber Kernel to FPGAs



jac Amber8 benchmark:

List time (% of nonbond) = 4.72 (5.19)
Direct Ewald time = **70.82**
Recip Ewald time = 14.76
Total Ewald time (% of nonbond)= 86.23 (**94.81**)

FFT time (% of Recip) = 4.76 (32.24)

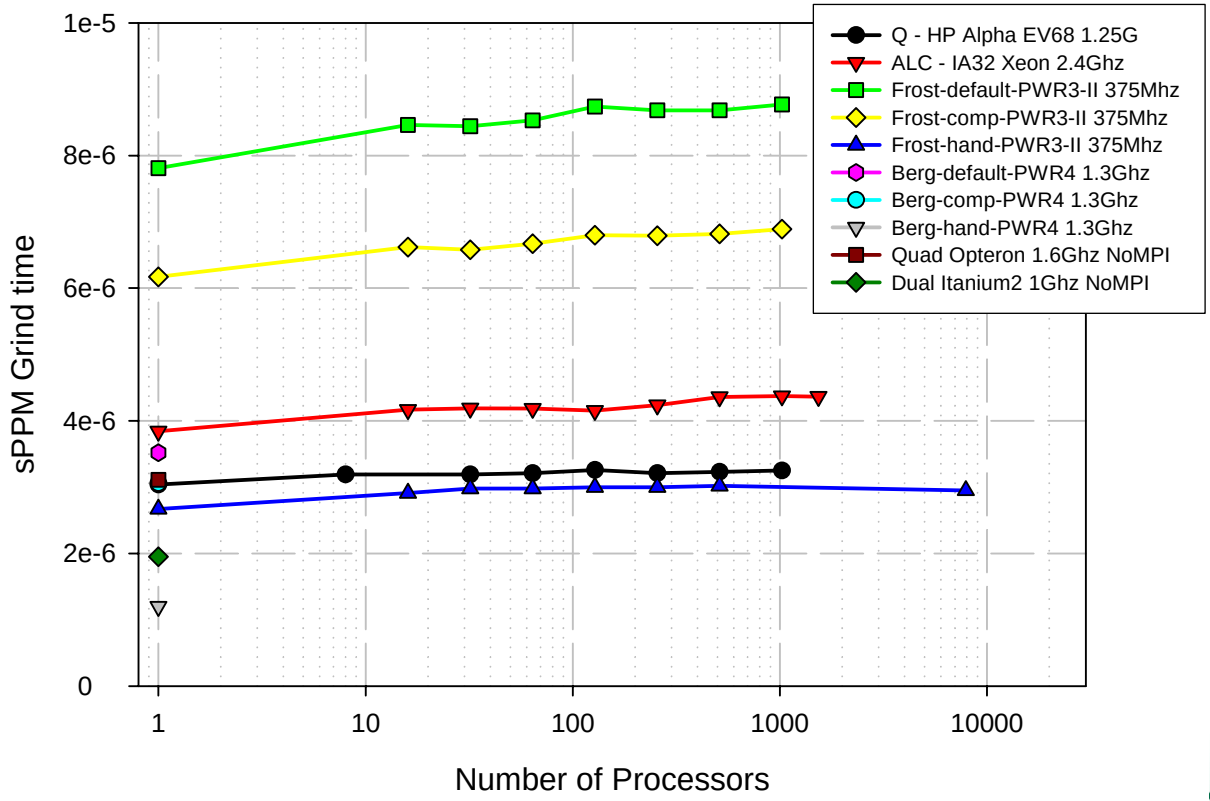
AMBER Summary

- Through performance analysis, we have identified the components that are limiting the scalability, and improved its scaling
- Amber scaling was limited to 128 nodes but now we have run experiments on 1024 nodes on Bluegene/L and on 2048 nodes on Cray XT3
- Achieved close to order of a nano-second/day on early evaluation stage supercomputing systems
- Mapping compute intensive kernel to SRC MapStation (a reconfigurable computing system)

Scalable Techniques for Performance Analysis, Evaluation, and Modeling

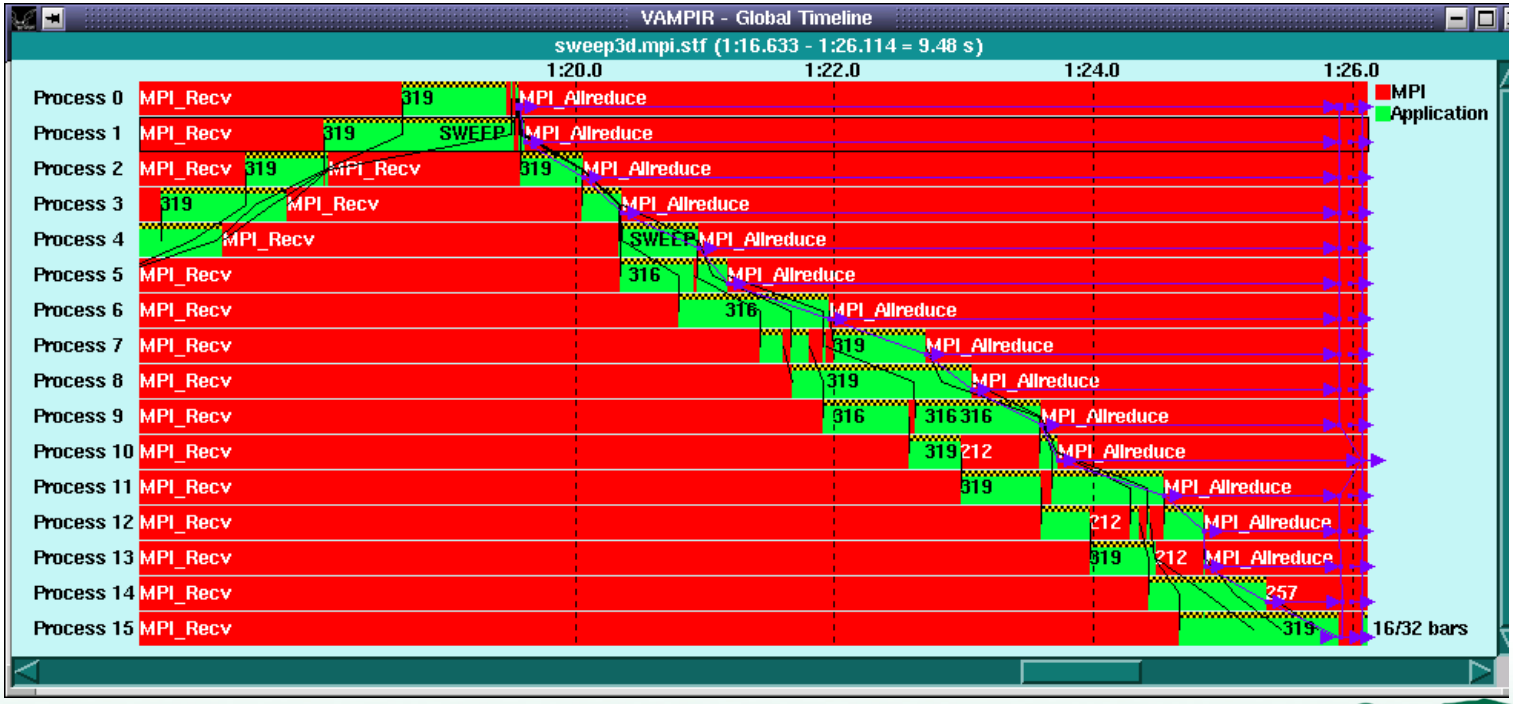
sPPM Across Architectures

⇨ We need more detailed information at scale

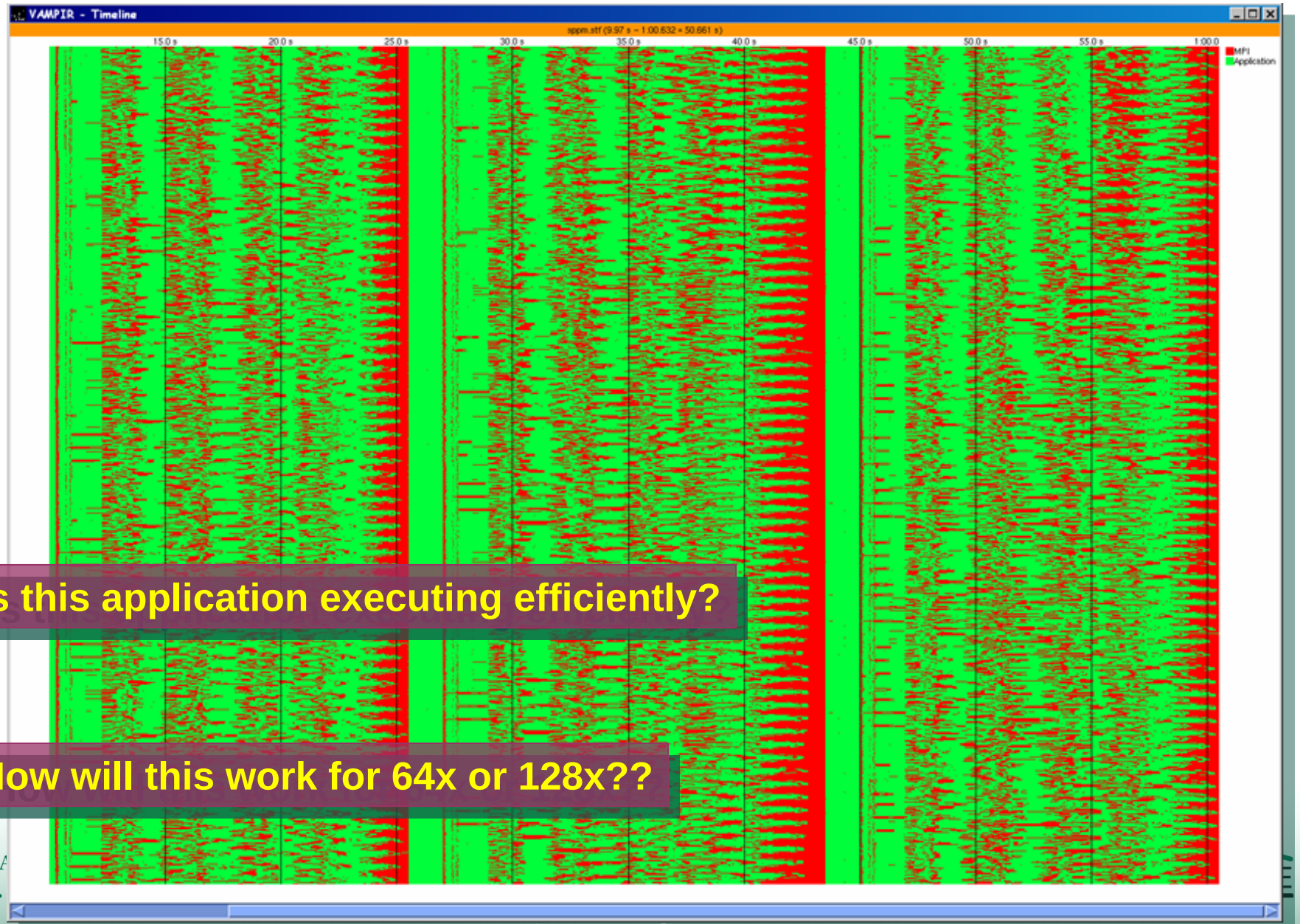


Traditional Performance Analysis of Communication Operations

- MPI's profiling layer promotes construction and portability of tools
- Many MPI tools use tracing
  - Produces very detailed information about communication activity



Timeline with 1024 tasks



Is this application executing efficiently?

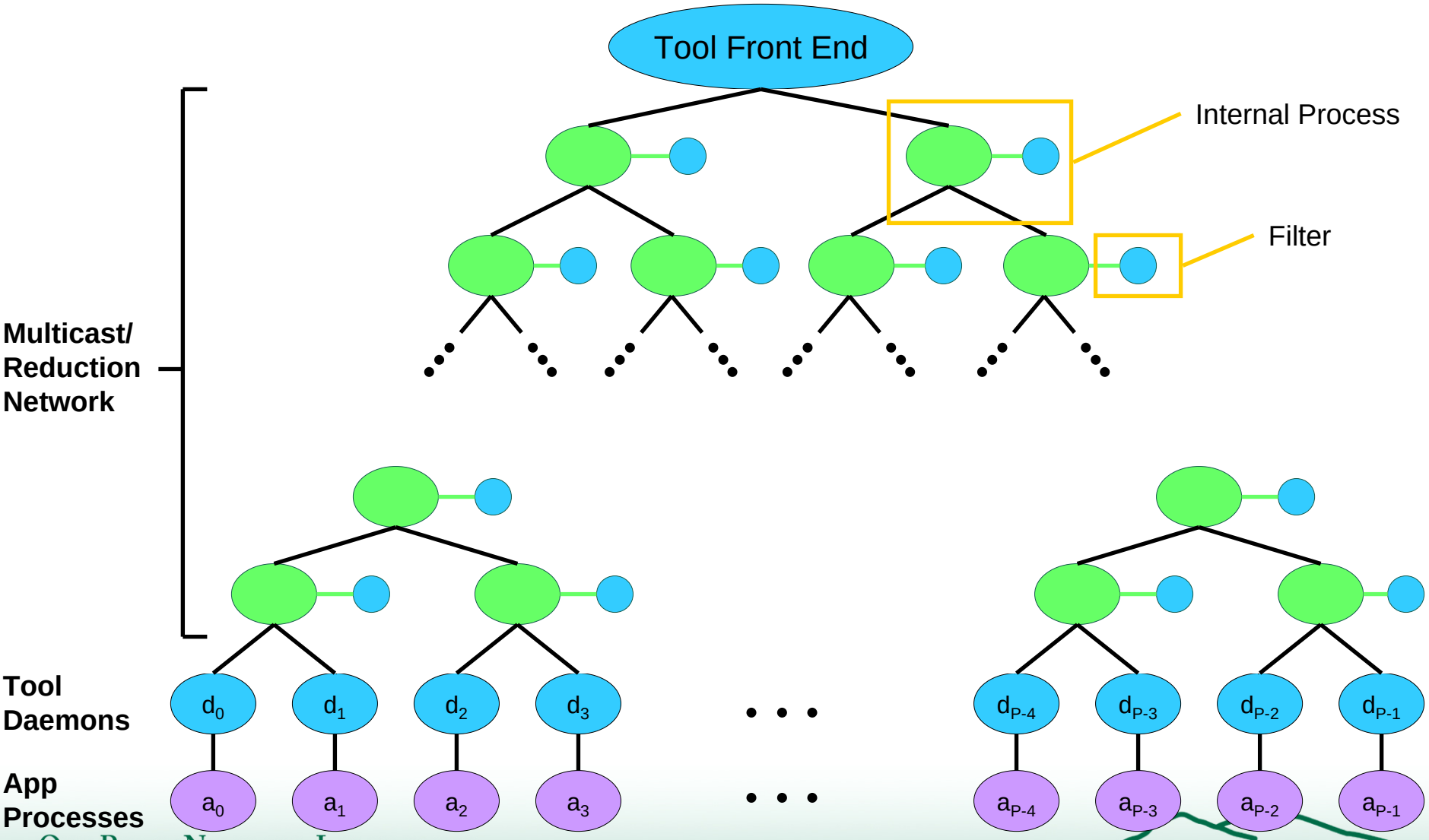
How will this work for 64x or 128x??

Scalability of Tools is Critical!

⇨ Several levels of performance analysis technology do not scale well

Concurrency	O(100)	O(1,000)	O(10,000)	O(100,000)
Instrumentation	OK	OK	OK	OK
Instrumentation management	OK	Hurdle	Hurdle /Barrier	Barrier
Data management	OK	Hurdle	Hurdle /Barrier	Barrier
Data interpretation	Hurdle	Barrier	Barrier	Barrier

Scalable Tool Instrumentation with MRNet

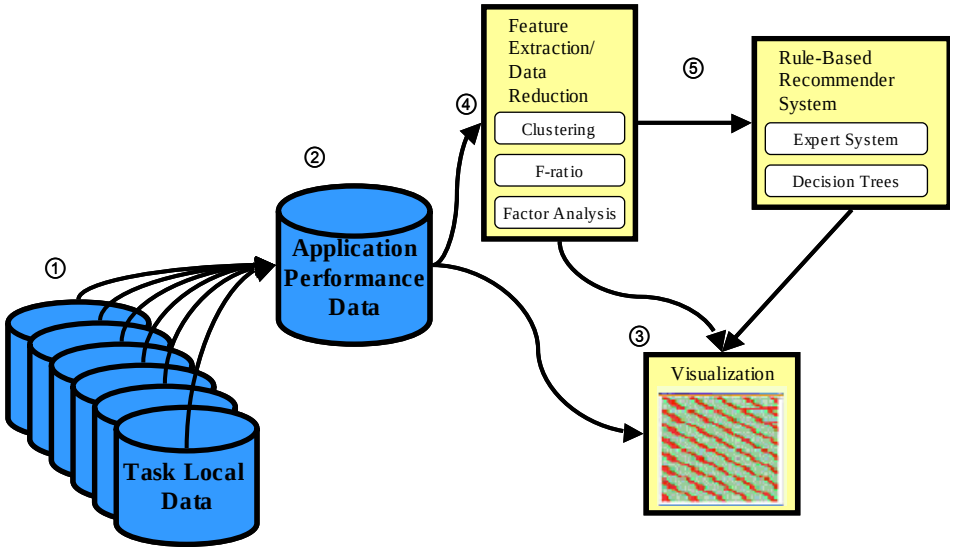


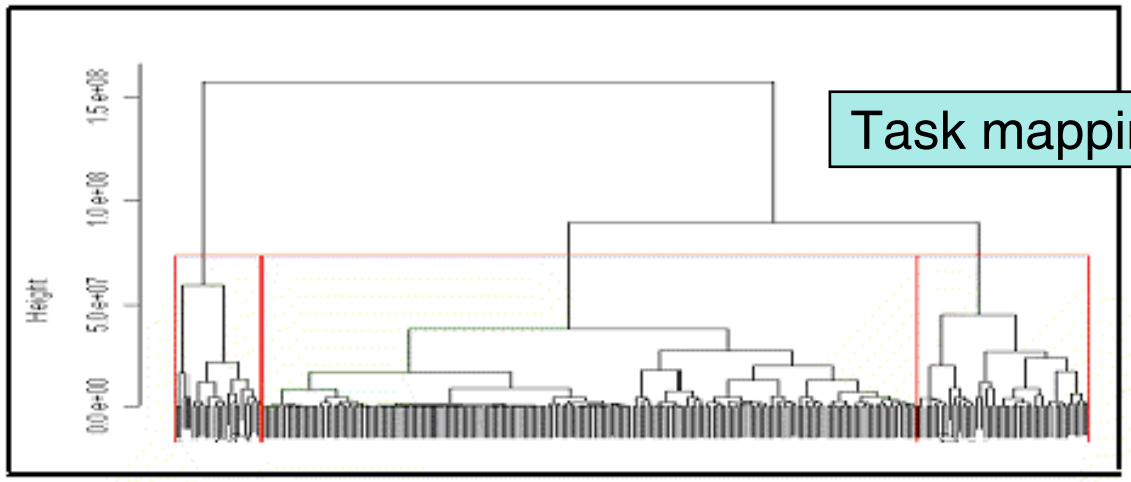
Currently scales to over 1000.

Work by Philip Roth.

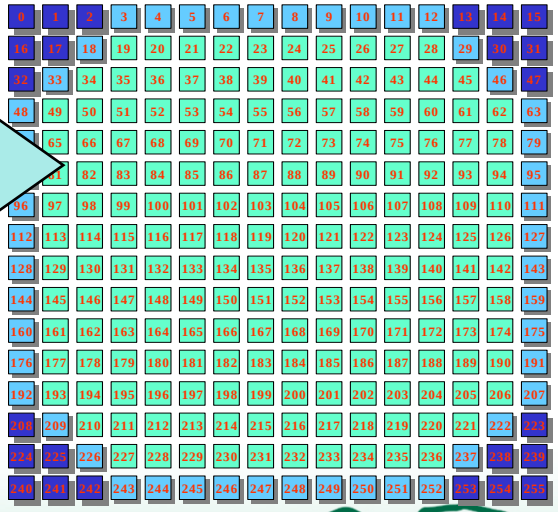
Multivariate Statistical Analysis of Hardware Counter Data

- Hardware counters produce huge amounts of data on large systems
- Multivariate statistical techniques help distill important features
- Clustering, Factor analysis, PCA





Task mapping



Joint work with Dong Ahn @ LLNL.

Automatic Classification for MPI Trace Analysis

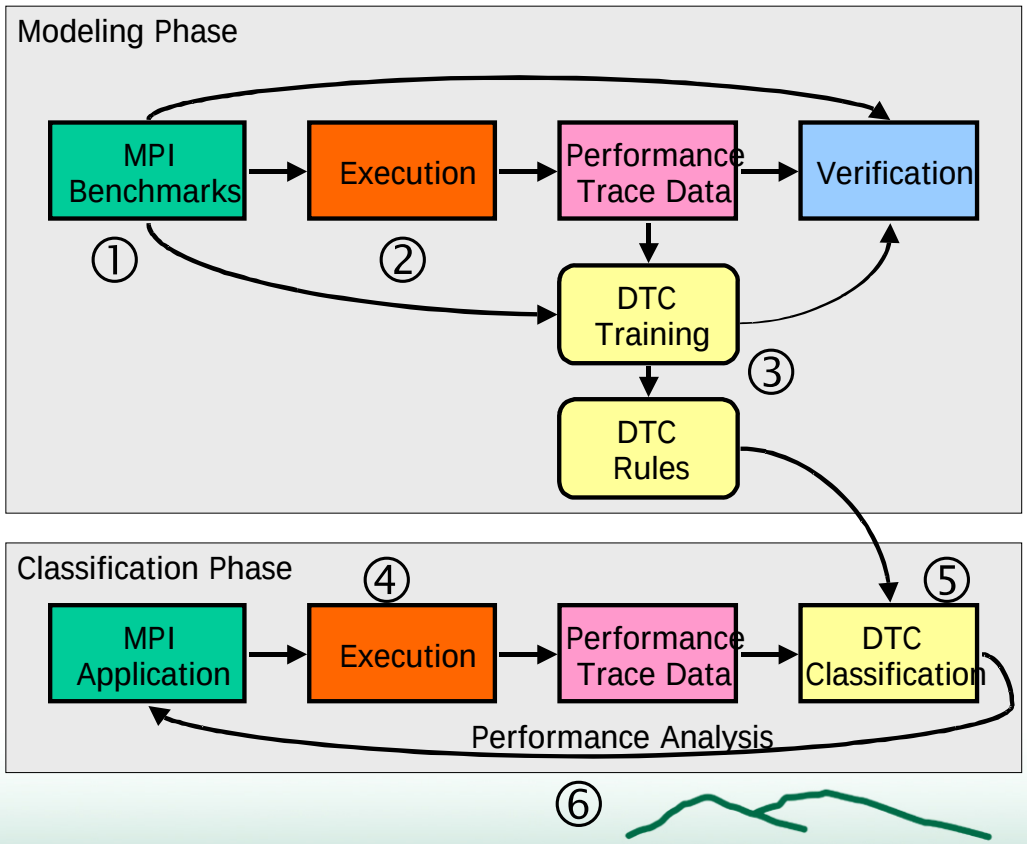
- ⇨ Use decision tree classification (a supervised learning technique) to classify application's messages automatically
- ⇨ Compare an application's message operations to 'normal' communication for a particular MPI configuration

➢Modeling Phase (once)

- –Use benchmarks to generate decision tree
- –Both efficient and inefficient

➢Classification Phase (many)

- –Execute application
- –Analyze application trace with classifier based on decision tree



Performance Engineering with Performance Assertions and Models

- Use performance assertions to verify the performance explicitly

```
pa_start(&pa, "$nFlops", PA_AEQ, "11 * %g * %g", &ym, &xm);
for (j=ys; j<ys+ym; j++) {
  for (i=xs; i<xs+xm; i++) {
    if (i == 0 || j == 0 || i == Mx-1 || j == My-1) {
      f[j][i] = x[j][i];
    } else {
      u       = x[j][i];
      uxx     = (two*u - x[j][i-1] - x[j][i+1])*hydhx;
      uyy     = (two*u - x[j-1][i] - x[j+1][i])*hxdhy;
      f[j][i] = uxx + uyy - sc*PetscExpScalar(u);
    }
  }
}
pa_end(pa);
PetscLogFlops(11*ym*xm);
```

- Expression
  - **"$nFlops", PA_AEQ, "11 * %g * %g", &ym, &xm**
  - Empirically measure number of floating point operations with instrumentation
  - Test approximate equality (±10%) to **'11 * ym * xm'** ?
- Empirical measurements verify performance model

Joint work with Pat Worley.

Aggressively Evaluate New Technologies

Recent and Ongoing Evaluations

- Cray X1
  - P.A. Agarwal, R.A. Alexander et al., "Cray X1 Evaluation Status Report," ORNL, Oak Ridge, TN, Technical Report ORNL/TM-2004/13, 2004.
  - T.H. Dunigan, Jr., M.R. Fahey et al., "Early Evaluation of the Cray X1," Proc. ACM/IEEE Conference High Performance Networking and Computing (SC03), 2003.
  - T.H. Dunigan, Jr., J.S. Vetter et al., "Performance Evaluation of the Cray X1 Distributed Shared Memory Architecture," IEEE Micro, 25(1):30-40, 2005.
- SGI Altix
  - T.H. Dunigan, Jr., J.S. Vetter, and P.H. Worley, "Performance Evaluation of the SGI Altix 3700," Proc. International Conf. Parallel Processing (ICPP), 2005.
- Cray XD1
  - M.R. Fahey, S.R. Alam et al., "Early Evaluation of the Cray XD1," Proc. Cray User Group Meeting, 2005, pp. 12.
- SRC Mapstation
  - M.C. Smith, J.S. Vetter, and X. Liang, "Accelerating Scientific Applications with the SRC-6 Reconfigurable Computer: Methodologies and Analysis," Proc. Reconfigurable Architectures Workshop (RAW), 2005.
- Underway
  - XD1 FPGAs
  - ClearSpeed
  - EnLight
  - Multicore processors
  - IBM BlueGene/L

Summary

- Performance analysis, evaluation, and modeling will become increasingly challenging over the next decade due to uncertainty from several factors
  - Scale
  - Architecture complexity
  - Application complexity
- We must embrace the uncertainty by
  - Understanding our applications and architectures in detail with empirical measurement, models
  - Working with apps teams to adapt to new technology
  - Developing performance engineering techniques that provide insight
  - Aggressively evaluating new technologies

Acknowledgements and More Info

- This research was sponsored by the Office of Mathematical, Information, and Computational Sciences, Office of Science, U.S. Department of Energy under Contract No. DE-AC05-00OR22725 with UT-Batelle, LLC. Accordingly, the U.S. Government retains a non-exclusive, royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for U.S. Government purposes.

- http://www.csm.ornl.gov/ft
- http://www.csm.ornl.gov/evaluation
- Email: vetterjs@ornl.gov