S8709—Accelerating Molecular Modeling on Desktop and Pre-Exascale Supercomputers

John E. Stone

Theoretical and Computational Biophysics Group

Beckman Institute for Advanced Science and Technology

University of Illinois at Urbana-Champaign

http://www.ks.uiuc.edu/Research/gpu/

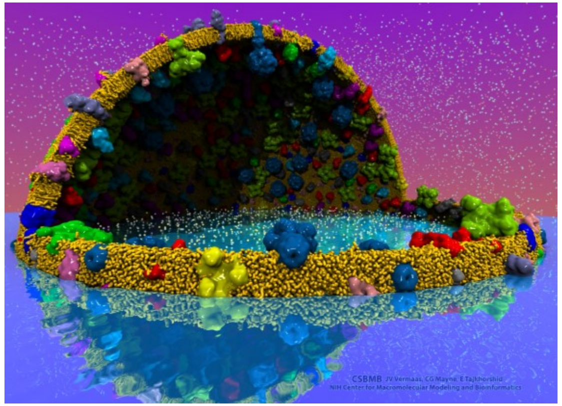
S8709, GPU Technology Conference

4:00-4:50, San Carlos Room, Hilton,

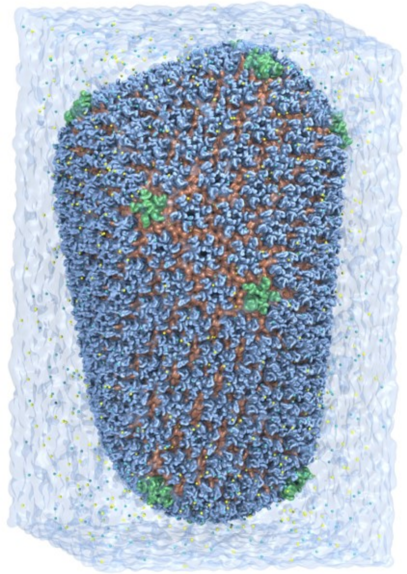
San Jose, CA, Monday March 26th, 2018

VMD – "Visual Molecular Dynamics"

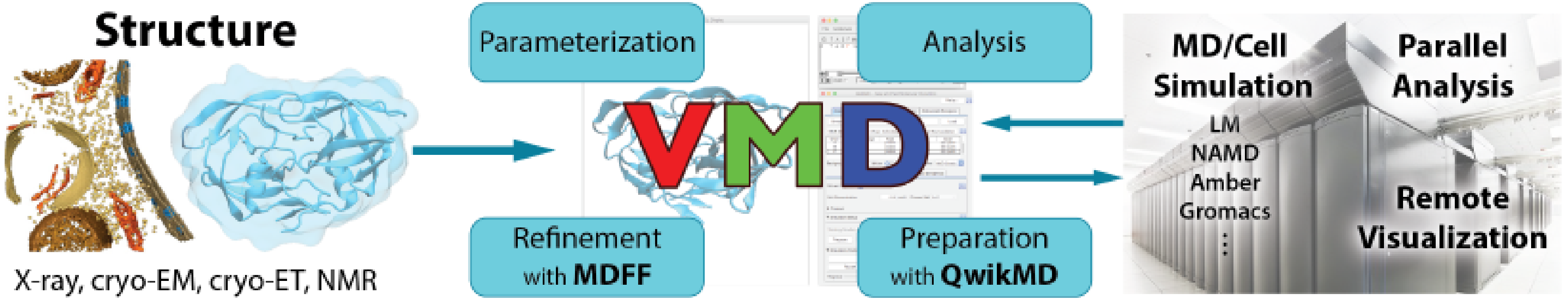
- Visualization and analysis of:
  - Molecular dynamics simulations
  - Lattice cell simulations
  - Quantum chemistry calculations
  - Sequence information
- User extensible scripting and plugins
- http://www.ks.uiuc.edu/Research/vmd/


CSBMB JV Vermaas, CG Mayne, E Tajkhorshid
NIH Center for Macromolecular Modeling and Bioinformatics

Cell-Scale Modeling



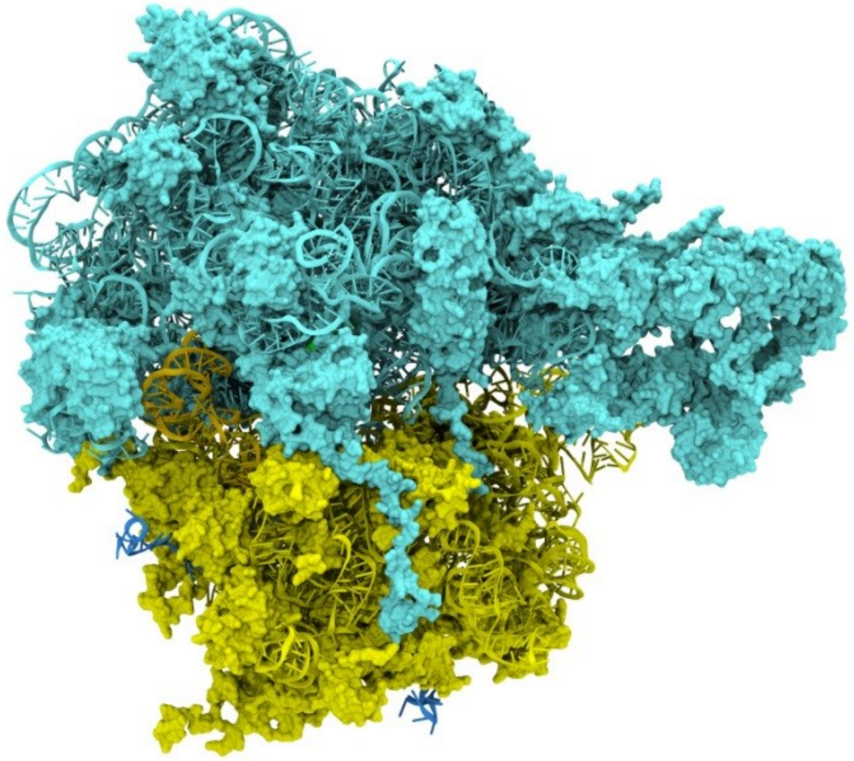
MD Simulation



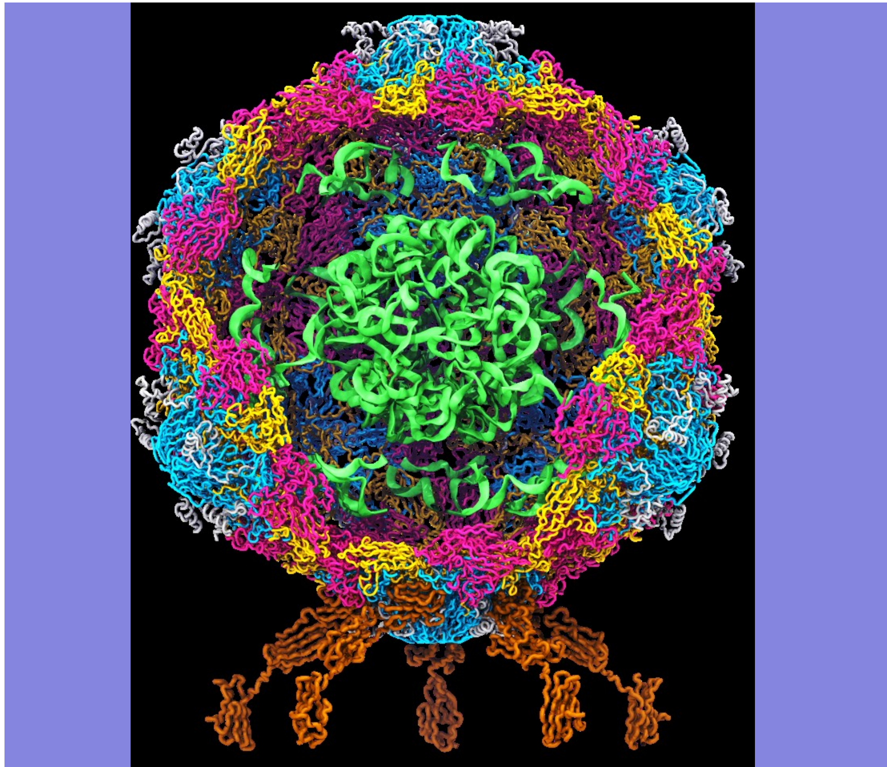
Goal: A Computational Microscope

Study the molecular machines in living cells

Ribosome: target for antibiotics



Poliovirus



VMD Petascale Visualization and Analysis

- Analyze/visualize large trajectories too large to transfer off-site:
  - User-defined parallel analysis operations, data types
  - Parallel rendering, movie making
- Supports GPU-accelerated Cray XK7 nodes for both visualization and analysis:
  - **GPU accelerated trajectory analysis w/ CUDA**
  - **OpenGL and GPU ray tracing for visualization and movie rendering**
- Parallel I/O rates up to **275 GB/sec** on 8192 Cray XE6 nodes – can read in **231 TB in 15 minutes!**

Parallel VMD currently available on:

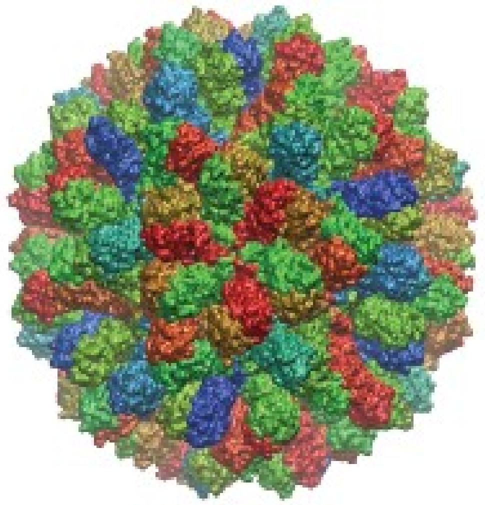
ORNL Titan, NCSA Blue Waters, Indiana Big Red II, CSCS Piz Daint, and similar systems

NCSA Blue Waters Hybrid Cray XE6 / XK7
22,640 XE6 dual-Opteron CPU nodes
4,224 XK7 nodes w/ Telsa K20X GPUs

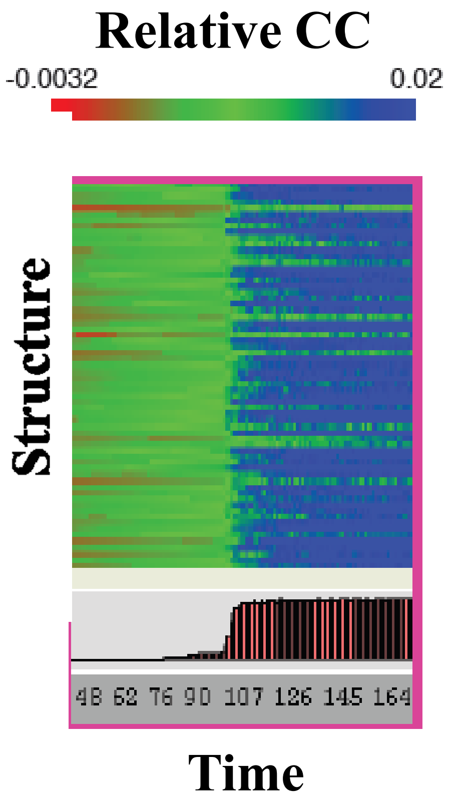
Parallel MDFF Cross Correlation Analysis on Cray XK7

Rabbit Hemorrhagic Disease Virus (RHDV)	
Traj. frames	10,000
Structure component selections	720
Single-node XK7 (projected)	336 hours (14 days)
128-node XK7	3.2 hours 105x speedup
2048-node XK7	19.5 minutes 1035x speedup

Calculation of 7M CCs would take **5 years** using serial CPU algorithm!



RHDV colored by relative CC



Stone et al., Faraday Discuss., 169:265-283, 2014.

Making Our Research Tools Easily Accessible

- Docker "container" images available in NVIDIA NGC registry
  - Users obtain Docker images via registry, download and run on the laptop, workstation, cloud, or supercomputer of their choosing
  - **https://ngc.nvidia.com/registry/**
  - **https://ngc.nvidia.com/registry/hpc-vmd**

- Cloud based deployment
  - Full virtual machines (known as "AMI" in Amazon terminology)
  - Amazon AWS EC2 GPU-accelerated instances:

    http://www.ks.uiuc.edu/Research/cloud/

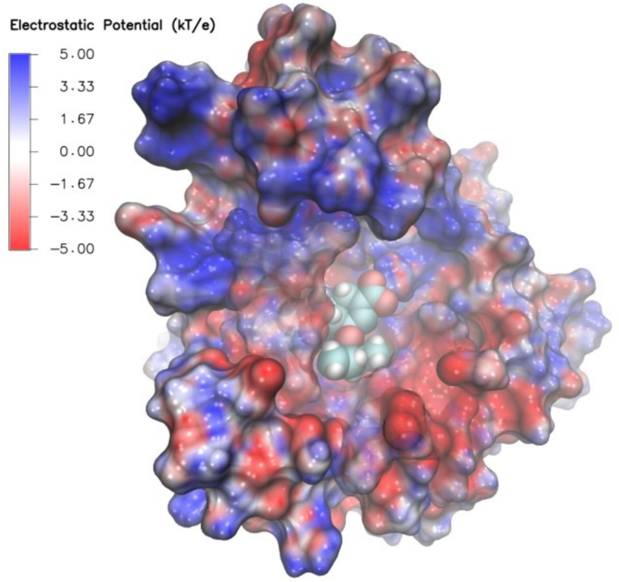
Clusters, Supercomputers

Workstations,
Servers,
Cloud

Molecular dynamics-based refinement and validation for sub-5 Å cryo-electron microscopy maps. Abhishek Singharoy, Ivan Teo, Ryan McGreevy, John E. Stone, Jianhua Zhao, and Klaus Schulten. *eLife*, 10.7554/eLife.16105, 2016. (66 pages).
QwikMD-integrative molecular dynamics toolkit for novices and experts. Joao V. Ribeiro, Rafael C. Bernardi, Till Rudack, John E. Stone, James C. Phillips, Peter L. Freddolino, and Klaus Schulten. *Scientific Reports*, 6:26536, 2016.
High performance molecular visualization: In-situ and parallel rendering with EGL. John E. Stone, Peter Messmer, Robert Sisneros, and Klaus Schulten. *2016 IEEE International Parallel and Distributed Processing Symposium Workshop (IPDPSW)*, pp. 1014-1023, 2016.

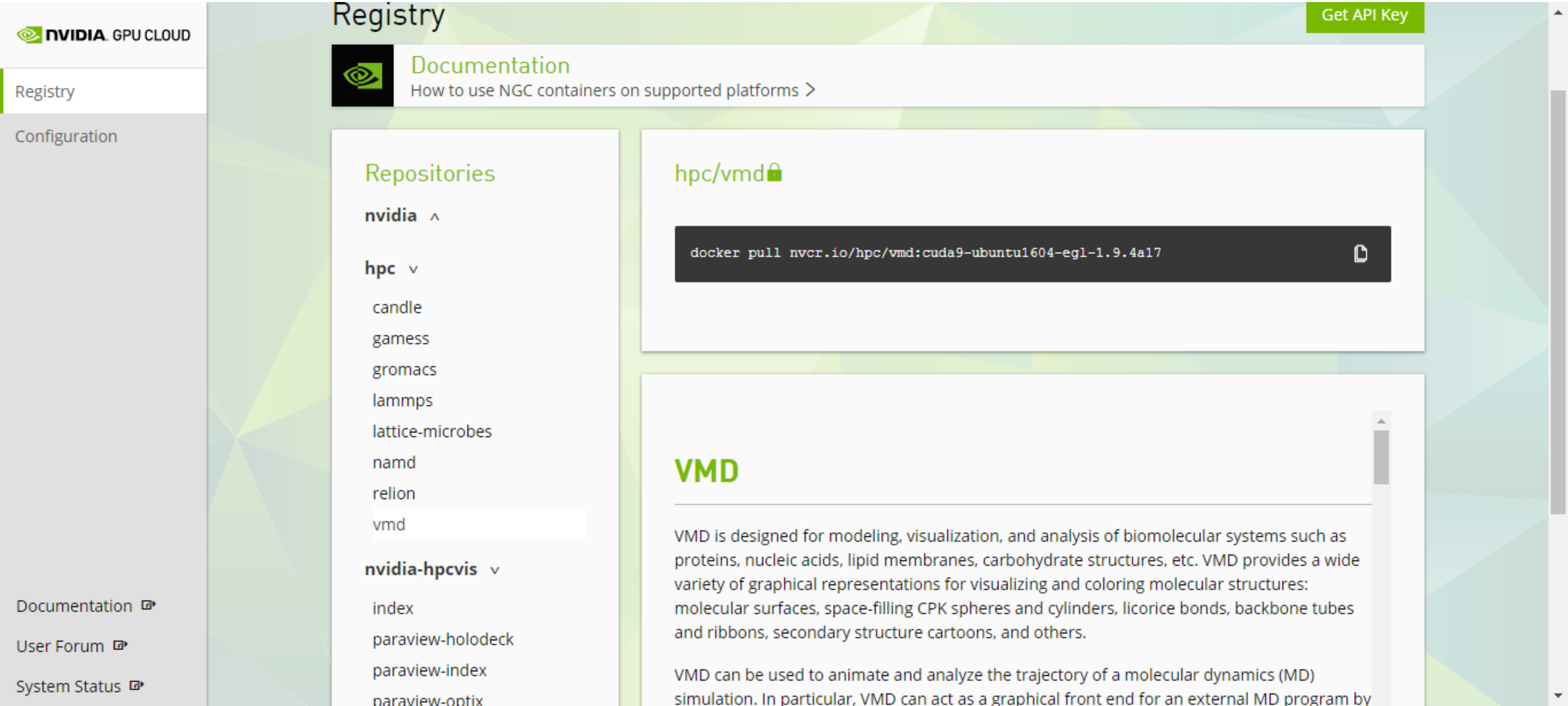
VMD EGL-Enabled NGC Container

- **https://ngc.nvidia.com/registry/**
- CUDA-accelerated analysis
- EGL off-screen rendering – no windowing system needed
- OptiX high-fidelity GPU ray tracing engine built in
- All dependencies included
- **Easy to deploy on a wide range of GPU accelerated platforms**



High performance molecular visualization: In-situ and parallel rendering with EGL. J. E. Stone, P. Messmer, R. Sisneros, and K. Schulten. *2016 IEEE International Parallel and Distributed Processing Symposium Workshop (IPDPSW)*, pp. 1014-1023, 2016.

VMD / NAMD / LM, NGC Containers



Registry

Documentation

How to use NGC containers on supported platforms >

Repositories

nvidia ^

hpc v

candle
gamess
gromacs
lammps
lattice-microbes
namd
relion
vmd

nvidia-hpcvis v

index
paraview-holodeck
paraview-index
paraview-optix

hpc/vmd

```
docker pull nvcr.io/hpc/vmd:cuda9-ubuntu1604-egl-1.9.4a17
```

VMD

VMD is designed for modeling, visualization, and analysis of biomolecular systems such as proteins, nucleic acids, lipid membranes, carbohydrate structures, etc. VMD provides a wide variety of graphical representations for visualizing and coloring molecular structures: molecular surfaces, space-filling CPK spheres and cylinders, licorice bonds, backbone tubes and ribbons, secondary structure cartoons, and others.

VMD can be used to animate and analyze the trajectory of a molecular dynamics (MD) simulation. In particular, VMD can act as a graphical front end for an external MD program by

VMD on IBM OpenPOWER, CORAL Systems

- VMD has supported POWER hardware since 1998!
- OpenPOWER + CORAL: Linux OS running in little-endian mode:
  - VMD 1.9.3 (Nov 2016): first release on OpenPOWER
  - POWER VSX instructions, hand-vectorized CPU kernels [1]
- In-progress VMD 1.9.4 development:
  - **VMD supports full OpenGL via GLX and EGL on POWER**
  - **ORNL Summit: Power9, CUDA 9.x w/ Tesla V100 (Volta) and NVLink**
  - Ongoing work: **NUMA** optimality, and more **pervasive use of NVLink**

[1] Early Experiences Porting the NAMD and VMD Molecular Simulation and Analysis Software to GPU-Accelerated OpenPOWER Platforms. J. E. Stone, A.-P. Hynninen, J. C. Phillips, K. Schulten. International Workshop on OpenPOWER for HPC (IWOPH'16), LNCS 9945, pp. 188-206, 2016.

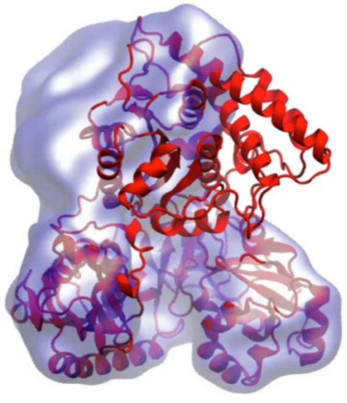
Benefits of P9+NVLink for VMD

- Rapid access to host-side data too large to fit entirely in GPU memory
  - Many existing VMD CUDA kernels already used this strategy w/ PCIe, performance gains from NVLink are large and immediate
- Rapid peer-to-peer GPU data transfers:
  - Bypass host whenever possible, perform nearest-neighbor exchanges for pairwise calculations, e.g. those that arise in algorithms for simulation trajectory clustering
  - Use aggregate GPU memory to collectively store/cache large data – well suited for high-fidelity ray tracing of scenes containing massive amounts of geometry

Molecular Dynamics Flexible Fitting (MDFF)

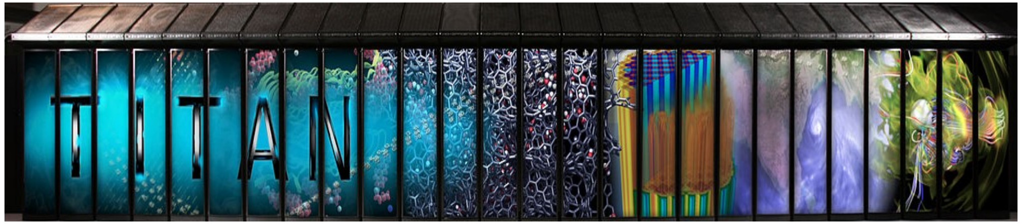
$$\begin{tikzcd}\text{X-ray crystallography} \arrow[r] & \text{MDFF} & \text{Electron microscopy} \arrow[l]\end{tikzcd}$$

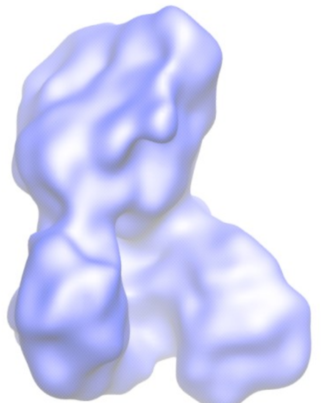
APS at Argonne



FEI microscope

ORNL Titan





Molecular dynamics-based refinement and validation for sub-5Å cryo-electron microscopy maps. A. Singharoy, I. Teo, R. McGreevy, J. E. Stone, J. Zhao, and K. Schulten. eLife 2016;10.7554/eLife.16105

Molecular Dynamics Flexible Fitting - Theory

Two terms are added to the MD potential

$$U_{total} = U_{MD} + U_{EM} + U_{SS}$$

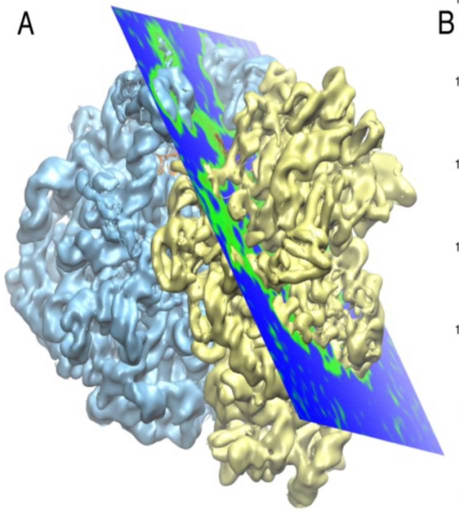
An external potential derived from the EM map is defined on a grid as

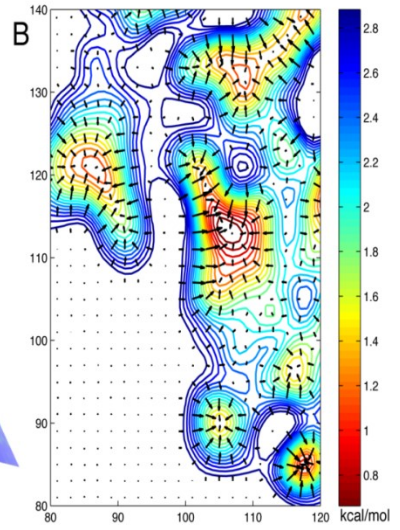
$$U_{EM}(\mathbf{R}) = \sum_j w_j V_{EM}(\mathbf{r}_j)$$

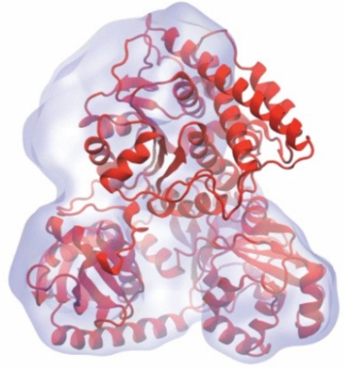
$$V_{EM}(\mathbf{r}) = \begin{cases} \xi \left(1 - \frac{\Phi(\mathbf{r}) - \Phi_{thr}}{\Phi_{max} - \Phi_{thr}}\right) & \text{if } \Phi(\mathbf{r}) \geq \Phi_{thr}, \\ \xi & \text{if } \Phi(\mathbf{r}) < \Phi_{thr}. \end{cases}$$

A mass-weighted force is then applied to each atom

$$\mathbf{f}_i^{EM} = -\nabla U_{EM}(\mathbf{R}) = -w_i \partial V_{EM}(\mathbf{r}_i)/\partial r_i$$

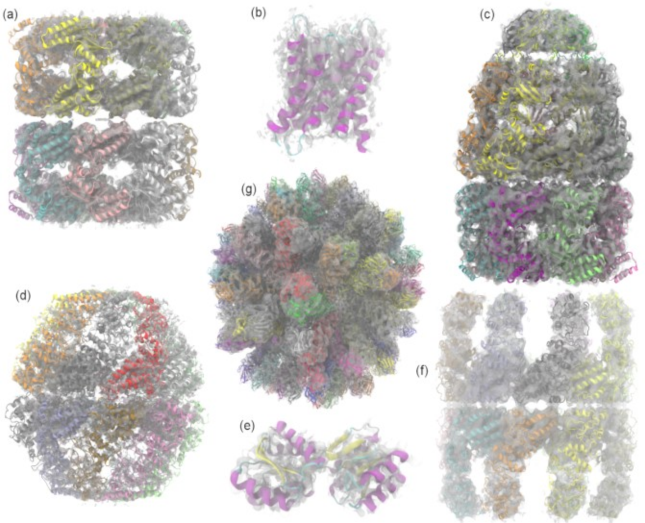


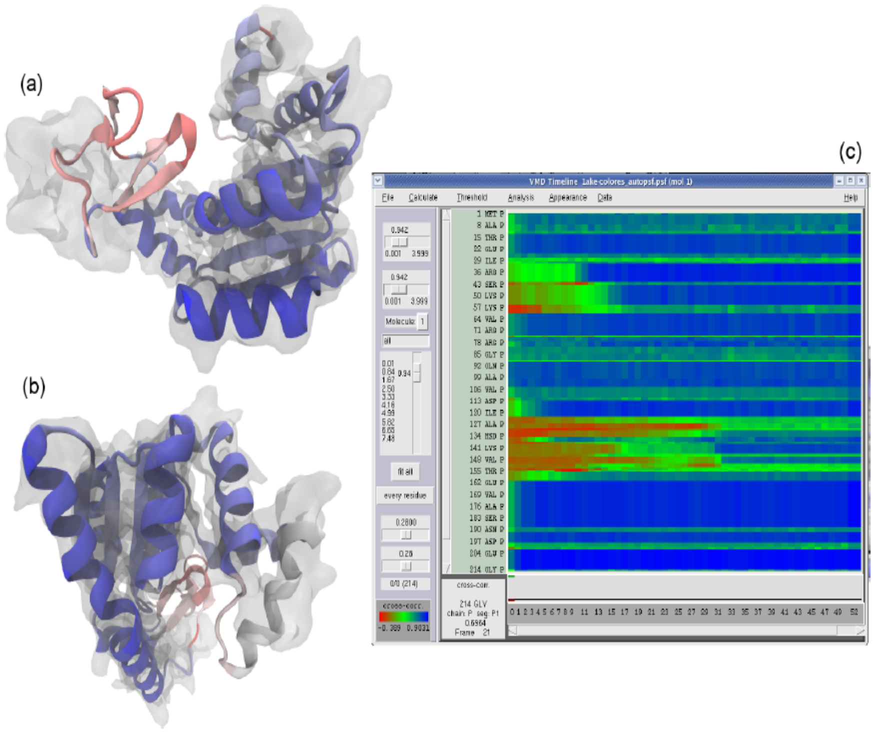




Evaluating Quality-of-Fit for Structures Solved by Hybrid Fitting Methods

Compute Pearson correlation to evaluate quality-of-fit between a reference cryo-EM density map and a **simulated density map** from an **all-atom structure**.





MDFF Cross Correlation Timeline

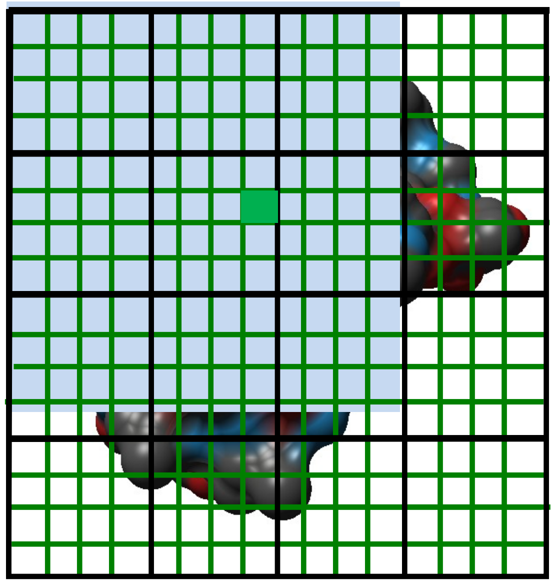
Regions with poor fit **Regions with good fit**

Simulated Density Map Algorithm

- Build spatial acceleration data structures, optimize data for GPU
- Compute 3-D density map:

$$\rho(\vec{r};\vec{r}_1,\vec{r}_2,\ldots,\vec{r}_N)=\sum_{i=1}^{N} e^{\frac{-|\vec{r}-\vec{r}_i|^2}{2\alpha^2}}$$

- Truncated Gaussian and spatial acceleration grid ensure **linear time-complexity**



3-D density map lattice point and the neighboring spatial acceleration cells it references

Single-Pass GPU Cross-Correlation

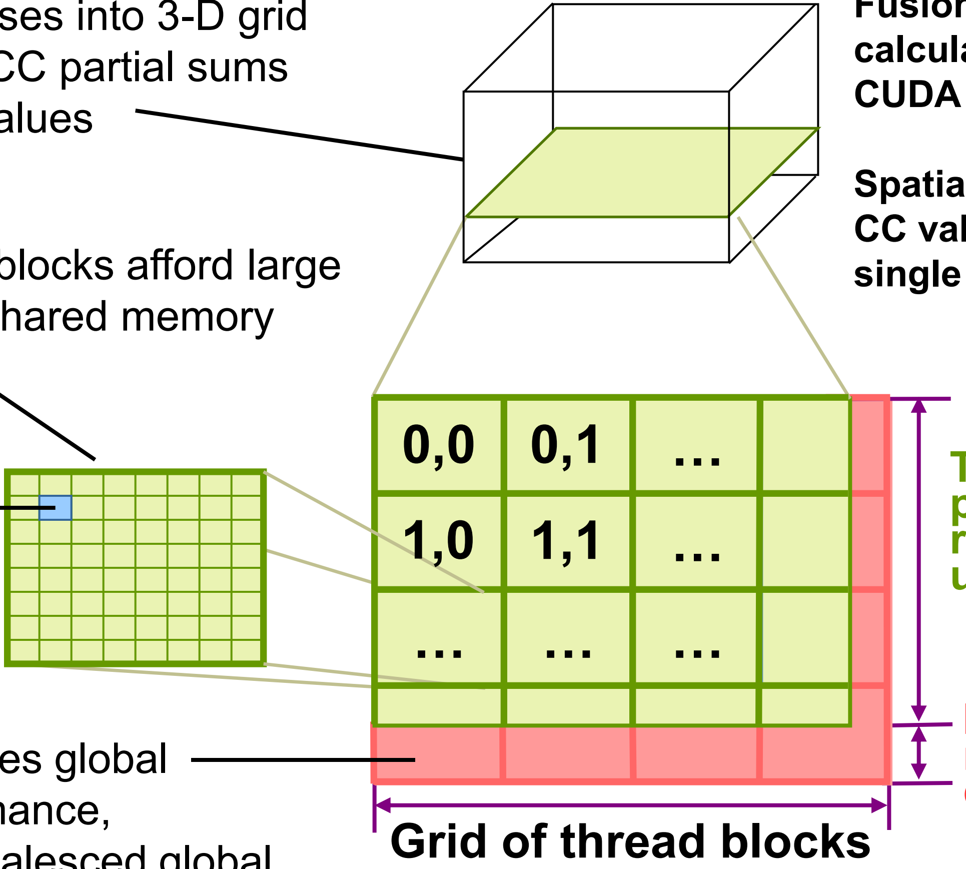
3-D density map decomposes into 3-D grid of 8x8x8 tiles containing CC partial sums and local CC values

Fusion of density and CC calculations into a single CUDA kernel!!!

Spatial CC map and overall CC value computed in a single pass

Small 8x8x2 CUDA thread blocks afford large per-thread register count, shared memory

Each thread computes 4 z-axis density map lattice points and associated CC partial sums



0,0
0,1
…
1,0
1,1
…
…
…
…
Threads producing results that are used
Inactive threads, region of discarded output
Grid of thread blocks

Padding optimizes global memory performance, guaranteeing coalesced global memory accesses

VMD Tesla V100 Cross Correlation Performance

Rabbit Hemorrhagic Disease Virus: 702K atoms, 6.5Å resolution
Volta GPU architecture almost 2x faster than previous gen Pascal:

Application and Hardware platform	**Runtime,**	**Speedup vs. Chimera,**	**VMD+GPU**
Chimera Xeon E5-2687W (2 socket) [1]	15.860s,	1x	
VMD-CUDA IBM Power8 + 1x Tesla K40 [2]	0.488s,	32x	**0.9x**
VMD-CUDA Intel Xeon E5-2687W + 1x Quadro K6000 [1,2]	0.458s,	35x	**1.0x**
VMD-CUDA Intel Xeon E5-2698v3 + 1x Tesla P100	**0.090s,**	176x	**5.1x**
VMD-CUDA IBM Power8 "Minsky" + 1x Tesla P100	**0.080s,**	198x	**5.7x**
VMD-CUDA Intel Xeon E5-2697Av4 + 1x Tesla V100	**0.050s,**	317x	**9.2x**
VMD-CUDA IBM Power9 "Newell" + 1x Tesla V100	**0.049s,**	**323x**	**9.3x**

[1] GPU-Accelerated Analysis and Visualization of Large Structures Solved by Molecular Dynamics Flexible Fitting. J. E. Stone, R. McGreevy, B. Isralewitz, and K. Schulten. Faraday Discussions 169:265-283, 2014.
[2] Early Experiences Porting the NAMD and VMD Molecular Simulation and Analysis Software to GPU-Accelerated OpenPOWER Platforms. J. E. Stone, A.-P. Hynninen, J. C. Phillips, K. Schulten. International Workshop on OpenPOWER for HPC (IWOPH'16), LNCS 9945, pp. 188-206, 2016.

Volta-Specific CC Optimization Opportunities

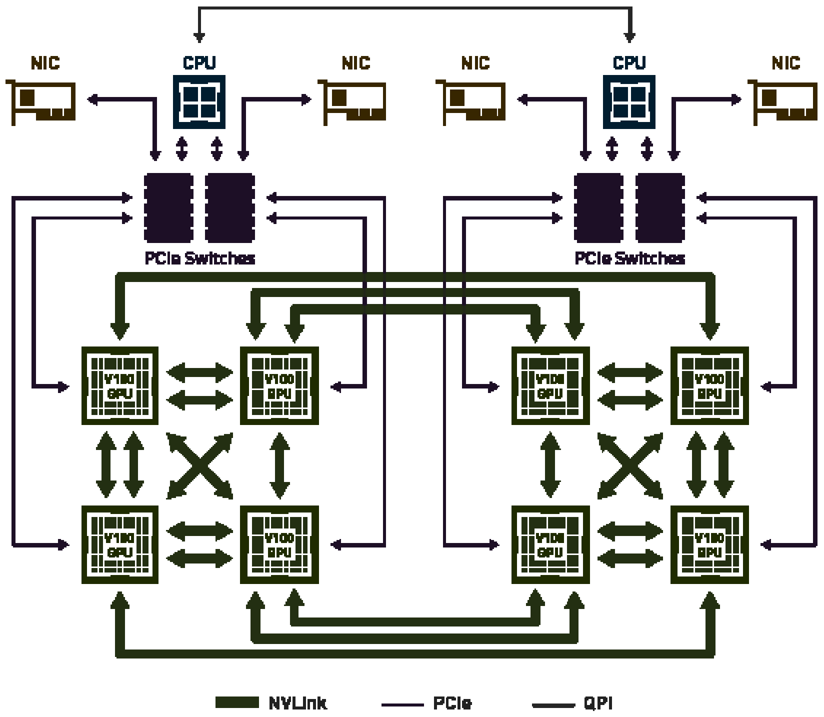
- Optimized precision for both ref/simulated maps:
  - Improved memory bandwidth, lower arithmetic cost
  - FP16: half-precision EM density map representation
  - INT8: byte density map representation **for EM tomograms**
- Explore use of Tensor Core for certain parts of cross correlation calculations, convolutions for noise filters:
  - Challenge: CUDA 9.x APIs for TC limit the range of data movement patterns that perform well
  - Difficult to prevent TC from becoming mem bandwidth-bound
  - Ongoing TC experiments for VMD image segmentation kernels may lead to schemes that can work for cross correlation and other calculations

Challenges Posed by Next-Gen GPU-Dense Workstation/Node Designs

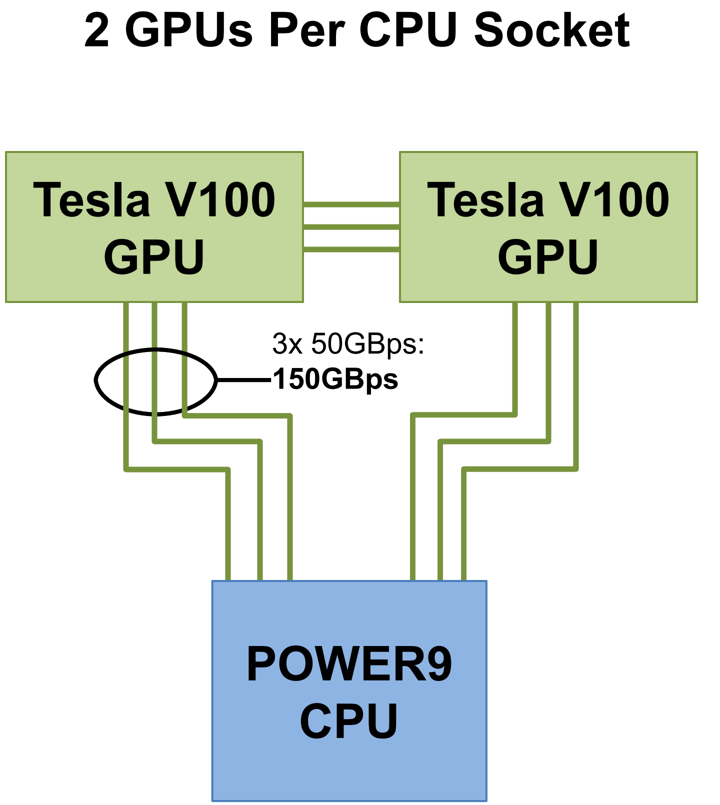
Application and Hardware platform	**Runtime,**	**VMD+GPU**
VMD-CUDA Intel Xeon E5-2687W + 1x Quadro K6000 [1,2]	0.458s,	**1.0x**
VMD-CUDA IBM Power9 "Newell" + 1x Tesla V100	**0.049s,**	**9.3x**

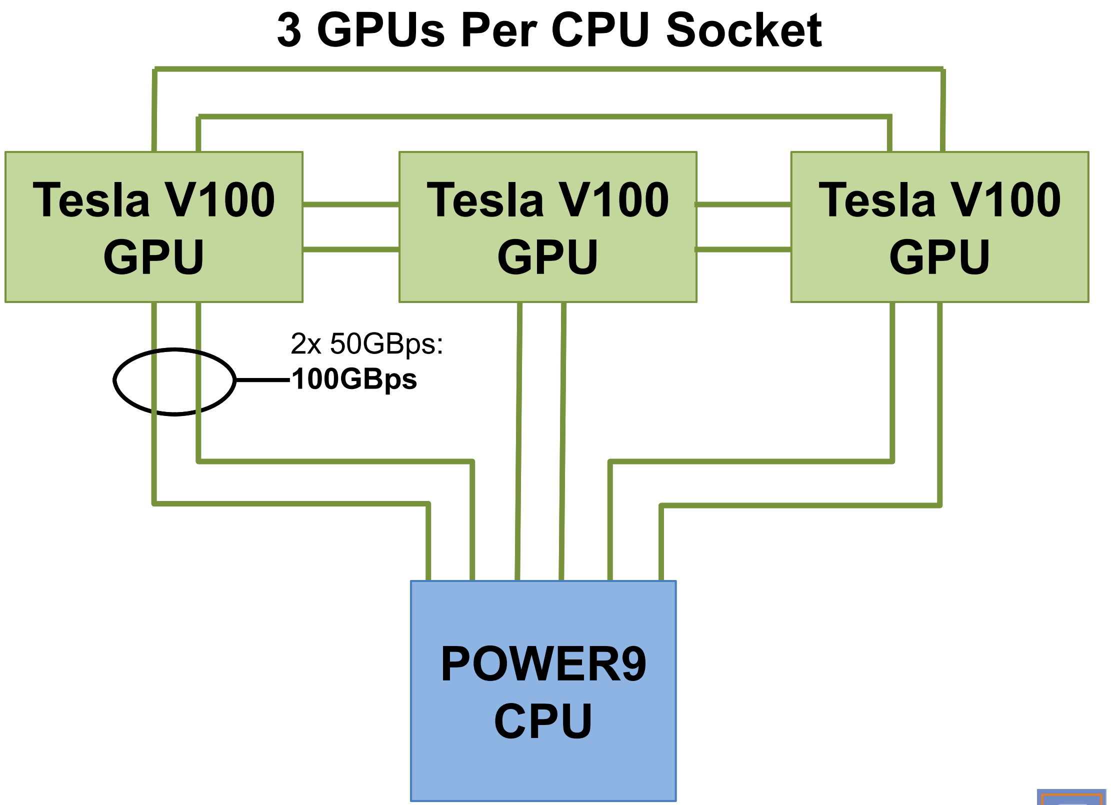
- ~9x observed performance gain from Kepler to Volta GPUs
- CPUs and PCIe have not matched this performance gain
- New GPU-dense nodes have far less CPU available to manage GPUs and execute non-GPU code
  - More GPUs per CPU socket, fewer CPU threads per GPU
  - ORNL Summit 3 GPUs/socket, 7 CPU cores per GPU

NVIDIA DGX-1

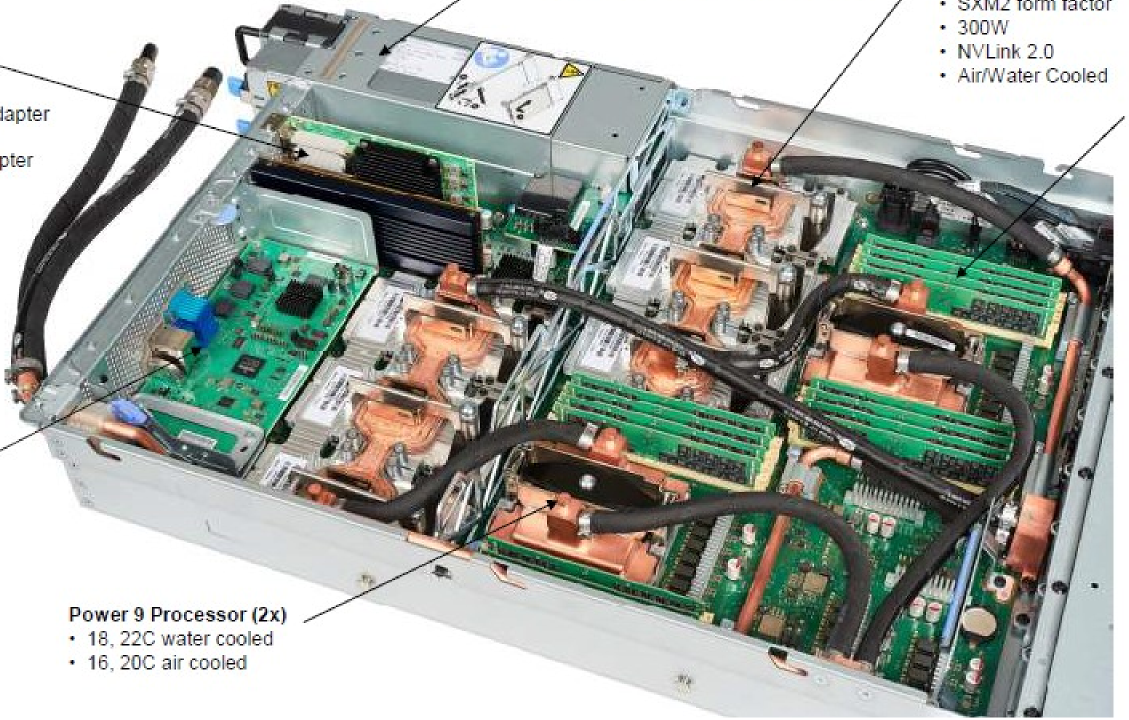

NIC
CPU
NIC
NIC
CPU
NIC
PCIe Switches
PCIe Switches
V100 GPU
NVLink
PCIe
QPI

IBM AC922 NVLink 2.0 Topologies





IBM AC922 w/ 6 GPUs



Power Supplies (2x)
- 2200W
- 200VAC, 277VAC, 400VDC input

NVidia Volta GPU
- 3 per socket
- SXM2 form factor
- 300W
- NVLink 2.0
- Air/Water Cooled

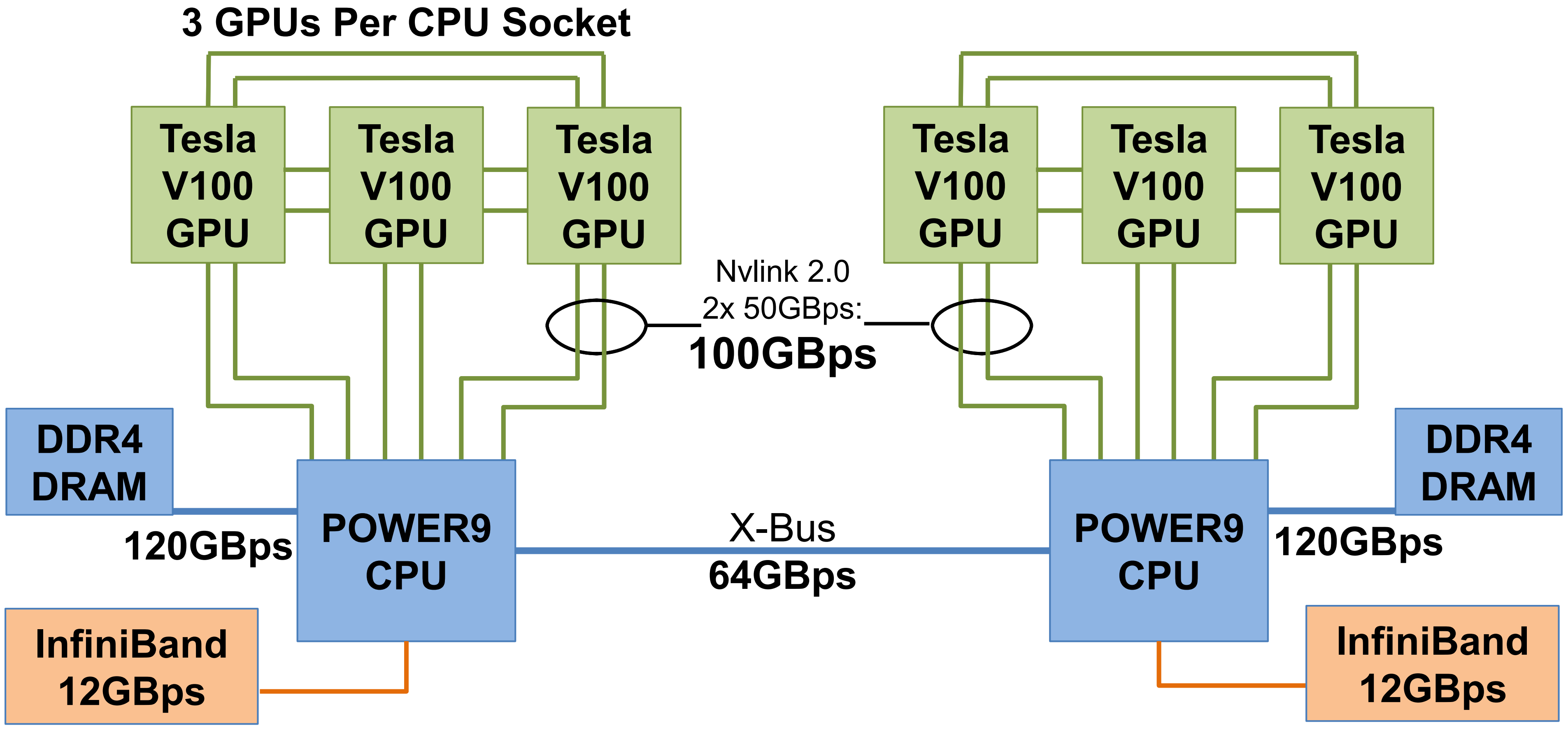
PCIe slot (4x)
- Gen4 PCIe
- 2, x16 HHHL Adapter
- 1, Shared slot
- 1 x8 HHHL Adapter

Memory DIMM's (16x)
- 8 DDR4 IS DIMMs per socket
- 8, 16, 32,64, 128GB DIMMs

BMC Card
- IPMI
- 1 Gb Ethernet
- VGA
- 1 USB 3.0

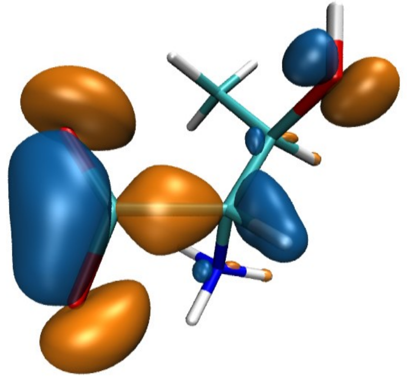
Power 9 Processor (2x)
- 18, 22C water cooled
- 16, 20C air cooled

IBM AC922 Summit Node



Computing Molecular Orbitals

- Animation of (classical mechanics) molecular dynamics trajectories provides insight into simulation results
- To do the same for QM or hybrid QM/MM simulations one must compute MOs at ~**5-10 FPS** or more

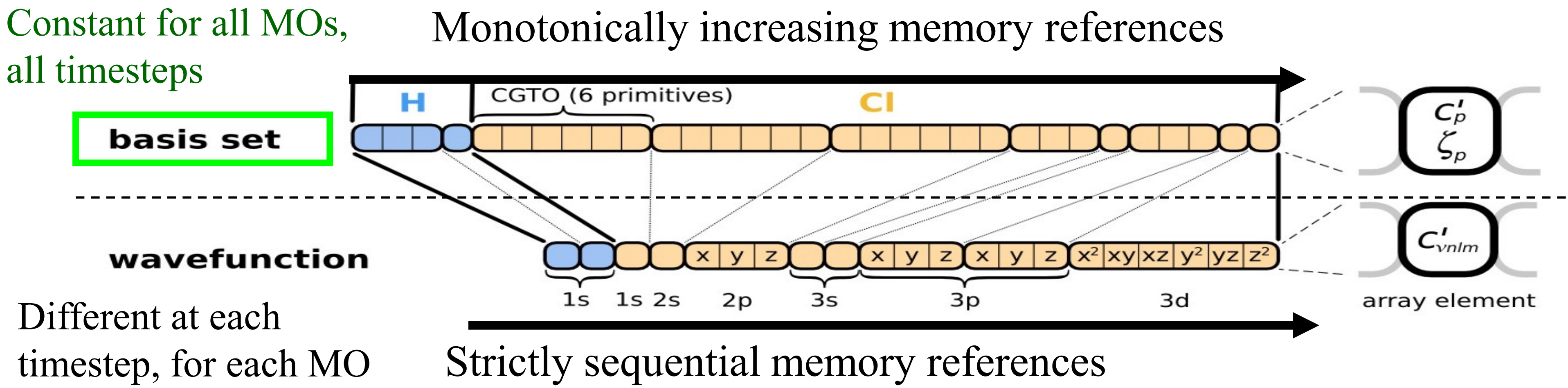


NAMD goes quantum: An integrative suite for hybrid simulations. Melo, M. C. R.; Bernardi, R. C.; Rudack T.; Scheurer, M.; Riplinger, C.; Phillips, J. C.; Maia, J. D. C.; Rocha, G. D.; Ribeiro, J. V.; Stone, J. E.; Neese, F.; Schulten, K.; Luthey-Schulten, Z.; **Nature Methods**, 2018.

http://dx.doi.org/10.1038/nmeth.4638

High Performance Computation and Interactive Display of Molecular Orbitals on GPUs and Multi-core CPUs. J. E. Stone, J. Saam, D. Hardy, K. Vandivort, W. Hwu, K. Schulten, *2nd Workshop on General-Purpose Computation on Graphics Processing Units (GPGPU-2), ACM International Conference Proceeding Series*, volume 383, pp. 9-18, 2009.

GPU Traversal of Atom Type, Basis Set, Shell Type, and Wavefunction Coefficients


Constant for all MOs, all timesteps
Monotonically increasing memory references
H
CGTO (6 primitives)
Cl
basis set
c'_p ζ_p
wavefunction
x y z x y z x y z
x^2 xy xz y^2 yz z^2
1s 1s 2s 2p 3s 3p 3d
$c'_{\nu nlm}$
array element
Different at each timestep, for each MO
Strictly sequential memory references

- Loop iterations always access same or consecutive array elements for all threads in a thread block:
  - Yields good constant memory and L1 cache performance
  - Increases shared memory tile reuse

MO GPU Parallel Decomposition

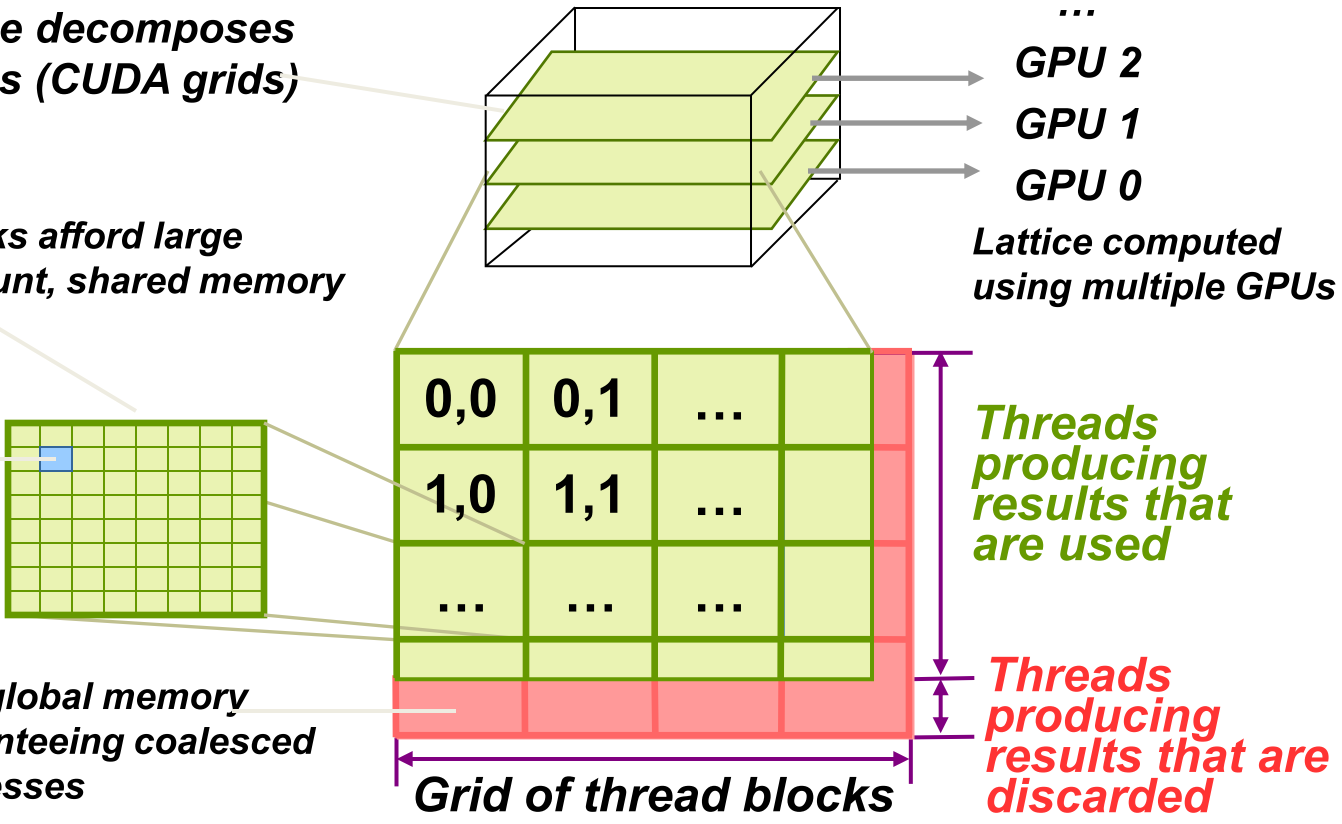
MO 3-D lattice decomposes into 2-D slices (CUDA grids)

Small 8x8 thread blocks afford large per-thread register count, shared memory

Each thread computes one MO lattice point.

Padding optimizes global memory performance, guaranteeing coalesced global memory accesses

Lattice computed using multiple GPUs


…
GPU 2
GPU 1
GPU 0
0,0
0,1
…
1,0
1,1
…
…
…
…
Threads producing results that are used
Threads producing results that are discarded
Grid of thread blocks

MO Kernel for One Grid Point (Naive C)

```
…
for (at=0; at<numatoms; at++) {
    int prim_counter = atom_basis[at];
    calc_distances_to_atom(&atompos[at], &xdist, &ydist, &zdist, &dist2, &xdiv);
    for (contracted_gto=0.0f, shell=0; shell < num_shells_per_atom[at]; shell++) {
        int shell_type = shell_symmetry[shell_counter];
        for (prim=0; prim < num_prim_per_shell[shell_counter];  prim++) {
            float exponent          = basis_array[prim_counter       ];
            float contract_coeff = basis_array[prim_counter + 1];
            contracted_gto += contract_coeff * expf(-exponent*dist2);
            prim_counter += 2;
        }
        for (tmpshell=0.0f, j=0, zdp=1.0f; j<=shell_type; j++, zdp*=zdist) {
           int imax = shell_type - j;
           for (i=0, ydp=1.0f, xdp=pow(xdist, imax); i<=imax; i++, ydp*=ydist, xdp*=xdiv)
              tmpshell += wave_f[ifunc++] * xdp * ydp * zdp;
        }
        value += tmpshell * contracted_gto;
        shell_counter++;
    }
} …..
```

Loop over atoms

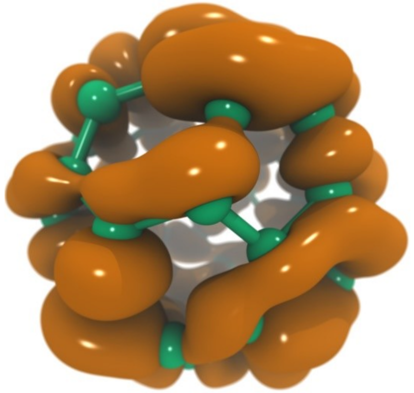
Loop over shells

Loop over primitives: largest component of runtime, due to **expf()**

Loop over angular momenta

(unrolled in real code)

VMD Tesla P100 Performance for C_{60} Molecular Orbitals, 516x519x507 grid

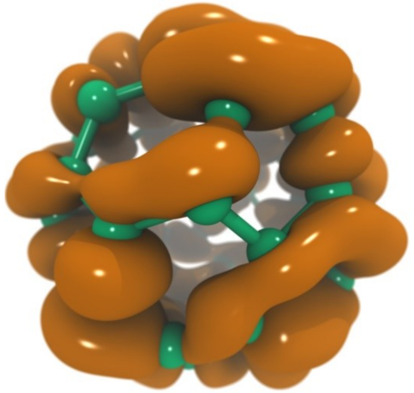


Hardware platform	Runtime,	Speedup
IBM Power8 (2 socket) (ORNL ‘crest’) [1]	8.03s,	0.4x
Intel Xeon E5-2660v3 (2 socket) [1]	7.14s,	0.5x
IBM Power8 (ORNL ‘crest’) + 1x Tesla K40 [1]	3.49s,	**1.0x**
Intel Xeon E5-2698v3 + 1x Tesla P100	**1.35s,**	**2.5x**
IBM Power8 “Minsky” + 1x Tesla P100	**1.09s,**	**3.3x**
IBM Power8 (ORNL ‘crest’) + 4x Tesla K40 [1]	0.91s,	3.8x
Intel Xeon E5-2698v3 + 4x Tesla P100	**0.37s,**	**9.4x**
IBM Power8 “Minsky” + 4x Tesla P100	**0.30s,**	**11.6x**

NVLink perf. boost w/ no code tuning (YET)

[1] Early Experiences Porting the NAMD and VMD Molecular Simulation and Analysis Software to GPU-Accelerated OpenPOWER Platforms. J. E. Stone, A.-P. Hynninen, J. C. Phillips, K. Schulten. International Workshop on OpenPOWER for HPC (IWOPH'16), LNCS 9945, pp. 188-206, 2016.

VMD Tesla V100 Performance for C_{60} Molecular Orbitals, 516x519x507 grid



Hardware platform		Runtime,	Speedup
IBM Power8 (ORNL 'crest') + 1x Tesla K40 [1]		3.49s,	**1.0x**
Intel Xeon E5-2697Av4	**+ 1x Tesla V100**	**0.610s,**	**5.7x**
Intel Xeon E5-2697Av4	**+ 2x Tesla V100**	**0.294s,**	**11.8x**
Intel Xeon E5-2697Av4	**+ 3x Tesla V100**	**0.220s,**	**15.9x**
IBM Power9 "Newell"	**+ 1x Tesla V100**	**0.394s,**	**8.8x**
IBM Power9 "Newell"	**+ 2x Tesla V100**	**0.207s,**	**16.8x**
IBM Power9 "Newell"	**+ 3x Tesla V100**	**0.151s,**	**23.1x**
IBM Power9 "Newell"	**+ 4x Tesla V100**	**0.130s,**	**26.8x**
IBM Power9 "Newell"	**+ 6x Tesla V100**	**0.156s,**	**22.3x**

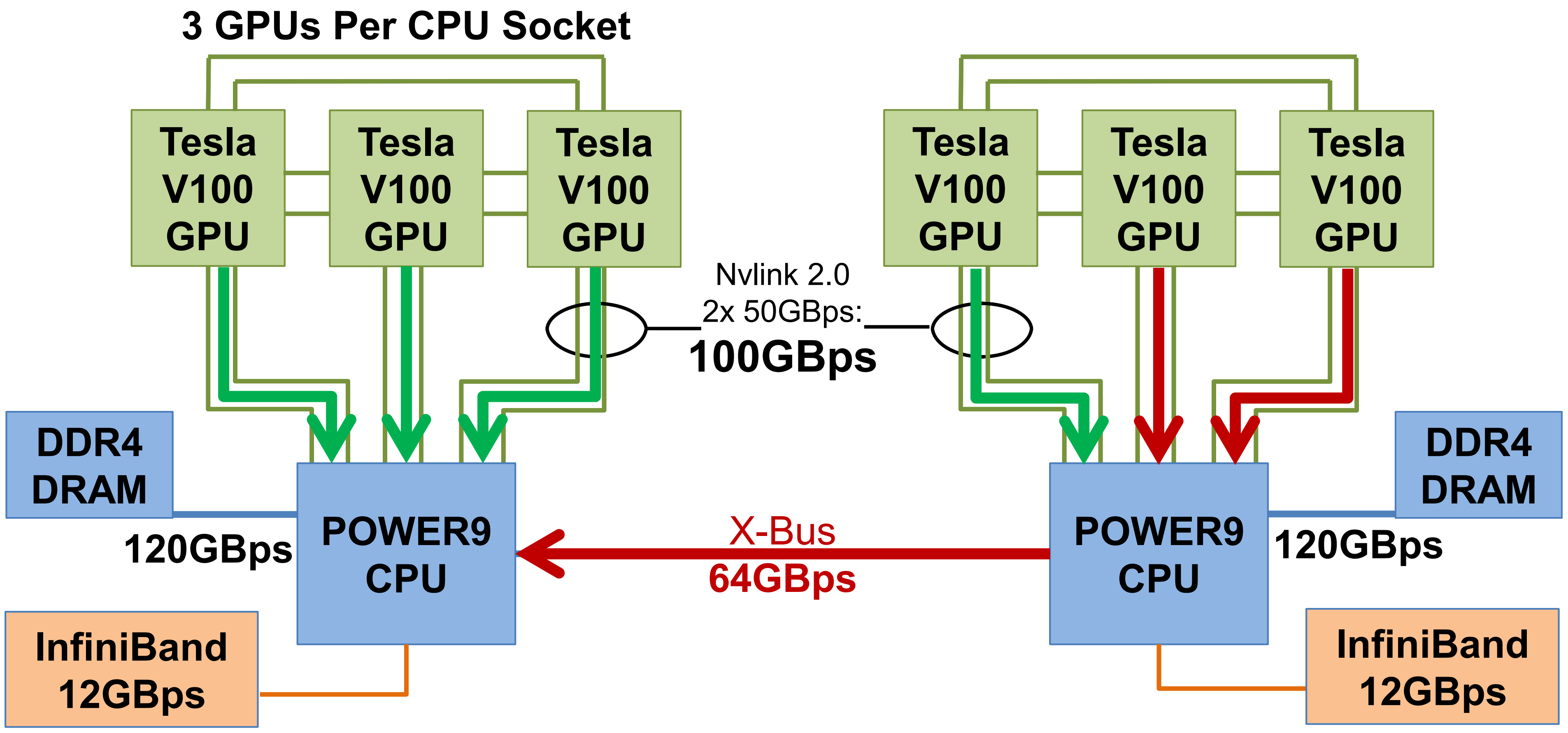
NVLink perf. boost w/ no code tuning (YET)

Need tune

[1] **Early Experiences Porting the NAMD and VMD Molecular Simulation and Analysis Software to GPU-Accelerated OpenPOWER Platforms.** J. E. Stone, A.-P. Hynninen, J. C. Phillips, K. Schulten. International Workshop on OpenPOWER for HPC (IWOPH'16), LNCS 9945, pp. 188-206, 2016.

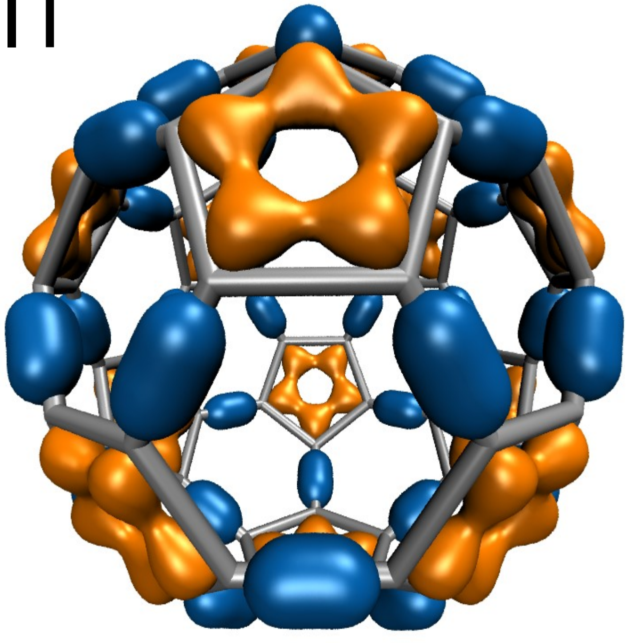
NIH

Molecular Orbital Alg. Behavior on Summit



Molecular Orbitals w/ NVRTC JIT

- Visualization of MOs aids in understanding the chemistry of molecular system
- MO spatial distribution is correlated with probability density for an electron(s)
- **Animation** of (classical mechanics) molecular dynamics trajectories provides insight into simulation results
  - To do the same for QM or QM/MM simulations MOs must be computed at **10 FPS** or more
  - **Large GPU speedups (up to 30x vs. current generation 4-core CPUs)** over existing tools makes this possible!
- **Run-time code generation** (JIT) and compilation via **CUDA NVRTC** enable further optimizations and the **highest performance to date: 1.8x faster than fully-general data-driven loops**



C_{60}

High Performance Computation and Interactive Display of Molecular Orbitals on GPUs and Multi-core CPUs. J. E. Stone, J. Saam, D. Hardy, K. Vandivort, W. Hwu, K. Schulten, *2nd Workshop on General-Purpose Computation on Graphics Processing Units (GPGPU-2), ACM International Conference Proceeding Series*, volume 383, pp. 9-18, 2009.

MO Kernel Structure, Opportunity for NRTC JIT…

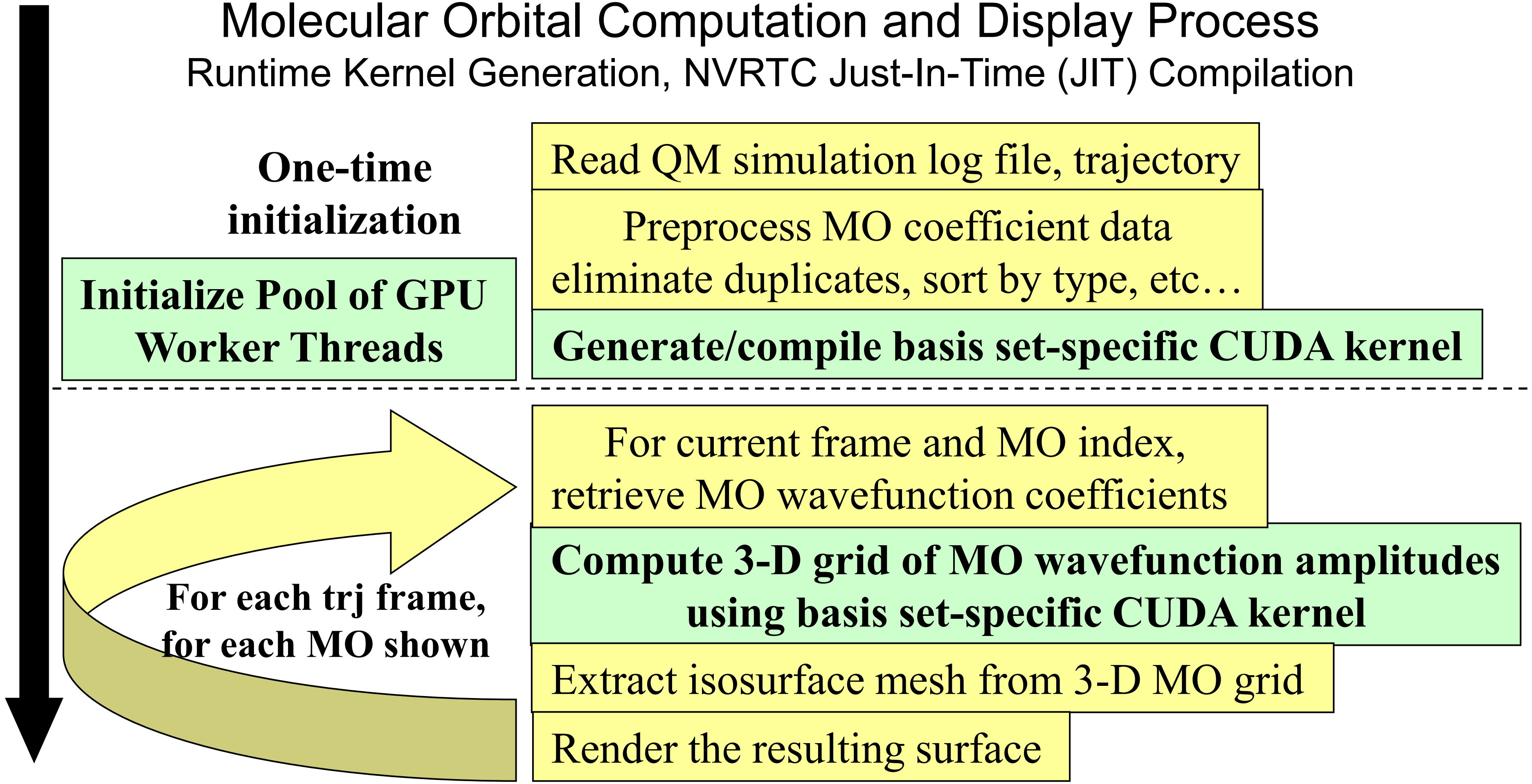
Data-driven execution, but representative loop trip counts in (…)

```
Loop over atoms (1 to ~200) {
    Loop over electron shells for this atom type (1 to ~6) {
        Loop over primitive functions for this shell type (1 to ~6) {
        }
        Loop over angular momenta for this shell type (1 to ~15) {}
    }
}
```

Small loop trip counts result in significant loop overhead. **Runtime kernel generation and NVRTC JIT compilation can achieve in a large (1.8x!) speed boost via loop unrolling, constant folding, elimination of array accesses, …**

Molecular Orbital Computation and Display Process

Runtime Kernel Generation, NVRTC Just-In-Time (JIT) Compilation


One-time initialization
Initialize Pool of GPU Worker Threads
Read QM simulation log file, trajectory
Preprocess MO coefficient data eliminate duplicates, sort by type, etc…
Generate/compile basis set-specific CUDA kernel
For each trj frame, for each MO shown
For current frame and MO index, retrieve MO wavefunction coefficients
Compute 3-D grid of MO wavefunction amplitudes using basis set-specific CUDA kernel
Extract isosurface mesh from 3-D MO grid
Render the resulting surface

```
for (shell=0; shell < maxshell; shell++)  {
  float contracted_gto = 0.0f;


  // Loop over the Gaussian primitives of CGTO
  int maxprim = const_num_prim_per_shell[shell_counter];
  int shell_type = const_shell_symmetry[shell_counter];
  for (prim=0; prim < maxprim;  prim++) {
    float exponent          = const_basis_array[prim_counter      ];
    float contract_coeff = const_basis_array[prim_counter + 1];
    contracted_gto += contract_coeff  * expf(-exponent*dist2);
    prim_counter += 2;
  }
```

General loop-based data-dependent MO CUDA kernel

Runtime-generated data-specific MO CUDA kernel compiled via **CUDA NVRTC** JIT…

```
contracted_gto = 1.832937 * expf(-7.868272*dist2);
contracted_gto += 1.405380 * expf(-1.881289*dist2);
contracted_gto += 0.701383 * expf(-0.544249*dist2);
```

1.8x Faster

General loop-based data-dependent MO CUDA kernel

```
  for (shell=0; shell < maxshell; shell++)  {
    float contracted_gto = 0.0f;

    // Loop over the Gaussian primitives of CGTO
    int maxprim = const_num_prim_per_shell[shell_counter];
    int shell_type = const_shell_symmetry[shell_counter];
    for (prim=0; prim < maxprim;  prim++) {
      float exponent         = const_basis_array[prim_counter      ];
      float contract_coeff = const_basis_array[prim_counter + 1];
      contracted_gto += contract_coeff  * expf(-exponent*dist2);
      prim_counter += 2;
    }

    float tmpshell=0;
    switch (shell_type) {
      case S_SHELL:
        value += const_wave_f[ifunc++] * contracted_gto;
        break;
[…..]
      case D_SHELL:
        tmpshell += const_wave_f[ifunc++] * xdist2;
        tmpshell += const_wave_f[ifunc++] * ydist2;
        tmpshell += const_wave_f[ifunc++] * zdist2;
        tmpshell += const_wave_f[ifunc++] * xdist * ydist;
```

Runtime-generated data-specific MO CUDA kernel compiled via **CUDA NVRTC JIT…**

```
contracted_gto = 1.832937 * expf(-7.868272*dist2);
contracted_gto += 1.405380 * expf(-1.881289*dist2);
contracted_gto += 0.701383 * expf(-0.544249*dist2);
// P_SHELL
tmpshell = const_wave_f[ifunc++] * xdist;
tmpshell += const_wave_f[ifunc++] * ydist;
tmpshell += const_wave_f[ifunc++] * zdist;
value += tmpshell * contracted_gto;

contracted_gto = 0.187618 * expf(-0.168714*dist2);
// S_SHELL
value += const_wave_f[ifunc++] * contracted_gto;

contracted_gto = 0.217969 * expf(-0.168714*dist2);
// P_SHELL
tmpshell = const_wave_f[ifunc++] * xdist;
tmpshell += const_wave_f[ifunc++] * ydist;
tmpshell += const_wave_f[ifunc++] * zdist;
value += tmpshell * contracted_gto;

contracted_gto = 3.858403 * expf(-0.800000*dist2);
// D_SHELL
tmpshell = const_wave_f[ifunc++] * xdist2;
tmpshell += const_wave_f[ifunc++] * ydist2;
```

1.8x Faster

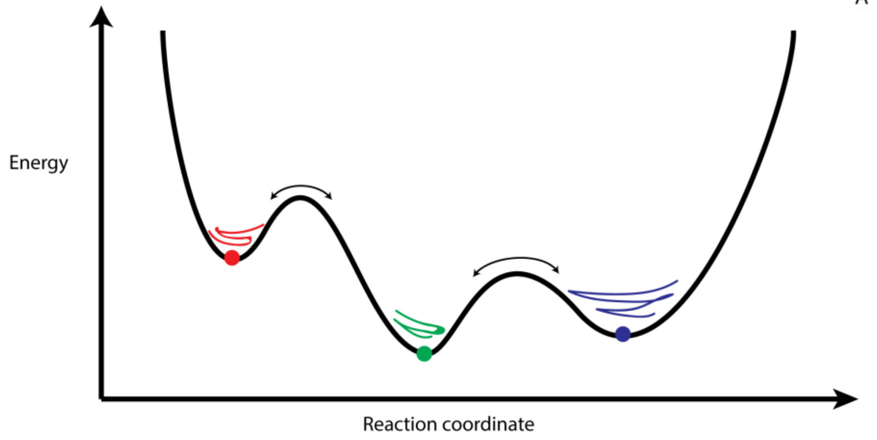
Challenges Adapting Large Software Systems for State-of-the-Art Hardware Platforms

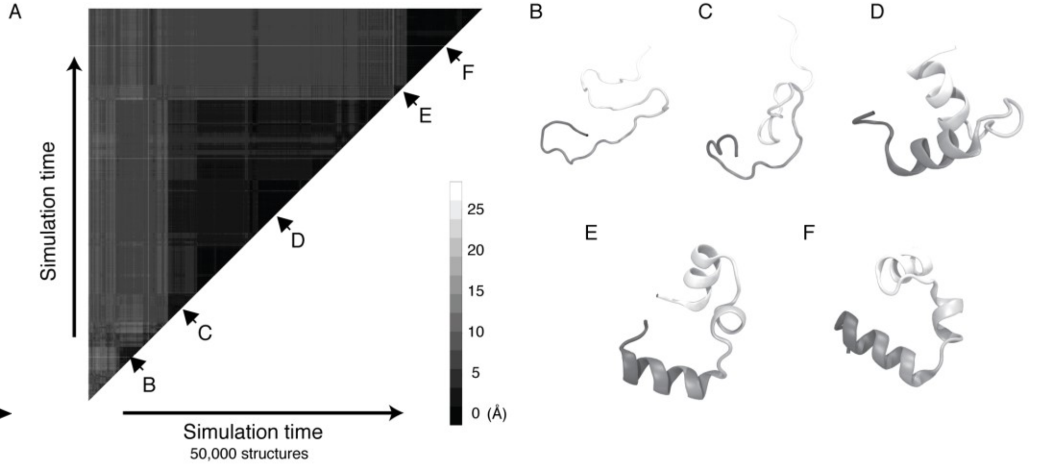
- Initial focus on key computational kernels eventually gives way to the need to optimize an **ocean of less critical routines**, due to observance of Amdahl’s Law
- Even though these less critical routines might be easily ported to CUDA or similar, the sheer number of routines often poses a challenge
- Need a low-cost approach for **getting “some” speedup** out of these second-tier routines
- In many cases, it is completely **sufficient to achieve memory-bandwidth-bound GPU performance with an existing algorithm**

Directive-Based Parallel Programming with OpenACC

- Annotate loop nests in existing code with #pragma compiler directives:
  - Annotate opportunities for parallelism
  - Annotate points where host-GPU memory transfers are best performed, indicate propagation of data
- Evolve original code structure to improve efficacy of parallelization
  - Eliminate false dependencies between loop iterations
  - Revise algorithms or constructs that create excess data movement

Clustering Analysis of Molecular Dynamics Trajectories





GPU-Accelerated Molecular Dynamics Clustering Analysis with OpenACC. J.E. Stone, J.R. Perilla, C. K. Cassidy, and K. Schulten. In, Robert Farber, ed., Parallel Programming with OpenACC, Morgan Kaufmann, Chapter 11, pp. 215-240, 2016.

Serial QCP RMSD Inner Product Loop

- Simple example where directive based parallelism can be applied easily and effectively
- Such a loop is inherently a memory-bandwidth-bound algorithm, so that's the goal for acceleration

```
for (int l=0; l<cnt; l++) {
  double x1, x2, y1, y2, z1, z2;
  x1 = crdx1[l];
  y1 = crdy1[l];
  z1 = crdz1[l];

  G1 += x1*x1 + y1*y1 + z1*z1;

  x2 = crdx2[l];
  y2 = crdy2[l];
  z2 = crdz2[l];

  G2 += x2*x2 + y2*y2 + z2*z2;

  a0 += x1 * x2;
  a1 += x1 * y2;
  a2 += x1 * z2;

  a3 += y1 * x2;
  a4 += y1 * y2;
  a5 += y1 * z2;

  a6 += z1 * x2;
  a7 += z1 * y2;
  a8 += z1 * z2;
}
```

OpenACC QCP RMSD Inner Product Loop

- Simple example where directive based parallelism can be applied easily and effectively
- Such a loop is inherently a memory-bandwidth-bound algorithm, so that's the goal for acceleration

```
// excerpted code that has been abridged for brevity…
void rmsdmat_qcp_acc(int cnt, int padcnt, int framecrdsz,
                     int framecount, const float * restrict crds,
long i, j, k;
#pragma acc kernels copyin(crds[0:tsz]), copy(rmsdmat[0:msz])
 for (k=0; k<(framecount*(framecount-1))/2; k++) {
   // compute triangular matrix index 'k' in a helper function
   // to ensure that the compiler doesn't think that we have
   // conflicts or dependencies between loop iterations
   acc_idx2sub_tril(long(framecount-1), k, &i, &j);
   long x1addr = j * 3L * framecrdsz;
   long x2addr = i * 3L * framecrdsz;

#pragma acc loop vector(256)
   for (long l=0; l<cnt; l++) {
   // abridged for brevity ...

   rmsdmat[k]=rmsd; // store linearized triangular matrix
 }
}
```

OpenACC QCP RMSD Inner Product Loop Performance Results

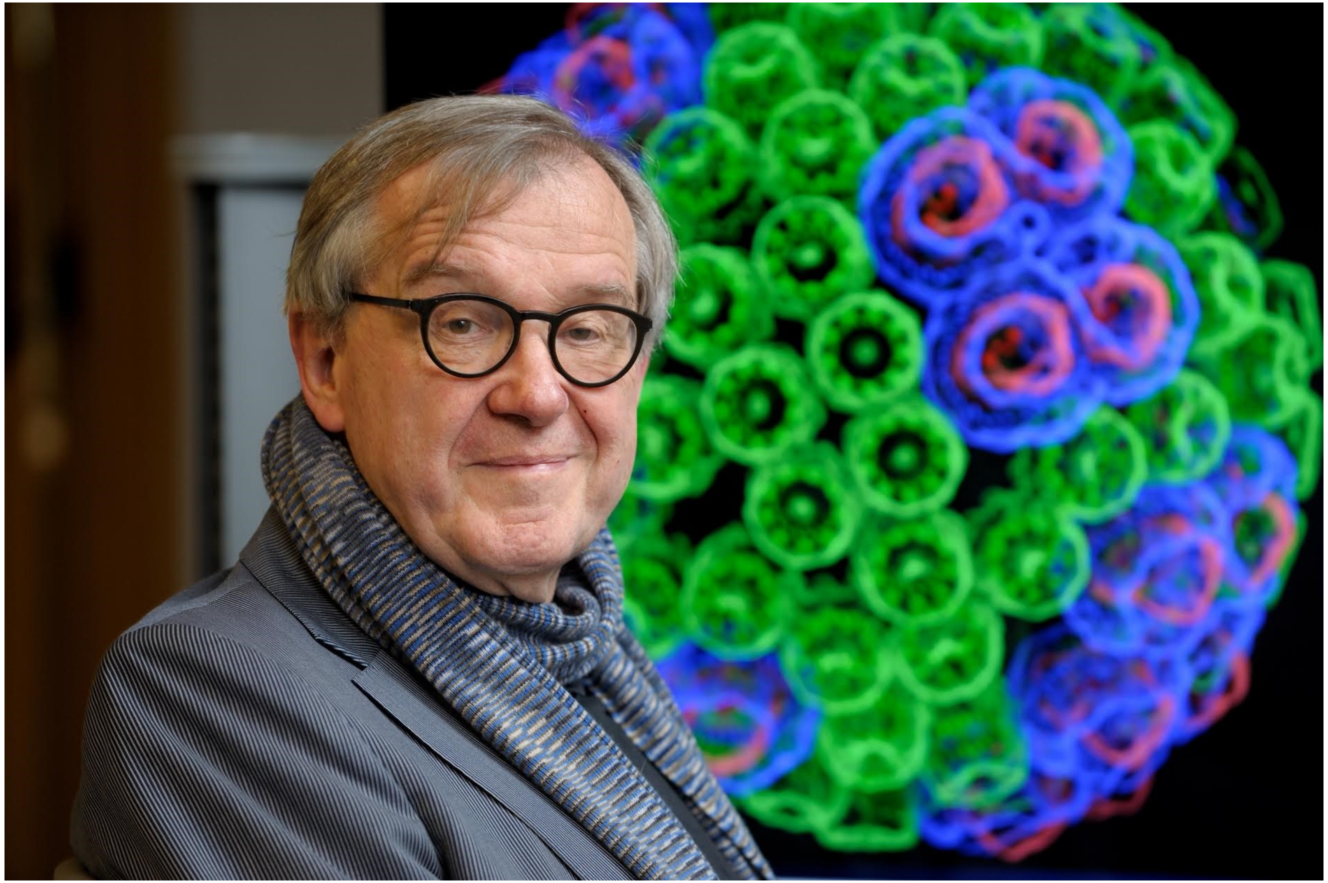
- Xeon 2867W v3, w/ hand-coded AVX and FMA intrinsics: 20.7s
- Tesla K80 w/ OpenACC: **6.5s (3.2x speedup)**
- OpenACC on K80 achieved 65% of theoretical peak memory bandwidth, with 2016 compiler and just a few lines of #pragma directives. Excellent speedup for minimal changes to code.
- Future OpenACC compiler revs should provide higher performance yet

Acknowledgements

- Theoretical and Computational Biophysics Group, University of Illinois at Urbana-Champaign
- NVIDIA CUDA and OptiX teams
- Funding:
  - NIH support: P41GM104601
  - DOE INCITE, ORNL Titan: DE-AC05-00OR22725
  - NSF Blue Waters:
    NSF OCI 07-25070, PRAC "The Computational Microscope",
    ACI-1238993, ACI-1440026

Please See These Other Talks:

- S8727 - Improving NAMD Performance on Volta GPUs
- S8718 - Optimizing HPC Simulation and Visualization Codes using Nsight Systems
- S8747 - ORNL Summit: Petascale Molecular Dynamics Simulations on the Summit POWER9/Volta Supercomputer
- SE150572 - OpenACC User Group Meeting
- S8665: VMD: Biomolecular Visualization from Atoms to Cells Using Ray Tracing, Rasterization, and VR



“When I was a young man, my goal was to look with mathematical and computational means at the inside of cells, one atom at a time, to decipher how living systems work. That is what I strived for and I never deflected from this goal.” – Klaus Schulten

Related Publications

http://www.ks.uiuc.edu/Research/gpu/

- **NAMD goes quantum: An integrative suite for hybrid simulations.** Melo, M. C. R.; Bernardi, R. C.; Rudack T.; Scheurer, M.; Riplinger, C.; Phillips, J. C.; Maia, J. D. C.; Rocha, G. D.; Ribeiro, J. V.; Stone, J. E.; Neese, F.; Schulten, K.; Luthey-Schulten, Z.; Nature Methods, 2018. (**In press)**
- **Challenges of Integrating Stochastic Dynamics and Cryo-electron Tomograms in Whole-cell Simulations.** T. M. Earnest, R. Watanabe, J. E. Stone, J. Mahamid, W. Baumeister, E. Villa, and Z. Luthey-Schulten. J. Physical Chemistry B, 121(15): 3871-3881, 2017.
- **Early Experiences Porting the NAMD and VMD Molecular Simulation and Analysis Software to GPU-Accelerated OpenPOWER Platforms.** J. E. Stone, A.-P. Hynninen, J. C. Phillips, and K. Schulten. International Workshop on OpenPOWER for HPC (IWOPH'16), LNCS 9945, pp. 188-206, 2016.
- **Immersive Molecular Visualization with Omnidirectional Stereoscopic Ray Tracing and Remote Rendering.** J. E. Stone, W. R. Sherman, and K. Schulten. High Performance Data Analysis and Visualization Workshop, IEEE International Parallel and Distributed Processing Symposium Workshop (IPDPSW), pp. 1048-1057, 2016.
- **High Performance Molecular Visualization: In-Situ and Parallel Rendering with EGL.** J. E. Stone, P. Messmer, R. Sisneros, and K. Schulten. High Performance Data Analysis and Visualization Workshop, IEEE International Parallel and Distributed Processing Symposium Workshop (IPDPSW), pp. 1014-1023, 2016.
- **Evaluation of Emerging Energy-Efficient Heterogeneous Computing Platforms for Biomolecular and Cellular Simulation Workloads.** J. E. Stone, M. J. Hallock, J. C. Phillips, J. R. Peterson, Z. Luthey-Schulten, and K. Schulten.25th International Heterogeneity in Computing Workshop, IEEE International Parallel and Distributed Processing Symposium Workshop (IPDPSW), pp. 89-100, 2016.

Related Publications

http://www.ks.uiuc.edu/Research/gpu/

- **Atomic Detail Visualization of Photosynthetic Membranes with GPU-Accelerated Ray Tracing.** J. E. Stone, M. Sener, K. L. Vandivort, A. Barragan, A. Singharoy, I. Teo, J. V. Ribeiro, B. Isralewitz, B. Liu, B.-C. Goh, J. C. Phillips, C. MacGregor-Chatwin, M. P. Johnson, L. F. Kourkoutis, C. Neil Hunter, and K. Schulten. J. Parallel Computing, 55:17-27, 2016.
- **Chemical Visualization of Human Pathogens: the Retroviral Capsids.** Juan R. Perilla, Boon Chong Goh, John E. Stone, and Klaus Schulten. SC'15 Visualization and Data Analytics Showcase, 2015.
- **Visualization of Energy Conversion Processes in a Light Harvesting Organelle at Atomic Detail.** M. Sener, J. E. Stone, A. Barragan, A. Singharoy, I. Teo, K. L. Vandivort, B. Isralewitz, B. Liu, B. Goh, J. C. Phillips, L. F. Kourkoutis, C. N. Hunter, and K. Schulten. SC'14 Visualization and Data Analytics Showcase, 2014.
  ***<b>Winner of the SC'14 Visualization and Data Analytics Showcase</b>
- **Runtime and Architecture Support for Efficient Data Exchange in Multi-Accelerator Applications.** J. Cabezas, I. Gelado, J. E. Stone, N. Navarro, D. B. Kirk, and W. Hwu. IEEE Transactions on Parallel and Distributed Systems, 26(5):1405-1418, 2015.
- **Unlocking the Full Potential of the Cray XK7 Accelerator.** M. D. Klein and J. E. Stone. Cray Users Group, Lugano Switzerland, May 2014.
- **GPU-Accelerated Analysis and Visualization of Large Structures Solved by Molecular Dynamics Flexible Fitting.** J. E. Stone, R. McGreevy, B. Isralewitz, and K. Schulten. Faraday Discussions, 169:265-283, 2014.
- **Simulation of reaction diffusion processes over biologically relevant size and time scales using multi-GPU workstations**. M. J. Hallock, J. E. Stone, E. Roberts, C. Fry, and Z. Luthey-Schulten. Journal of Parallel Computing, 40:86-99, 2014.

Related Publications
http://www.ks.uiuc.edu/Research/gpu/

- **GPU-Accelerated Molecular Visualization on Petascale Supercomputing Platforms.** J. Stone, K. L. Vandivort, and K. Schulten. UltraVis'13: Proceedings of the 8th International Workshop on Ultrascale Visualization, pp. 6:1-6:8, 2013.
- **Early Experiences Scaling VMD Molecular Visualization and Analysis Jobs on Blue Waters.** J. Stone, B. Isralewitz, and K. Schulten. In proceedings, Extreme Scaling Workshop, 2013.
- **Lattice Microbes: High-performance stochastic simulation method for the reaction-diffusion master equation.** E. Roberts, J. Stone, and Z. Luthey-Schulten. J. Computational Chemistry 34 (3), 245-255, 2013**.**
- **Fast Visualization of Gaussian Density Surfaces for Molecular Dynamics and Particle System Trajectories.** M. Krone, J. Stone, T. Ertl, and K. Schulten. *EuroVis Short Papers,* pp. 67-71, 2012.
- **Immersive Out-of-Core Visualization of Large-Size and Long-Timescale Molecular Dynamics Trajectories.** J. Stone, K. L. Vandivort, and K. Schulten. G. Bebis et al. (Eds.): *7th International Symposium on Visual Computing (ISVC 2011)*, LNCS 6939, pp. 1-12, 2011.
- **Fast Analysis of Molecular Dynamics Trajectories with Graphics Processing Units – Radial Distribution Functions.** B. Levine, J. Stone, and A. Kohlmeyer. *J. Comp. Physics*, 230(9):3556-3569, 2011.

Related Publications
http://www.ks.uiuc.edu/Research/gpu/

- **Quantifying the Impact of GPUs on Performance and Energy Efficiency in HPC Clusters.** J. Enos, C. Steffen, J. Fullop, M. Showerman, G. Shi, K. Esler, V. Kindratenko, J. Stone, J Phillips. *International Conference on Green Computing,* pp. 317-324, 2010.
- **GPU-accelerated molecular modeling coming of age.** J. Stone, D. Hardy, I. Ufimtsev, K. Schulten. *J. Molecular Graphics and Modeling,* 29:116-125, 2010.
- **OpenCL: A Parallel Programming Standard for Heterogeneous Computing.** J. Stone, D. Gohara, G. Shi. *Computing in Science and Engineering,* 12(3):66-73, 2010.
- **An Asymmetric Distributed Shared Memory Model for Heterogeneous Computing Systems**. I. Gelado, J. Stone, J. Cabezas, S. Patel, N. Navarro, W. Hwu. *ASPLOS '10: Proceedings of the 15th International Conference on Architectural Support for Programming Languages and Operating Systems,* pp. 347-358, 2010.

Related Publications
http://www.ks.uiuc.edu/Research/gpu/

- **GPU Clusters for High Performance Computing**. V. Kindratenko, J. Enos, G. Shi, M. Showerman, G. Arnold, J. Stone, J. Phillips, W. Hwu. *Workshop on Parallel Programming on Accelerator Clusters (PPAC),* In Proceedings IEEE Cluster 2009, pp. 1-8, Aug. 2009.
- **Long time-scale simulations of in vivo diffusion using GPU hardware**. E. Roberts, J. Stone, L. Sepulveda, W. Hwu, Z. Luthey-Schulten. In *IPDPS'09: Proceedings of the 2009 IEEE International Symposium on Parallel & Distributed Computing*, pp. 1-8, 2009.
- **High Performance Computation and Interactive Display of Molecular Orbitals on GPUs and Multi-core CPUs**. J. E. Stone, J. Saam, D. Hardy, K. Vandivort, W. Hwu, K. Schulten, *2nd Workshop on General-Purpose Computation on Graphics Pricessing Units (GPGPU-2), ACM International Conference Proceeding Series*, volume 383, pp. 9-18, 2009.
- **Probing Biomolecular Machines with Graphics Processors**. J. Phillips, J. Stone. *Communications of the ACM,* 52(10):34-41, 2009.
- **Multilevel summation of electrostatic potentials using graphics processing units**. D. Hardy, J. Stone, K. Schulten. *J. Parallel Computing*, 35:164-177, 2009.

Related Publications
http://www.ks.uiuc.edu/Research/gpu/

- **Adapting a message-driven parallel application to GPU-accelerated clusters**. J. Phillips, J. Stone, K. Schulten. *Proceedings of the 2008 ACM/IEEE Conference on Supercomputing*, IEEE Press, 2008.
- **GPU acceleration of cutoff pair potentials for molecular modeling applications**. C. Rodrigues, D. Hardy, J. Stone, K. Schulten, and W. Hwu. *Proceedings of the 2008 Conference On Computing Frontiers*, pp. 273-282, 2008.
- **GPU computing**. J. Owens, M. Houston, D. Luebke, S. Green, J. Stone, J. Phillips. *Proceedings of the IEEE*, 96:879-899, 2008.
- **Accelerating molecular modeling applications with graphics processors**. J. Stone, J. Phillips, P. Freddolino, D. Hardy, L. Trabuco, K. Schulten. *J. Comp. Chem.*, 28:2618-2640, 2007.
- **Continuous fluorescence microphotolysis and correlation spectroscopy**. A. Arkhipov, J. Hüve, M. Kahms, R. Peters, K. Schulten. *Biophysical Journal*, 93:4006-4017, 2007.