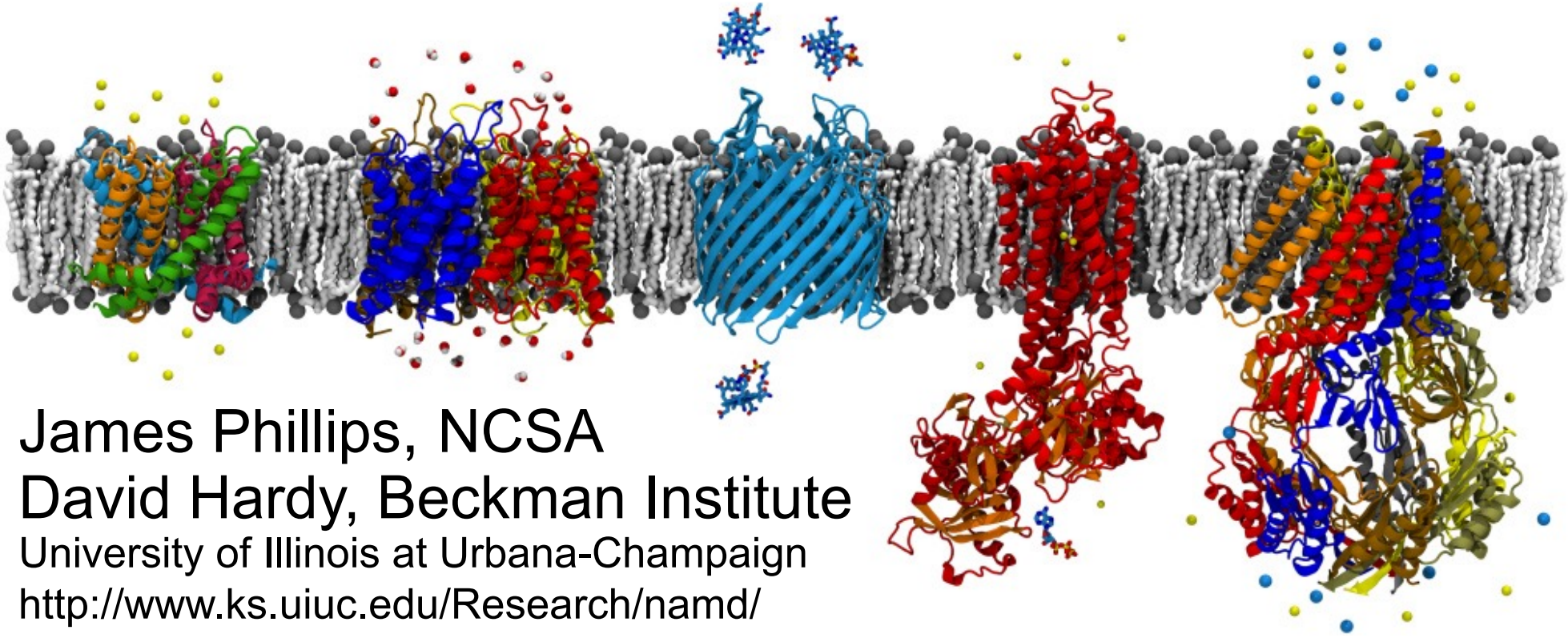
Experience with NAMD and Charm++ on Summit



James Phillips, NCSA
David Hardy, Beckman Institute
University of Illinois at Urbana-Champaign
http://www.ks.uiuc.edu/Research/namd/

NIH Biomedical Technology Research Center for Macromolecular Modeling and Bioinformatics

Developers of the widely used computational biology software **VMD** and **NAMD**

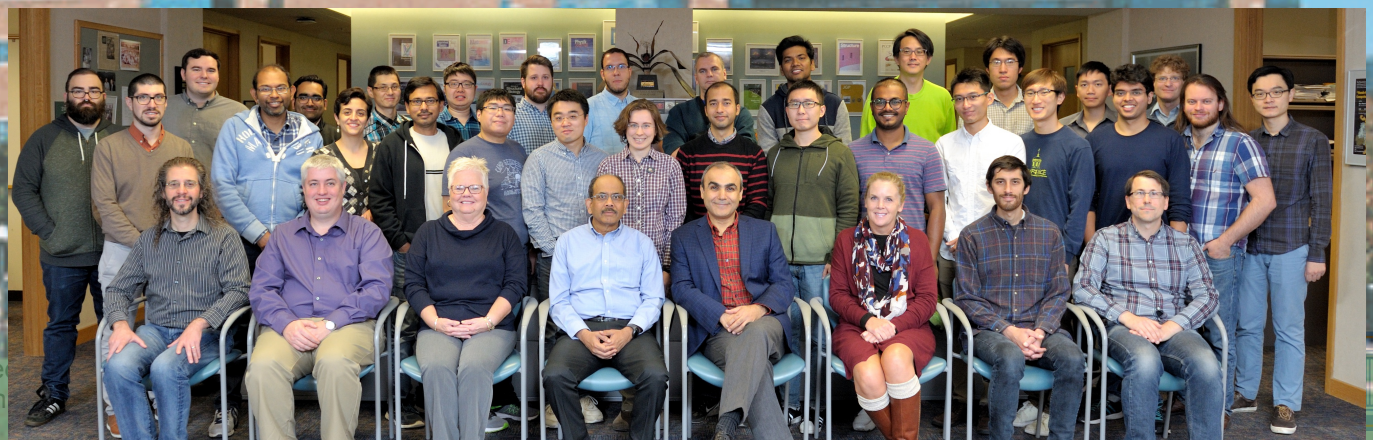
250,000 registered **VMD** users
80,000 registered **NAMD** users

600 publications (since 1972)
over **54,000** citations

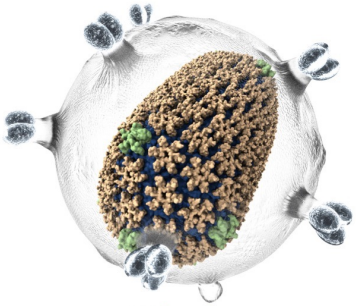
4 faculty members
8 developers
1 systems administrator
17 postdocs
46 graduate students
2 administrative staff

Perfect score (10.0) on 2017-2022 NIH renewal

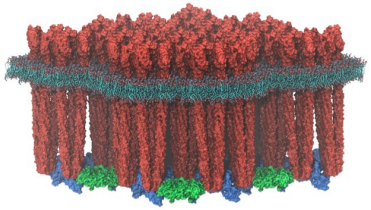
research projects include: virus capsids, bacteria, molecular motors, neurons and synapses, membrane transporters, bioenergetic membranes



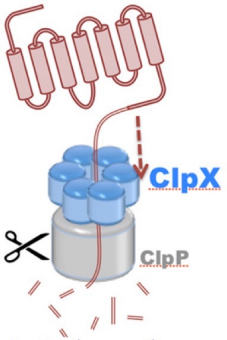
NIH Center Driving Projects 2017-2022



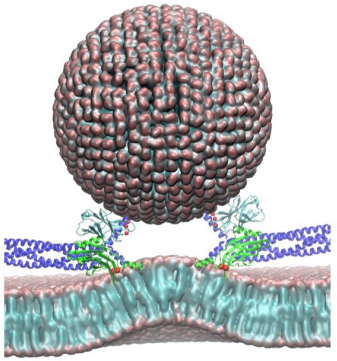
Viral Infection



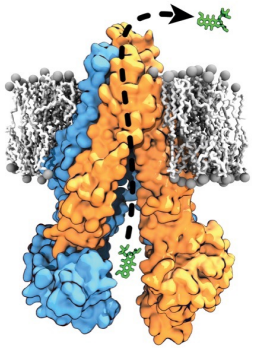
Symbiont Bacteria



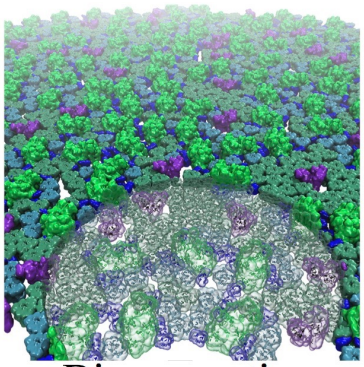
Molecular Motors



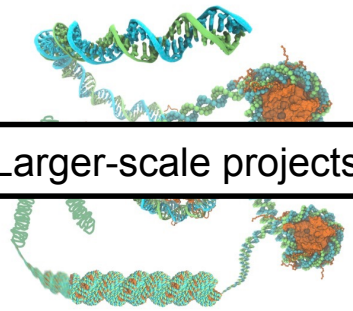
Neurons and Synapses



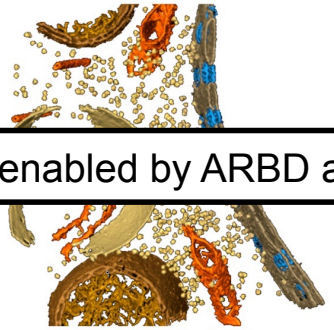
Membrane Transporters



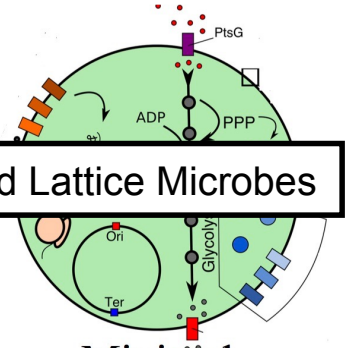
Bioenergetic Membranes



Chromatin



Bacterial & Eukaryal Systems



Minimal Cell

Larger-scale projects enabled by ARBD and Lattice Microbes

NAMD: Practical Supercomputing for Biomedical Research

"**widest-used application**" on NCSA Blue Waters, NSF-specified benchmark for successor machine

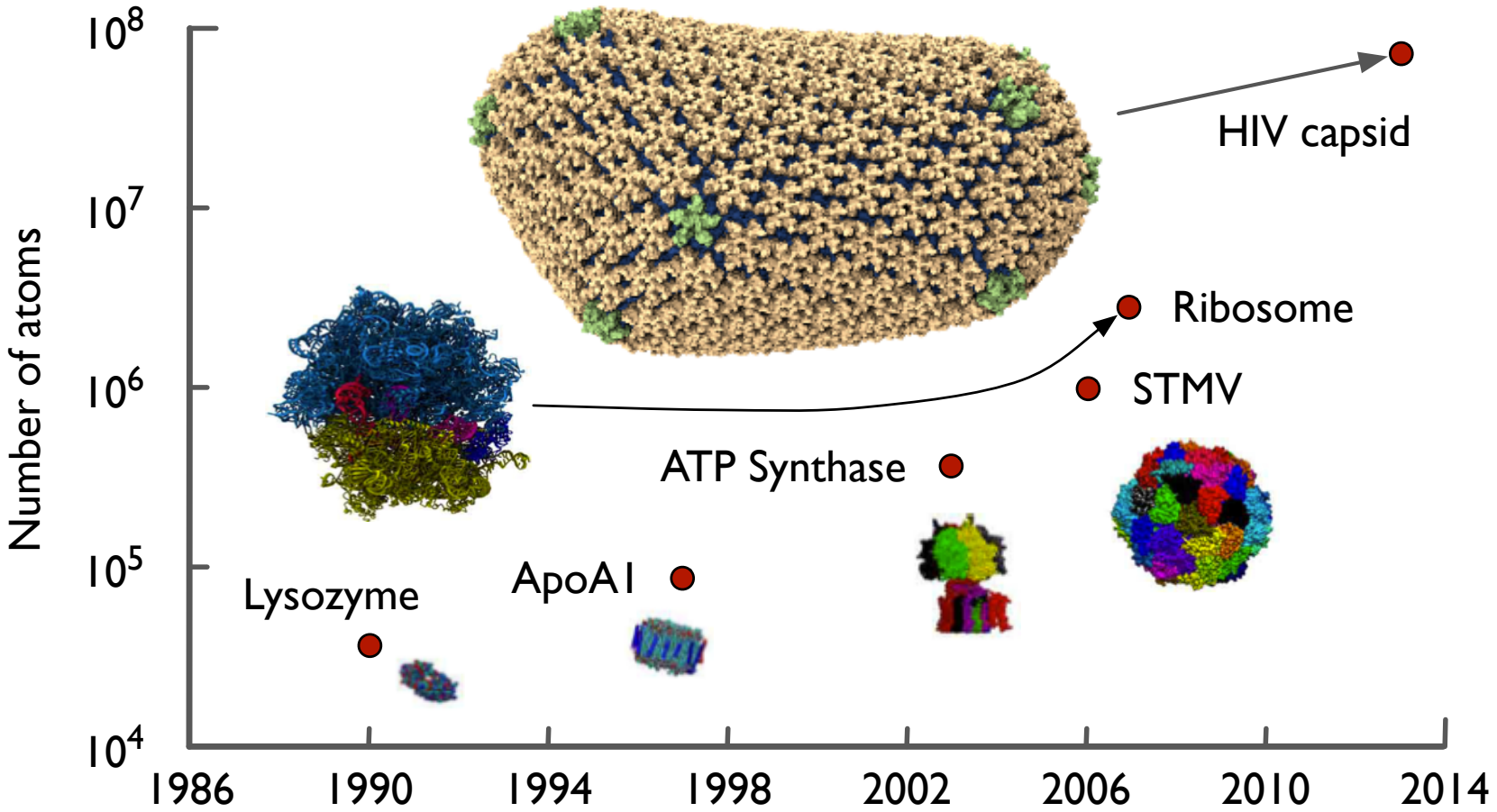
"**by a very large margin the most used code**" at Texas Advanced Computing Center (2nd largest)

Early adopters of workstation clusters (1993), Linux clusters (1998), and CUDA (**2007**).

Application readiness/early science projects on
- Argonne Theta (10 PF Cray KNL, completed)
- Oak Ridge Summit (200 PF Power9/Volta, 2018)
- ~~Argonne Aurora (200 PF Cray KNH, 2019)~~
- Argonne Aurora (1 EF Intel ???, 2021)

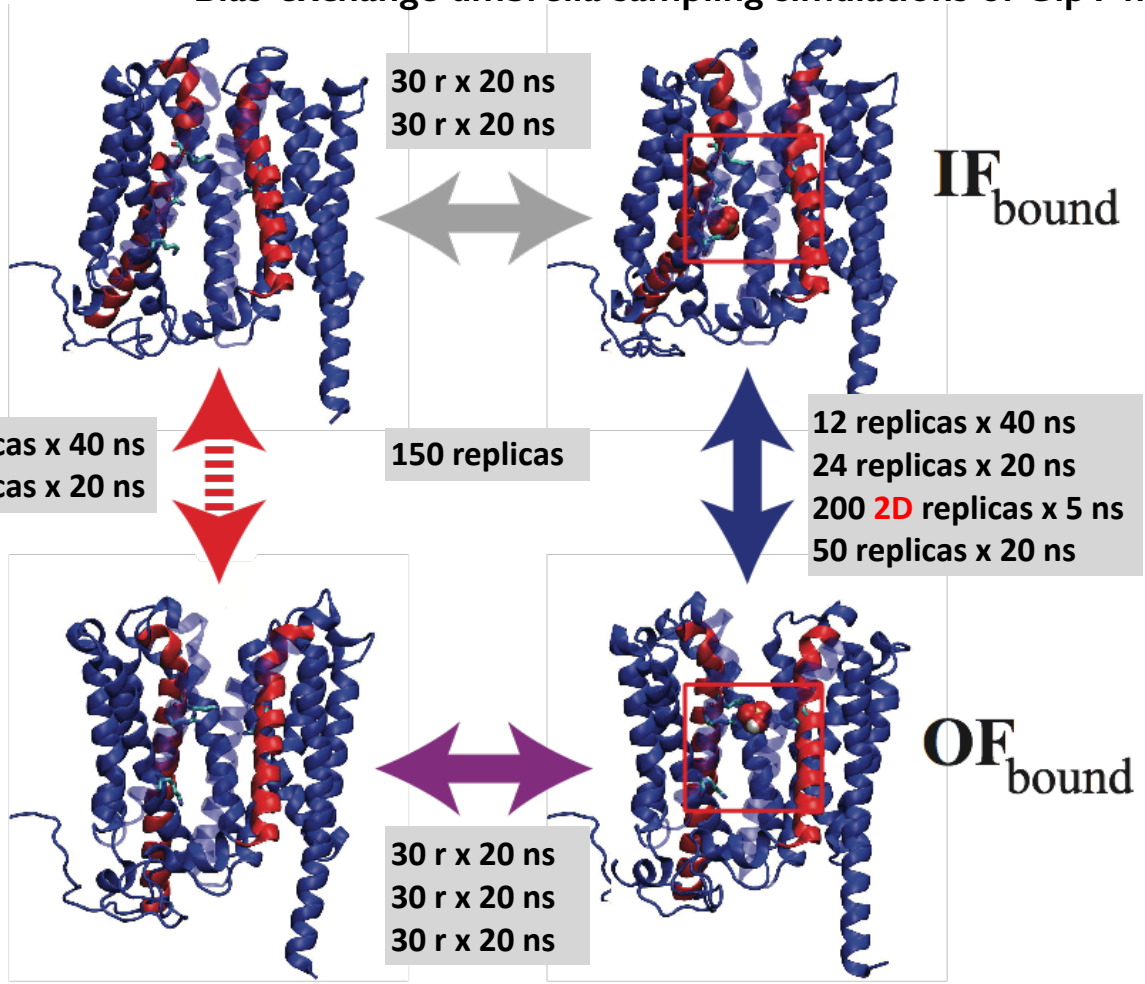
"For outstanding contributions to the development of widely used parallel software for large biomolecular systems simulation"

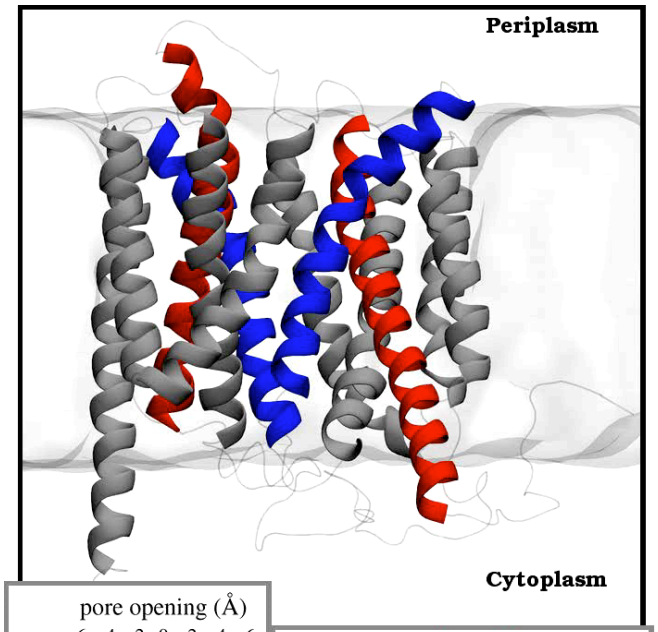
Need for petascale: Simulation follows structural discovery

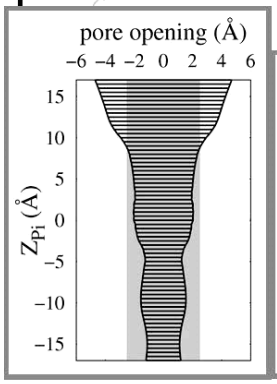


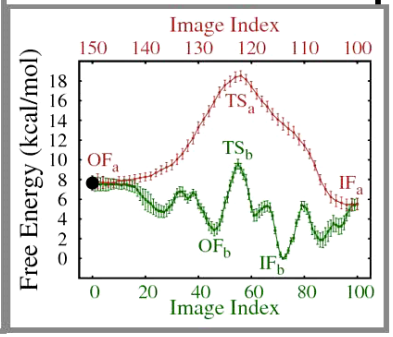
Multi-copy methodologies enable study of millisecond processes

Bias-exchange umbrella sampling simulations of GlpT membrane transporters



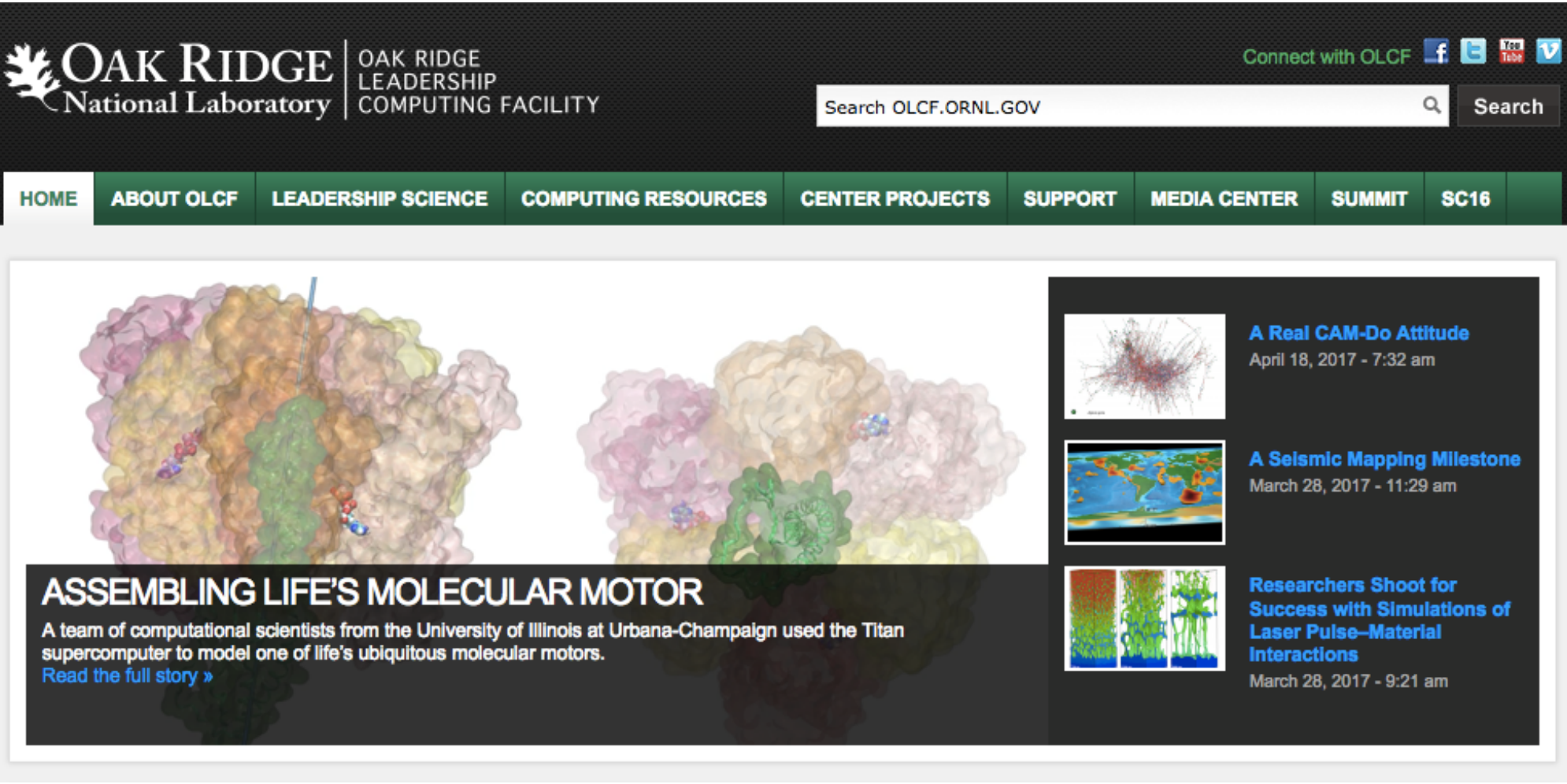






M. Moradi, G. Enkavi, and E. Tajkhorshid, *Nature Communications* **6**, 8393 (2015)

Long Timescale in a Large System



https://www.olcf.ornl.gov/2017/05/09/assembling-lifes-molecular-motor/

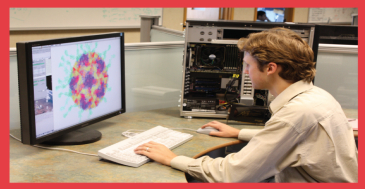
GPUs are critical for visualization and analysis

Large memory GPU-accelerated remote visualization must be ***embedded at supercomputer centers***.
Available now! See bluewaters.ncsa.illinois.edu/dcv

Storage

Compute

Visualization



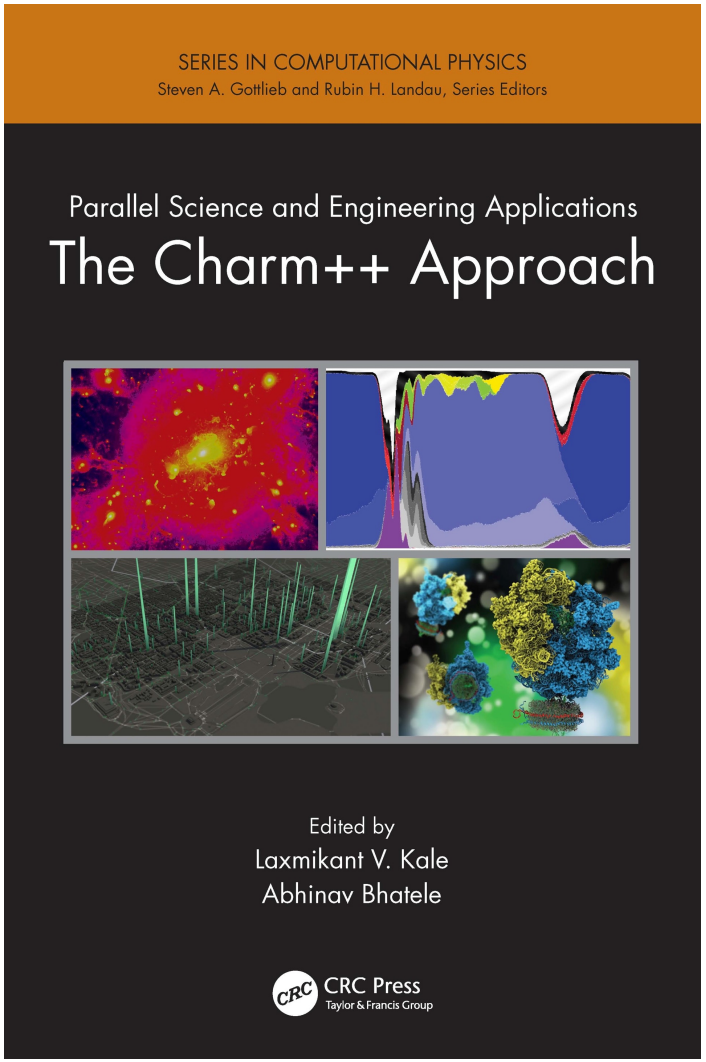
Compressed Video

1 Gigabit Network

NAMD is based on Charm++

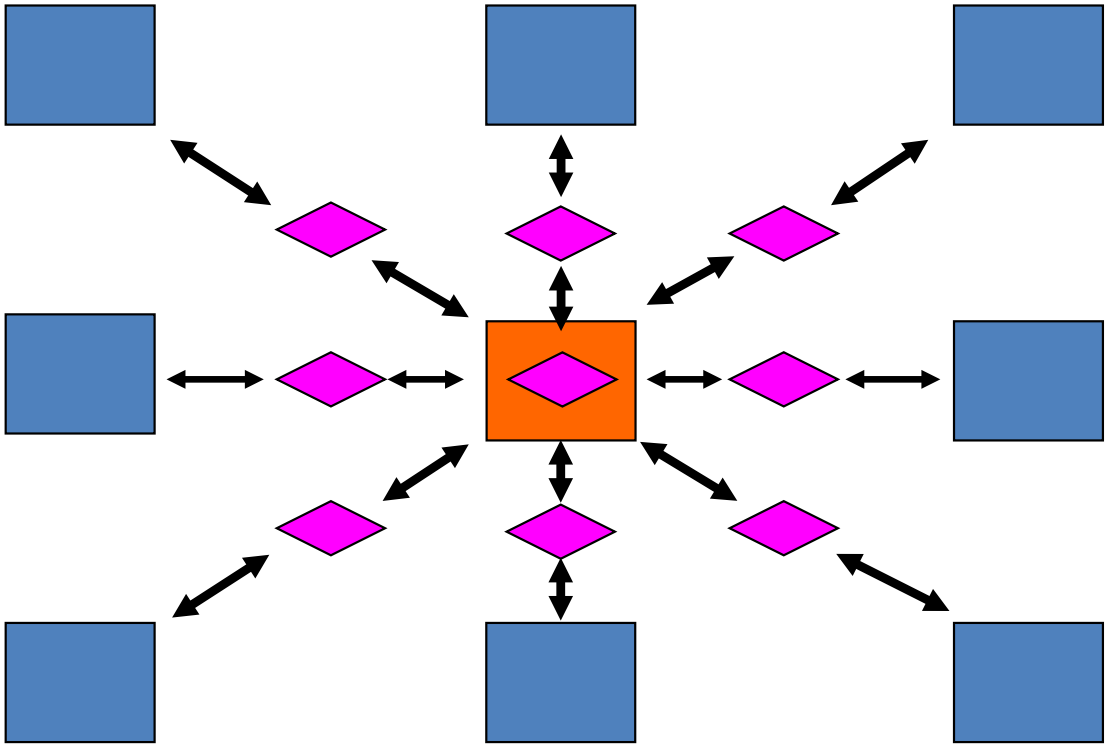
- Parallel C++ with *data driven* objects.
- Asynchronous method invocation.
- Prioritized scheduling of messages/execution.
- Measurement-based load balancing.
- Portable messaging layer.

Complete info at charmplusplus.org and charm.cs.illinois.edu



NAMD Hybrid Decomposition

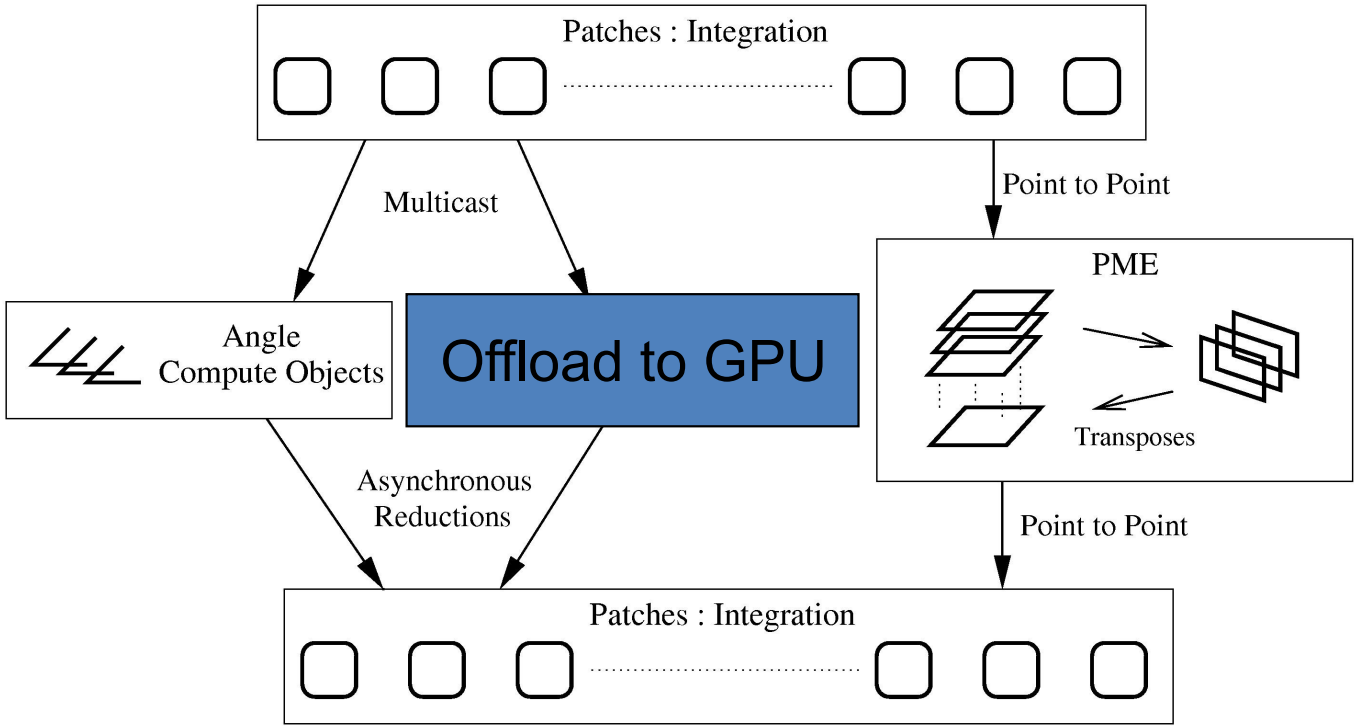
Kale et al., *J. Comp. Phys.* 151:283-312, 1999



- Spatially decompose data and communication
- Separate but related work decomposition
- **"Compute objects" create much greater amount of parallelization**, facilitating iterative, measurement-based load balancing system, all from **use of Charm++**

Overlap Calculations, Offload Nonbonded Forces

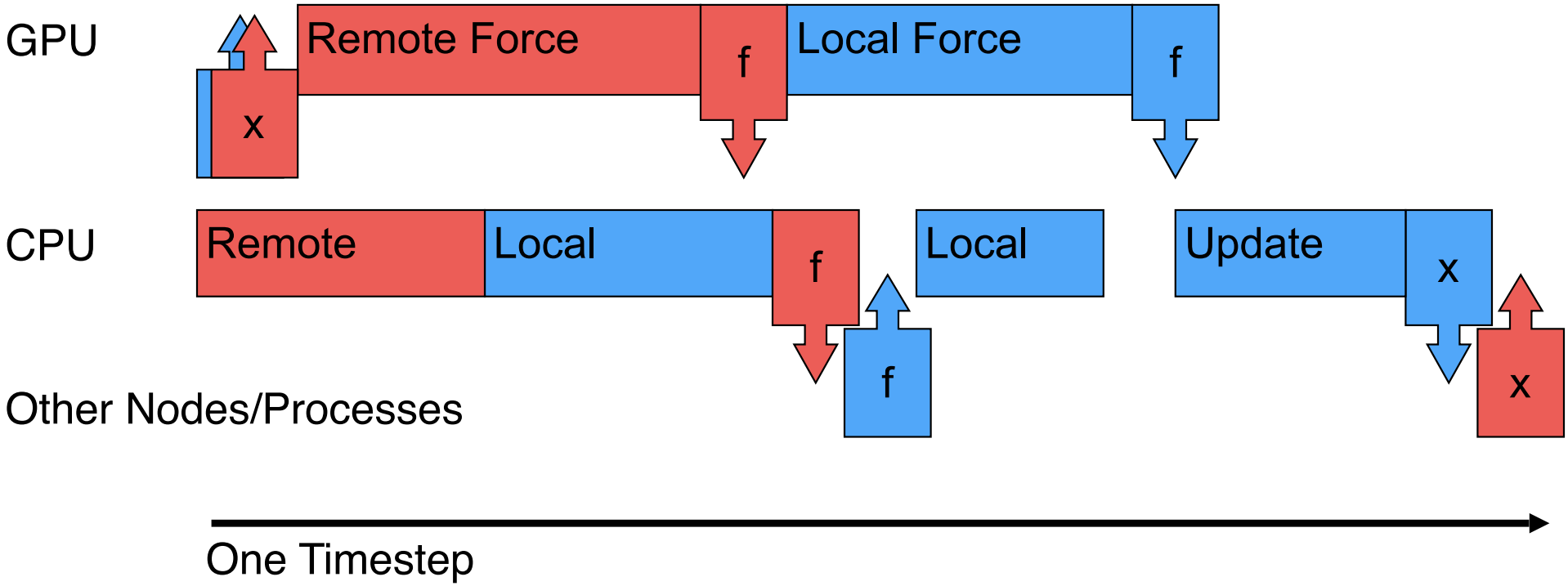
Phillips et al., *SC2002*



Objects are assigned to processors and queued as data arrives

Reduce Communication Latency by Separating Work Units

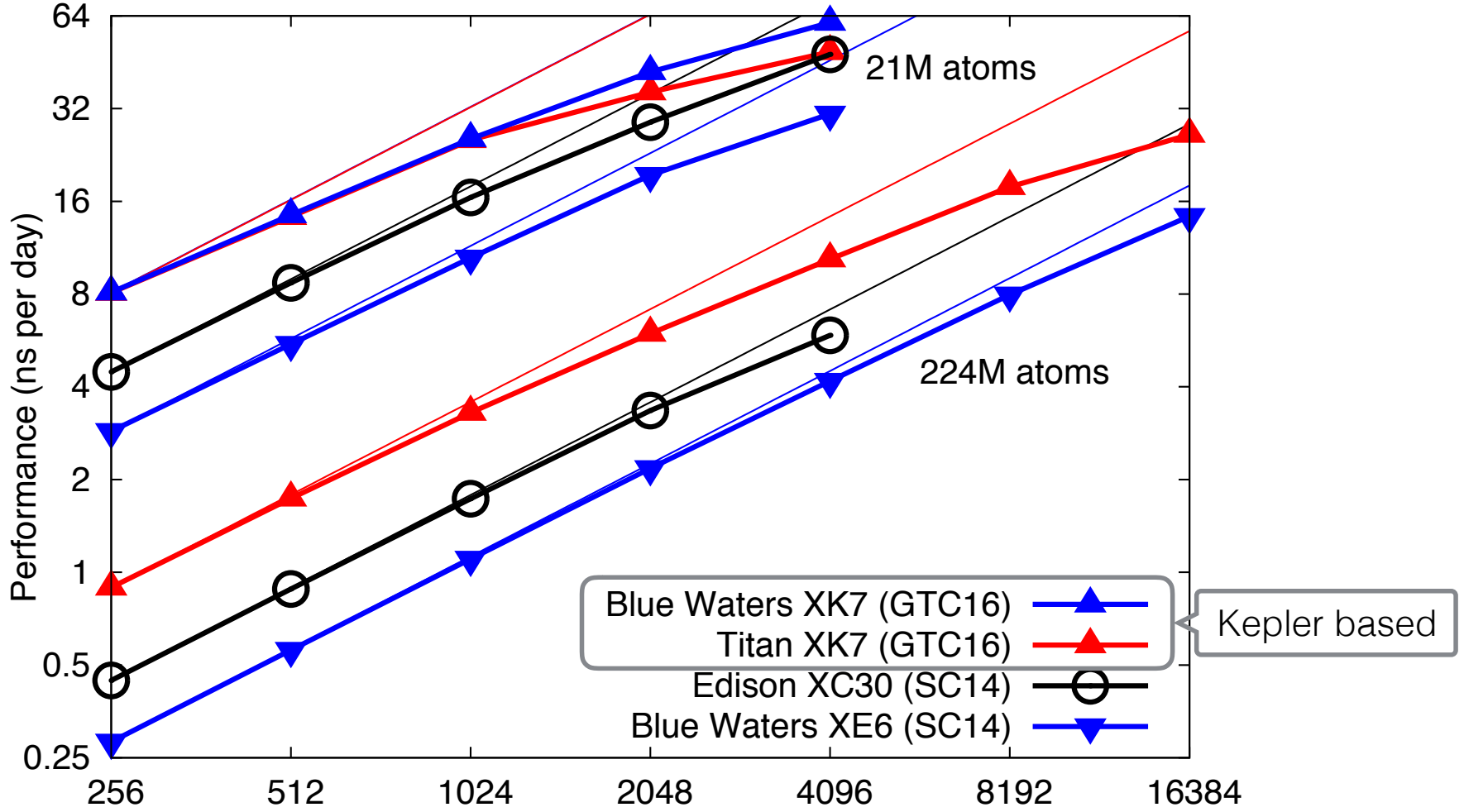
Phillips et al., SC2008



Early GPU Fits Into Parallel NAMD as Coprocessor

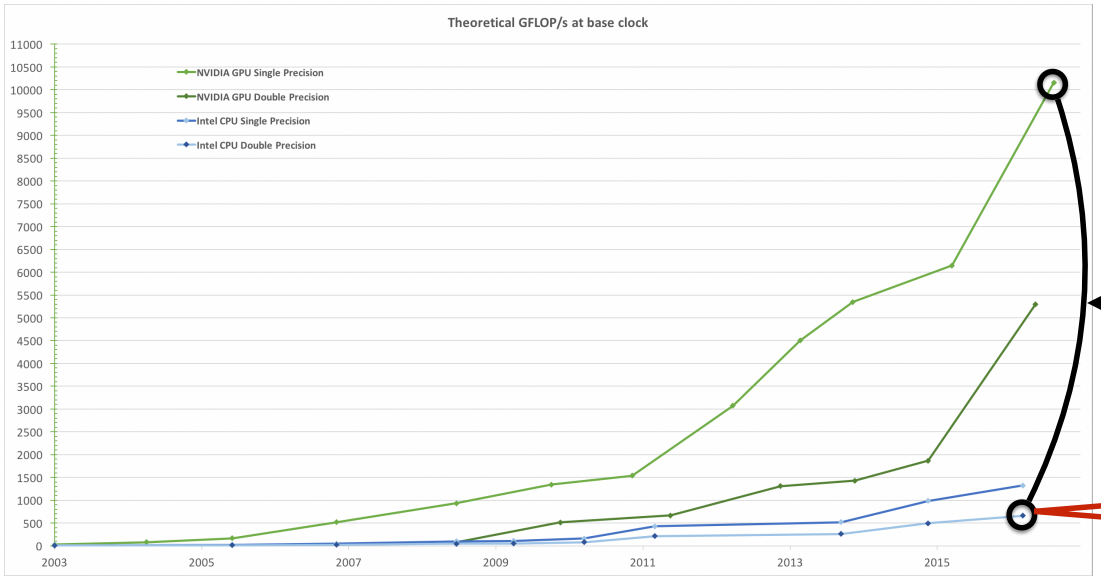
- Offload most expensive calculation: non-bonded forces
- Fits into existing parallelization
- **Extends existing code without modifying core data structures**
- Requires work aggregation and kernel scheduling considerations to optimize remote communication
- **GPU is treated as a coprocessor**

NAMD Scales Well on Kepler Based Computers



(2fs timestep)

Challenge: GPUs Continue to Outpace CPUs

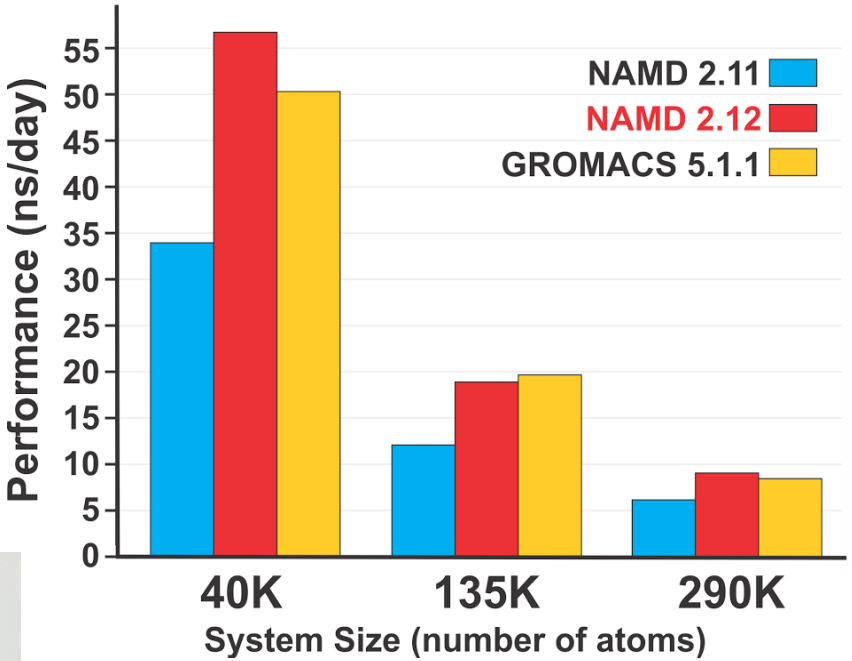


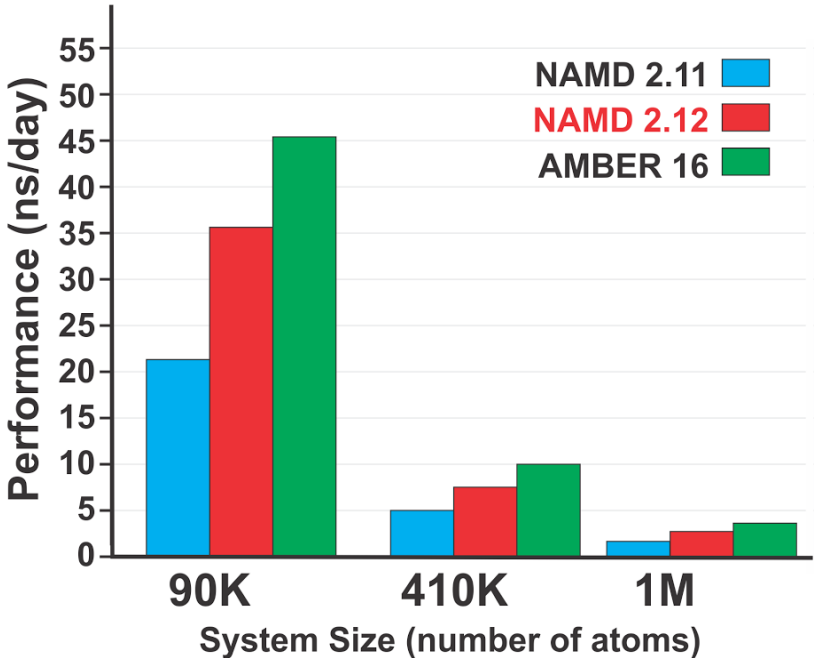
20x FLOP rate difference between GPU and CPU

Requires full use of CPU cores and vectorization!

- Balance between GPU and CPU capability keeps shifting towards GPU
- NVIDIA plots show only through Pascal — Volta widens the performance gap!
- Difference made worse by multiple GPUs per CPU (e.g. AWS, DGX, Summit)
- Past efforts to balance work between GPU and CPU are **now CPU bound**

Single-Node GPU Performance Optimization





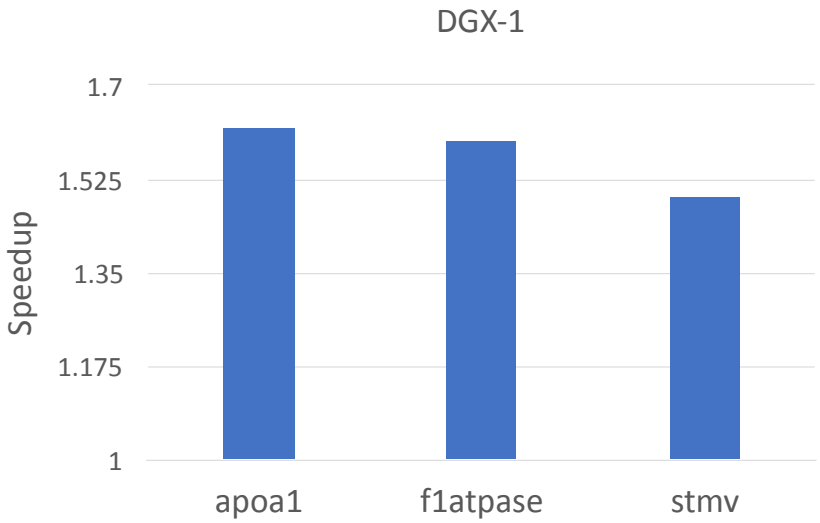
New kernels by **Antti-Pekka Hynninen**, formerly Oak Ridge, **NVIDIA**.

Stone, Hynninen, et al., *International Workshop on OpenPOWER for HPC (IWOPH'16)*, 2016.

Described at GTC 2016 **S6623** - Advances in NAMD GPU Performance

More Improvement from Offloading Bonded Forces

- GPU offloading for bonds, angles, dihedrals, impropers, exclusions, and crossterms
- Computation in single precision
- Forces are accumulated in 24.40 fixed point
- Virials are accumulated in 34.30 fixed point
- Code path exists for double precision accumulation on Pascal and newer GPUs
- **Reduces CPU workload and hence improves performance on GPU-heavy systems**



New kernels by **Antti-Pekka Hynninen, NVIDIA**

Supercomputers Increasing GPU to CPU Ratio

Blue Waters, Titan with Cray XK7 nodes
1 K20 / 16-core AMD Opteron

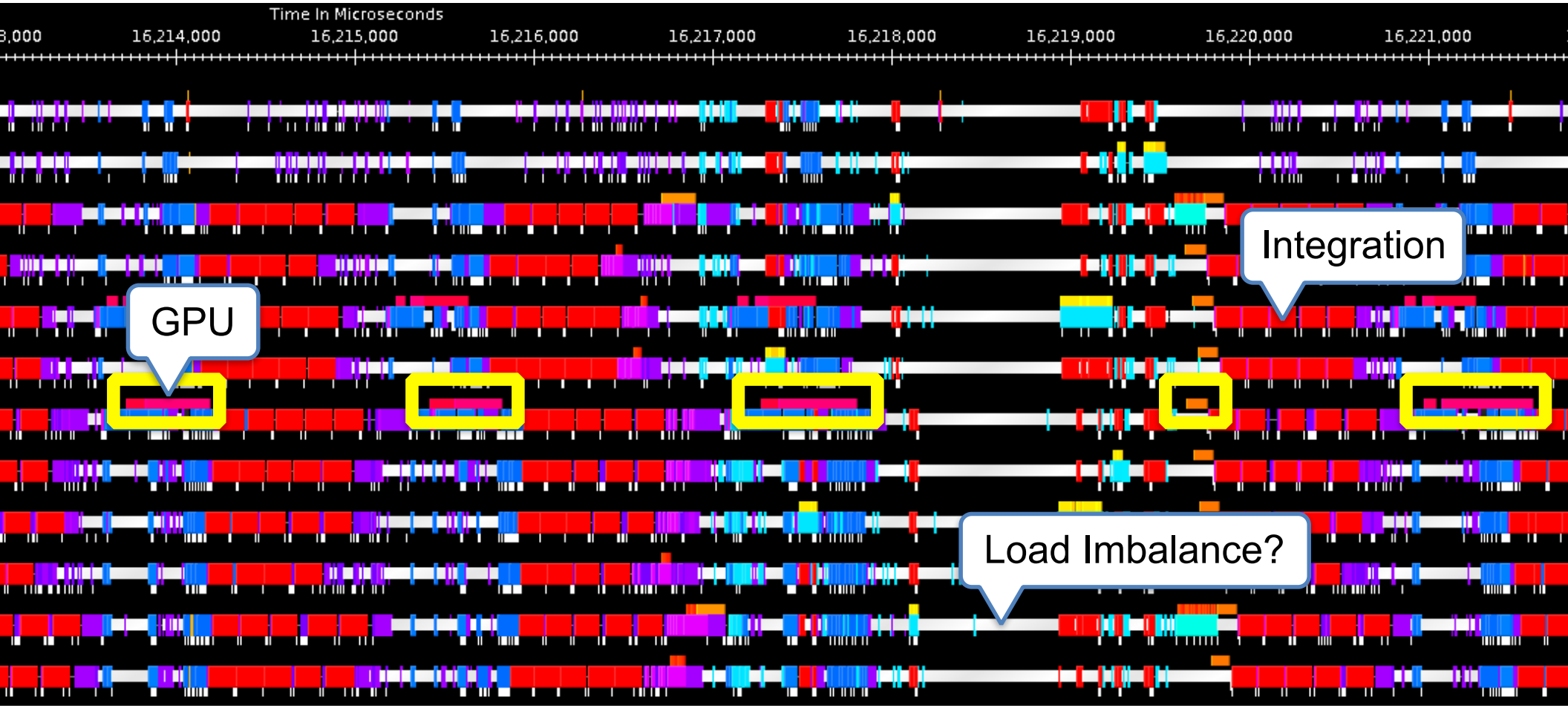
Summit nodes
6 Volta / 42 cores IBM Power 9

⇨ Only 7 cores supporting each Volta!

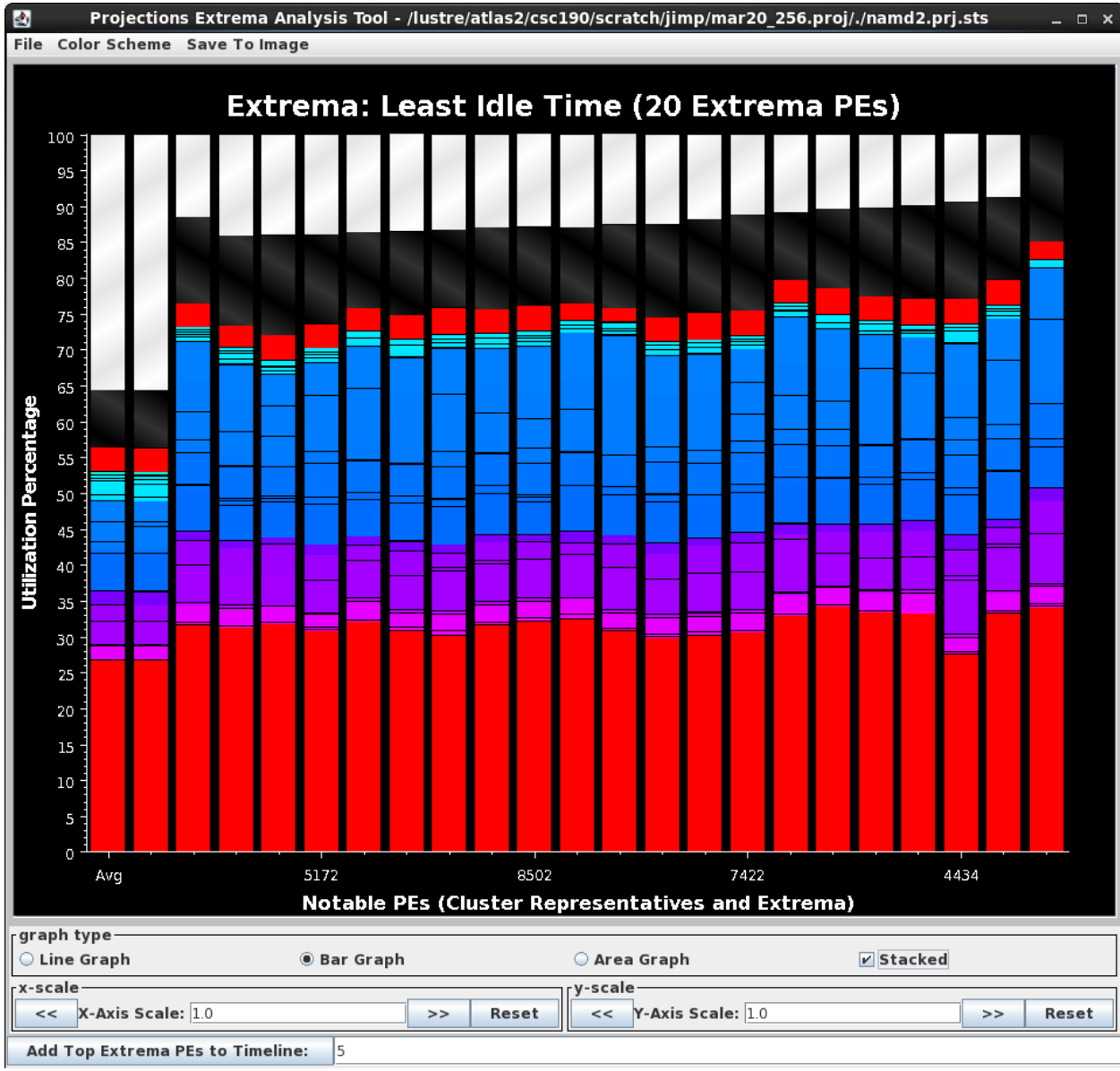
Running Charm++/NAMD on Summit

- IBM PAMI SMP machine layer provided by Charm++ runtime system
  - 30% better performance compared to MPI-based Charm++
  - No dedicated communication thread
- Single GPU per process (6 processes per node, 6 threads per process)
  - Leaving one core free per resource set seems to reduce noise
  - One core per socket is reserved by jsrun, so 8 unused cores per node
- With thread to core affinity:
  - jsrun -r6 -g1 -c7 namd2 +ignoresharing +ppn 6 +pemap 4-27:4,32-55:4,60-83:4,92-115:4,120-143:4,148-171:4
- Or without (expected to run slower, but sometimes faster):
  - jsrun --bind rs -r6 -g1 -c7 namd2 +ignoresharing +ppn 6

Charm++ *Projections* tool shows bottleneck

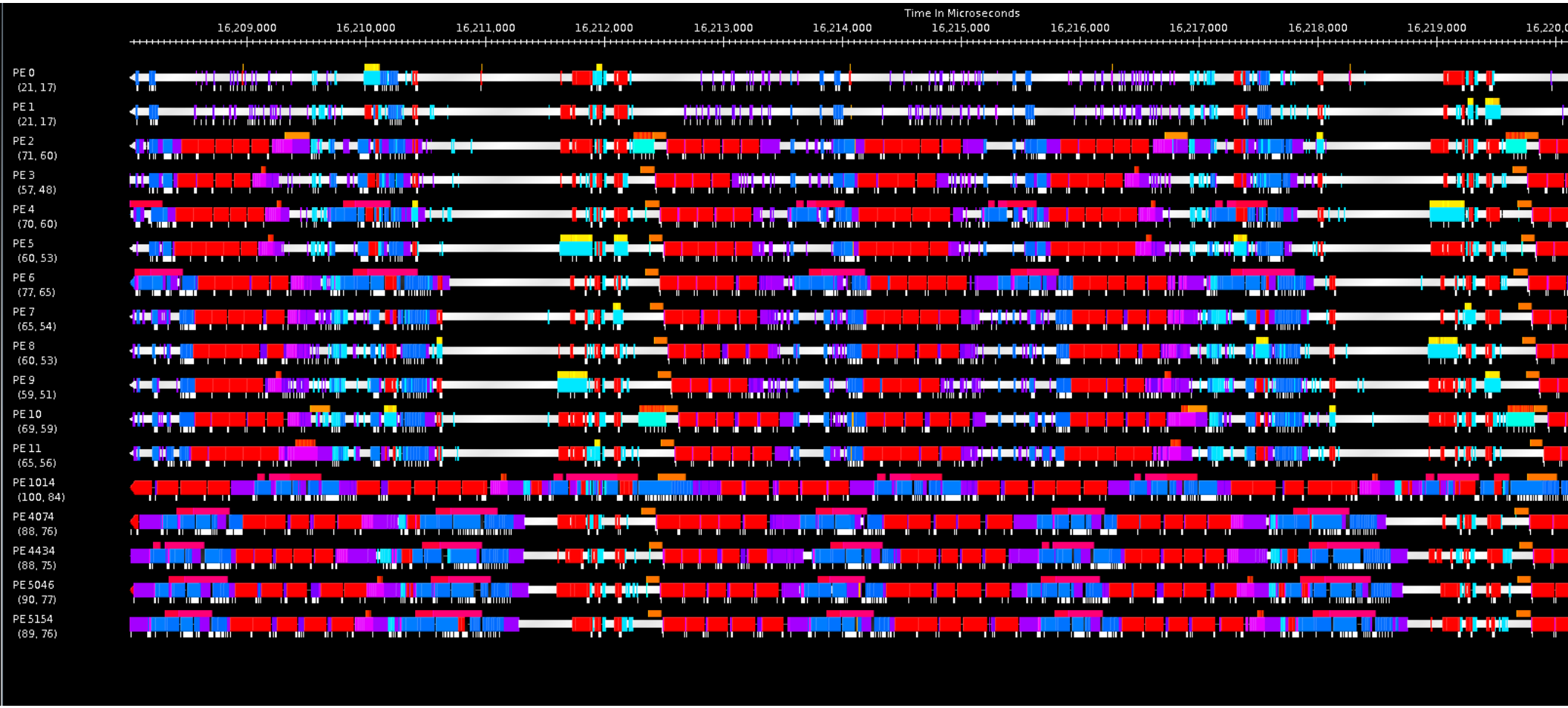


Charm++ *Projections* Extrema Tool Finds Problem PEs



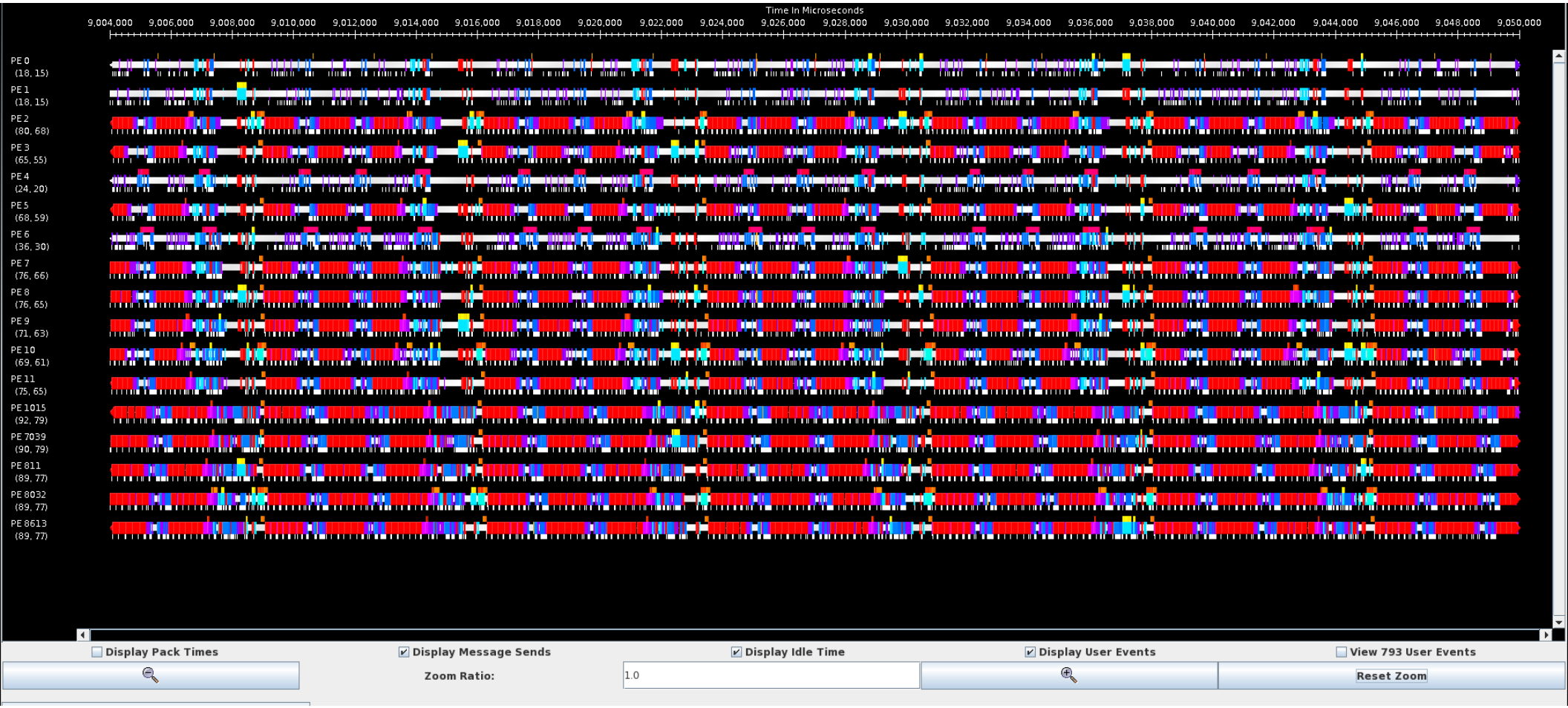
One PE has no idle time!

Also, overloaded PEs are all GPU hosts



Try removing patches from GPU host PEs

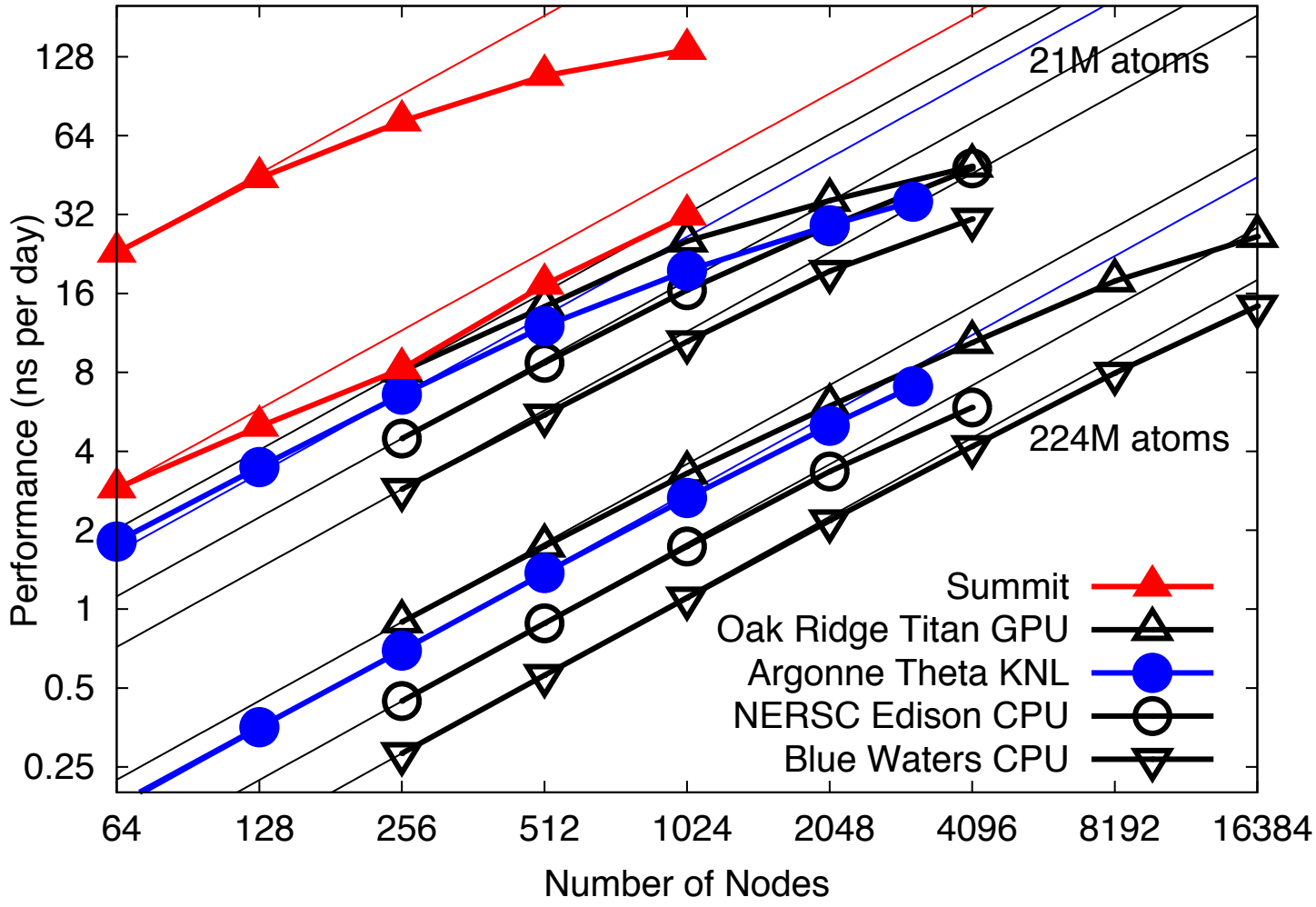
Overloaded PEs (256 nodes) are no longer GPU hosts



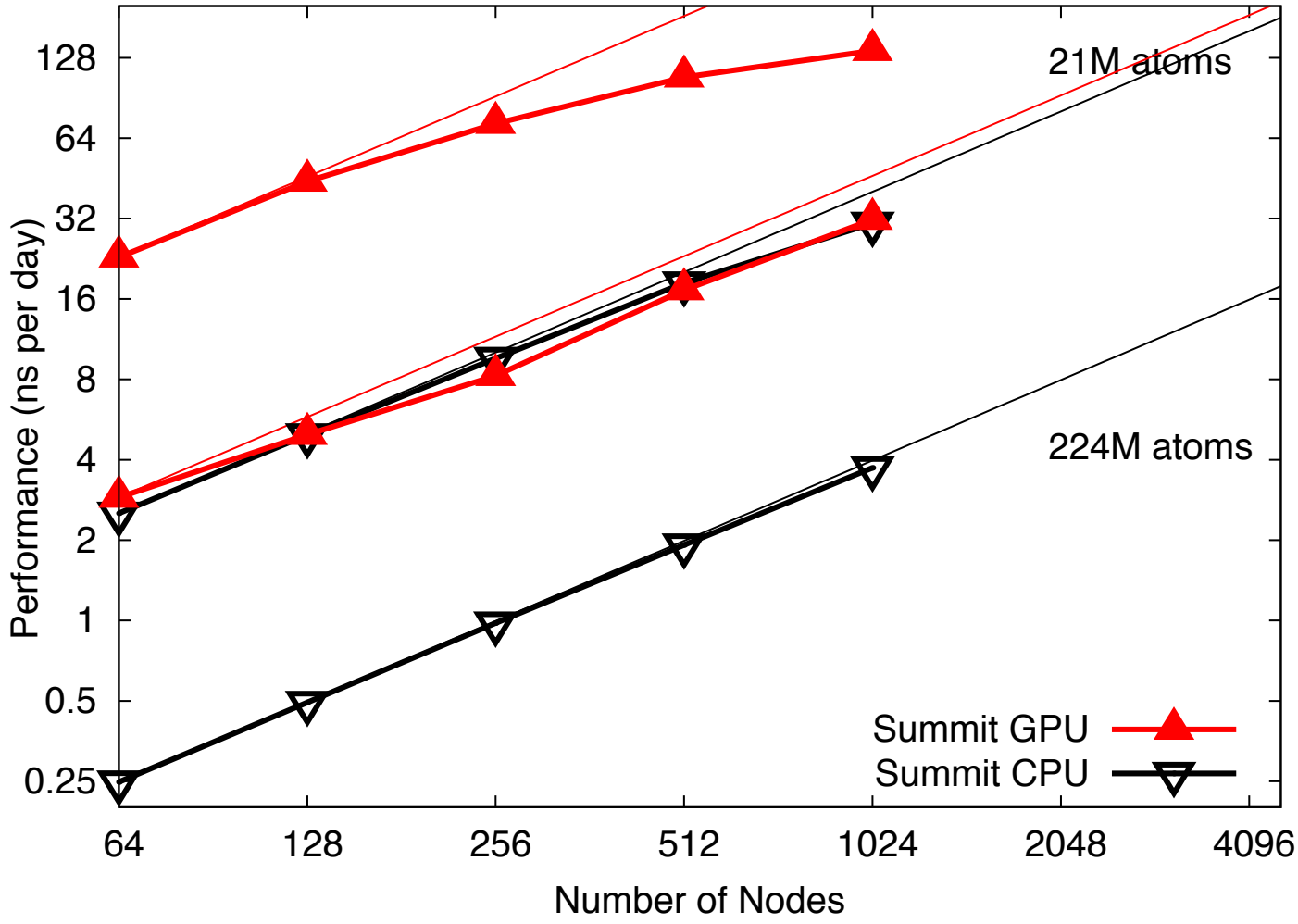
Overloaded PEs still have idle time

Now showing all PEs on process

Comparison for large benchmarks

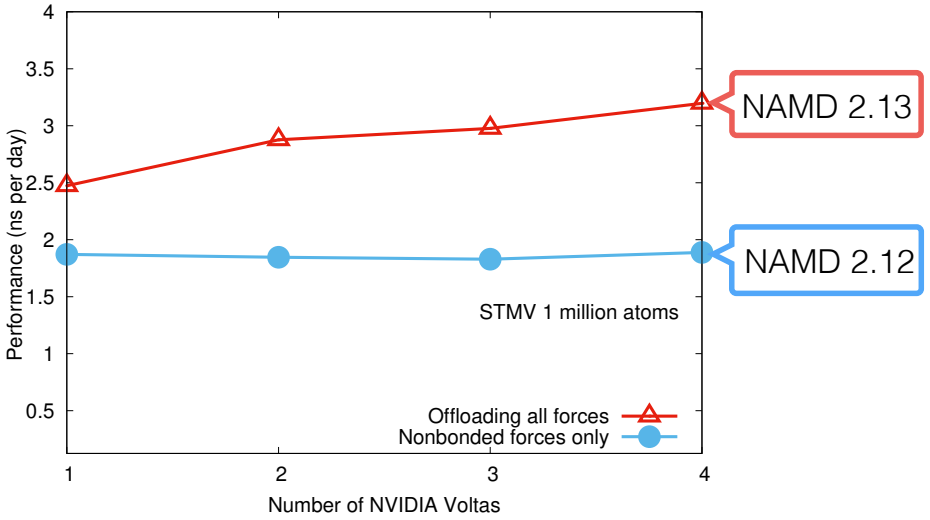


Comparison for large benchmarks

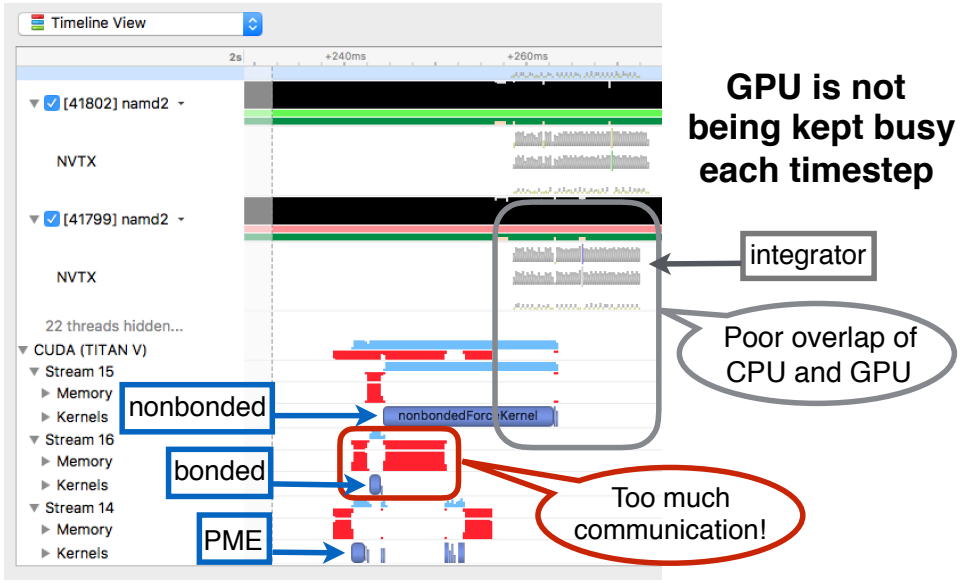


Challenge: NAMD Multi-GPU Scaling Limitations

Results on NVIDIA DGX-1
(Intel Haswell using 28-cores with Volta V100 GPUs)



Nsight Systems profiling of NAMD running
*STMV (1M atoms) on **1 Volta** & 28 CPU cores*



- NAMD on NVIDIA Volta is rate limited by any CPU work that grows with the number of atoms: integrator, reductions, **rigid bond constraints**, **random number generation**
- Performance on Summit is impacted by limited single-node multi-GPU scaling.
- Offloading the integrator will still get less than the available performance due to host-to-device memory copying for a CPU-based code — **overcome with GPU-based NAMD**

Strategies for Overcoming Bottleneck

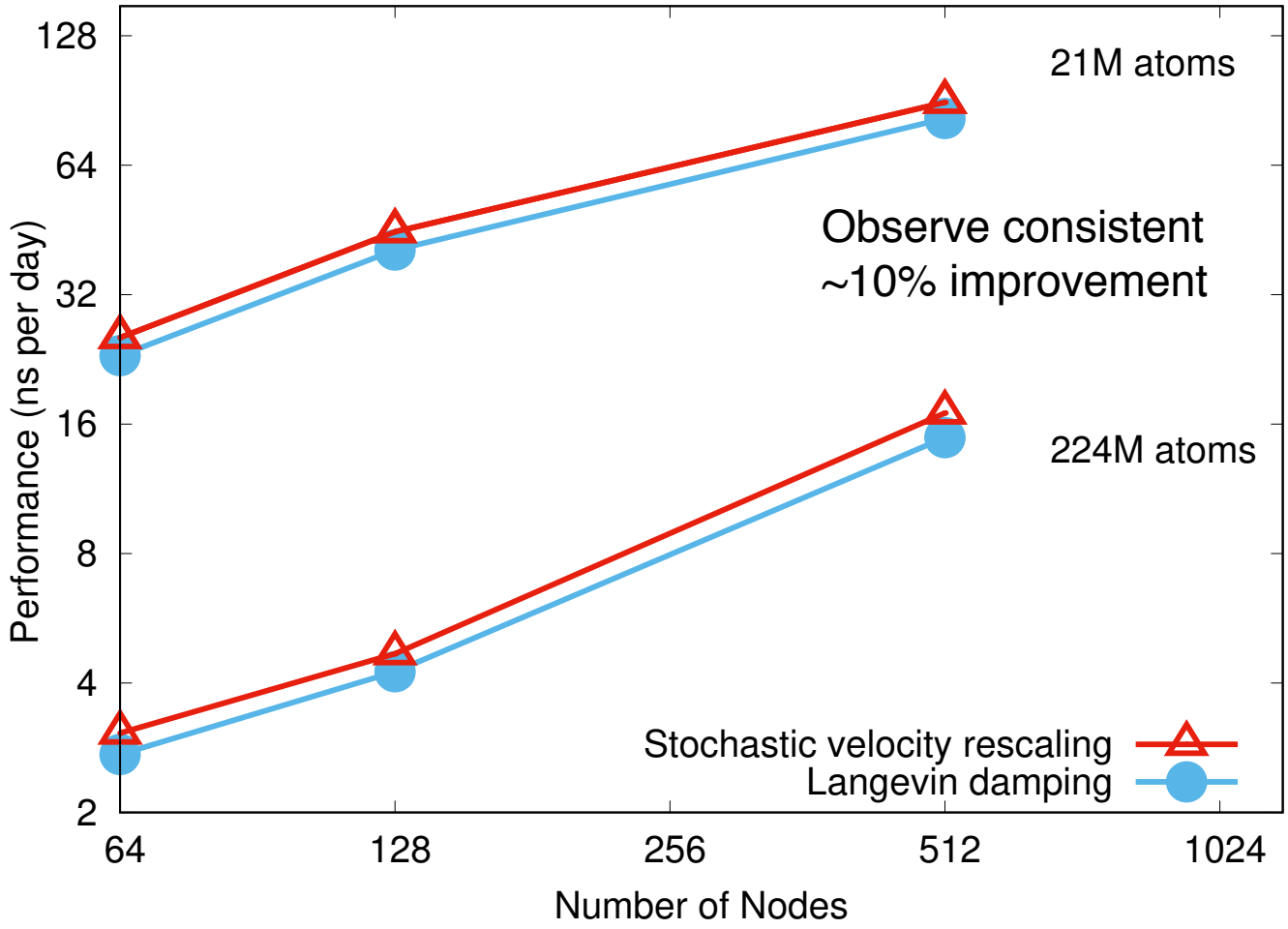
- Data structures for CPU vectorization
  - **Convert atom data storage from AOS (array of structures) form into vector friendly SOA (structure of arrays) form**
- Algorithms for CPU vectorization
  - **Replace non-vectorizing random number generator code with vectorized version**
  - **Replace rigid bond constraints sequential algorithm with one capable of fine-grained parallelism** (maybe LINCS or Matrix-SHAKE)
- **Offload integrator to GPU**
  - Main challenge is aggregating patch data
  - Use vectorized algorithms, adapt curand for Gaussian random numbers

Stochastic velocity rescaling thermostat

Bussi, Donadio, Parrinello, *J. Chem. Phys.* 126, 2007

- Replace Langevin thermostat with stochastic correction to classic Berendsen thermostat that samples canonical ensemble
- Rather than O(N) Gaussian random numbers every step, need only 2 Gaussian random numbers, around every 20 steps
- Preserves holonomic constraints so no additional rigid bond constraint is needed, as required to stabilize Langevin
- Observed 10-20% performance improvement on GPU-based runs

Stochastic velocity rescaling on Summit



Acknowledgments

Antti-Pekka Hynninen & Ke Li, NVIDIA
Sameer Kumar & Bilge Acun, IBM
Tjerk Straatsma, OLCF
William Kramer, NCSA
Alexander Bobyr & Michael Brown, Intel
Abhi Singharoy, ASU

NIH Center for Macromolecular Modeling and Bioinformatics
University of Illinois at Urbana-Champaign