Attacking HIV with Petascale Molecular Dynamics Simulations on Titan and Blue Waters

James Phillips
Beckman Institute, University of Illinois
http://www.ks.uiuc.edu/Research/namd/
HIV Infective Cycle

- Binding
- Fusion
- Budding
- Capsid uncoating
- Nuclear Import
- Integration into the host’s chromatin

Virion

Host Cell

Biomedical Technology Research Center for Macromolecular Modeling and Bioinformatics
Beckman Institute, University of Illinois at Urbana-Champaign - www.ks.uiuc.edu

GTC 2015
HIV Treatment

Fusion/Entry inhibitors

Currently no drug targets capsid uncoating or nuclear import!

Reverse Transcription (RNA to DNA) inhibitors

Integrase inhibitors

Host Cell

Protease inhibitors
HIV Capsid is Much Larger than Previously Simulated Systems

Collaborators:
Peijun Zhang, Angela Gronenborn - U. Pittsburgh
Christopher Aiken - Vanderbilt U.
G. Zhao, et al. Nature 497 (2013); exp + comp

Coarse-grained only!

lysozyme
ribosome

All five referees demanded:
Only coarse-grained, not all-atom!
HIV-1 virion

186 hexamers
12 pentamers
Modeling of the Hexameric Lattice using Molecular Dynamics Flexible Fitting


*Key person:*

Juan Perilla
(UIUC)
Modeling of the Hexameric Lattice using Molecular Dynamics Flexible Fitting


Key person: Juan Perilla (UIUC)
MD Simulation Furnishes Atom-Level Structure of Pentamer-of-Hexamers

Closed capsid is made of hexamers-of-hexamers pentamers-of-hexamers

1.5 µs (1.3 M atoms) simulation of pentameric center
HIV capsid contains 186-1300+ proteins,
One-Microsecond Simulation Includes 64 Million Atoms

Stable!

Key person:
Juan Perilla
(UIUC)
2013 HPCwire Editors’ Choice Award for Best Use of HPC in Life Sciences
Malleability of HIV-1 CA

Hexamer of hexamers bite angles along chiral axis

1300 proteins in different conformations

Native capsid bite angle distribution

Curvature is regulated by the trimer interface


Peijun Zhang - U. Pittsburgh
Capsid acts as an osmotic regulator

Results from 64 M atom, 1 μs molecular dynamics simulation!

Chloride ions permeate through the hexameric center
HIV-1 infection

HIV-1 uncoating: regulation by host factors

Host cell prevents infection by inducing premature uncoating


Cytoplasm

Cell factors interacting with HIV capsid!

Nucleus

- CypA
- TNPO3
- CPSF6
- NUP153
- NUP358
- rhTRIM5α
- TRIMCyp
- Inhibitor
- MX2
CypA Bridge Model MD Simulations Identify a Novel Catalytic Site

only polarizable force field yields stable bridge interaction

interaction confirmed by NMR
Competitive binding between CypA and TRIM

cypA binding pattern prevents TRIM binding, but leaves Nup interactions intact

Key person: Juan Perilla (UIUC)

F. Diaz-Griffero, Viruses (2011)
Chemical Detail (Every Atom) is Essential for Capsid Role

**Not always listen to referees!**

Don’t simplify before you understand!

Ions permeate through the capsid.

Curvature regulated by trimeric interface.

CypA bridges adjacent capsid subunits and thereby binds in particular pattern on capsid surface.
HIV Acknowledgments

Juan R. Perilla  Klaus Schulten  Peijun Zhang  Angela M. Gronenborn
Theoretical and Computational Biophysics Group  University of Pittsburgh School of Medicine

Laxmikant Kale  Christopher Aiken
Parallel Programming Lab  Department of Pathology and Immunology
Dept. of Computer Science  Vanderbilt University School of Medicine

University of Illinois at Urbana-Champaign

Center for the Physics of Living Cells

Biomedical Technology Research Center for Macromolecular Modeling and Bioinformatics
Beckman Institute, University of Illinois at Urbana-Champaign - www.ks.uiuc.edu
Developers of the widely used computational biology software VMD and NAMD

250,000 registered VMD users
72,000 registered NAMD users

600 publications (since 1972) over 54,000 citations

Research projects include: virus capsids, ribosome, photosynthesis, protein folding, membrane reshaping, animal magnetoreception

Achievements Built on People

5 faculty members
8 developers
1 systems administrator
17 postdocs
46 graduate students
3 administrative staff

Tajkorshid, Luthey-Schulten, Stone, Schulten, Phillips, Kale, Mallon
NAMD Serves NIH Users and Goals

Practical Supercomputing for Biomedical Research

• 72,000 users can’t all be computer experts.
  – 18% are NIH-funded; many in other countries.
  – 21,000 have downloaded more than one version.
  – 5000 citations of NAMD reference papers.

• One program available on all platforms.
  – Desktops and laptops – setup and testing
  – Linux clusters – affordable local workhorses
  – Supercomputers – free allocations on XSEDE
  – Blue Waters – sustained petaflop/s performance
  – GPUs - next-generation supercomputing

• User knowledge is preserved across platforms.
  – No change in input or output files.
  – Run any simulation on any number of cores.

• Available free of charge to all.
100 Million Atom Simulations Are Not Routine

- Simulation setup is a black art
  - Tools for adding solvent and ions don’t scale
  - Need to move tools and users towards new “js” file format
- Still some rough edges
  - Not all NAMD features usable at scale
- Trajectory and restart output performance
  - New Charm++ I/O library will help address this
- Simulations require leadership machines
  - Available resources are limited, allocation process is slow
- Lack of setup/visualization/analysis facilities
NIH Center Facilities Enable Petascale Biology

Over the past six years the Center has assembled all necessary hardware and infrastructure to prepare and analyze petascale molecular dynamics simulations, and makes these facilities available to visiting researchers.
Virtual Facilities Enable Petascale Anywhere

High-end visualization and analysis workstations currently available only in person at the Beckman Institute must be virtualized and embedded at supercomputer centers.

1 Gigabit Network

Compressed Video

Storage

Compute

Visualization
Remote Visualization Now

- TACC Stampede supports this today
  - Includes nodes with 1TB memory
  - Not virtualized, allocate full dedicated node
  - New Maverick cluster added
- Blue Waters – no visualization resource
- Titan – new Rhea “viz” cluster drops GPUs
- NIH Center - using NICE DCV for remote access
Jim Phillips monitors NAMD performance of thousands of cores on group’s 4K graphics system.
Focus on enabling petascale simulations
Type 1: Large systems of ~100 million atoms
- Scalable to all of Blue Waters or Titan (Phillips et al., SC14)
- In regular production use for multiple biomedical driving projects
- Amaro (UCSD) allocation on Blue Waters for 210M-atom influenza virus
Type 2: Replica exchange simulations of smaller systems
- Improved performance over NAMD 2.9, especially with GPUs
- Scalable multiple copy algorithms, *Comp. Phys. Comm.* 185:908-16
Various other improvements
- Xeon Phi port, GPU improvements including PME offload
- Semi- and non-periodic long-range electrostatics (multilevel summation)
NAMD Replica Exchange Example Application: Complete Description of Transport Cycle

Advanced Replica Exchange Simulation Protocol
Requiring a Combination of \textbf{Multiple Collective Variables}

Mahmoud Moradi
NAMD is based on Charm++

Complete info at charmplusplus.org
Charm++ Used by NAMD

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


- Spatially decompose data and communication.
- Separate but related work decomposition.
- “Compute objects” facilitate iterative, measurement-based load balancing system.
NAMD Overlapping Execution

Phillips et al., SC2002.

Objects are assigned to processors and queued as data arrives.
Overlapping GPU and CPU with Communication

Phillips et al., SC2008
Actual Timelines from NAMD
Generated using Charm++ tool “Projections” http://charm.cs.uiuc.edu/
Non-bonded remote kernel
Non-bonded local kernel
Integration
Incoming positions
Bonded on CPU
Results from GPU
Inter-node communication
Enabling Remote/Local Overlap

• Asking for priorities since 2008
  – Critical for Charm++ performance on CPU
  – With Kepler we get 1 bit on Tesla/Quadro

• Doesn’t order grid launches:

  • Workaround is small memset in low-priority stream
    – Doesn’t need priorities, so works on GeForce cards too!
Kepler Shuffle Instructions

• Reductions for energy and pressure tensor
• Old implementation limits synchronization:
  – Reduce multiple fields at same time
  – Warp-synchronous for final stages
• Shuffle implementation is simpler and faster!
  – Except now preprocessor code for older devices
  – “diff –D KEPLER_SHUFFLE” is very helpful
Maxwell Performance

Titan X is 60% faster than Titan Black,
30% faster than GTX 980.

NAMD ApoA1 benchmark on 14 cores 2.6 GHz E5-2650 v2 or E5-2660 v3
CUDA 7

• We’ve heard of it.
• Looking forward to C++11.
• Runtime compilation might be awesome.
• We’ve also heard of CUDA 6.5.
• It will be available on Cray XK7 “soon”.
• Until then we’re stuck with CUDA 5.5.
Trends Affecting Performance

• GPU performance increasing
  – Performance limit will be code on CPU
  – Most highly tuned CPU code moved to GPU
  – Remaining CPU code is also less efficient
  – Therefore CPU must run serial code well

• CPU serial performance static

• CPU core counts increasing
Suggested Strategy

• Focus on CPU-side code
  – Port to GPU or optimize/parallelize on CPU
  – Stream results off GPU to increase overlap
  – Use CPUs with best single-thread performance

• Focus on communication
  – Reduce communication overhead on CPU
  – General parallel scalability improvements
  – Map decomposition to machine torus topology
    • Also applies to replica exchange partitions
Torus Adaptation
• Job partitioning for multiple copy sampling algorithms
• Mapping NAMD spatial decomposition domains onto machine torus
• Mapping particle-mesh Ewald (PME) electrostatics onto spatial decomposition

Additional Techniques
• Coarsening of PME grid to reduce long-range communication
• Offloading of PME interpolation onto GPUs
• Removal of implicit synchronization in pressure control algorithm
Irregular Torus Topologies

- IBM Blue Gene L/P/Q provide jobs with complete, regular, power-of-two torus.
- Cray XE/XK job topology is unpredictable.
  - Scheduler works around already running jobs.
  - May not be compact or contiguous.
    - New Blue Waters scheduler addresses this.
  - Even full-machine jobs skip over I/O nodes.
Convert Torus to Optimized Mesh

• Start with Charm++ TopoManager API
  – Provides node coordinates and torus dimensions.

• Extend with TopoManagerWrapper class
  – Ensure same torus coordinates for entire physical node.
  – Shift torus coordinates to eliminate largest gap in node list.
  – Re-order dimensions from longest to shortest occupied span.
  – Provide functions for sorting list of ranks along ordered list of dimensions by “snake scanning” curve (seen on next slide).

• Recursive bisection on these “snake scanning” curves is the basis of all torus-mapping algorithms to follow.
Mapping Charm++ Partitions

Gaps in torus

Bisection 1

Bisection 2
Mapping NAMD Spatial Domains

• Priorities are:
  1. Evenly distributed patch load across available PEs
  2. Compact patch set within physical node to minimize communication
  3. Torus topology adaptation – only impacts largest runs

• Simultaneous recursive bisection of patch mesh and PE mesh:
  – Re-order patch and PE mesh dimensions longest to shortest.
  – When dividing PEs, divide patches along corresponding dimension, if possible, before falling back to next-longest dimension.
  – Divide PEs on physical node boundaries.
  – Divide patches to balance load with at least one patch per PE.

• Within physical node, sort patches along PME slabs/pencils.
Mapping NAMD Spatial Domains

Processors

Patches

Gaps in torus
Mapping NAMD Spatial Domains

Processes

Gaps in torus

Patches
Mapping PME Electrostatics

• Want to align X-Y grid of Z pencils to patches.
  – Needs to work even on non-torus machines.
• Assign X-Y coordinates to PEs.
  – Average coordinate of patches on PE (or node, etc.)
• Recursively bisect…
  – Z pencils on longer dimension boundary (5x5=2x5+3x5).
  – PEs proportionately (25=10+15) on same coordinate.
• Optimize Y-X-Y FFT transposes by placing X and Y pencils with same Z coordinate on contiguous ranks.
NAMD PME CUDA Kernel

• Bottleneck for 100M atoms is PME FFT communication
  – Switch from 4\textsuperscript{th}-order to 8\textsuperscript{th}-order interpolation on coarser grid
• Doing 8\textsuperscript{th}-order PME on GPU improves critical path
• CPU may be bottleneck for 8\textsuperscript{th}-order PME
  – Especially as GPU non-bonded gets faster…
• Simplest design that might possibly work:
  – One stream per host PE (preserve control flow)
  – One atom per warp with warp-synchronous programming
  – Atomics to accumulate charge grid in global memory
    • One per thread so accesses coalesce
    • Also build “used” flags arrays for x-y pencils and z plane
PME Kernel Aggregation

- Initial version slower than PME on CPU
- First, one launch per PE, not per patch
- Second, one charge array per node
  - First version to beat PME on CPU
  - Node-level coordination a challenge in Charm++
  - Reduces number of messages sent per node!
    - Need to backport to PME on CPU version
    - May help CPU-only version, but not as much
Results from GPU

Integration

Bonded on CPU

Incoming positions

Non-bonded remote kernel

Non-bonded local kernel
Non-bonded kernels

PME kernel polling

PME kernel submissions
Non-bonded kernel runs longer to calculate pair-lists

PME kernel priority streams steal slots from non-bonded
Performance Results

• Petascale simulation preparation is not easy.
  – Benchmarks based on 1.06M-atom STMV
  – 5x2x2 grid = 21M atoms ~ “small petascale”
  – 7x6x5 grid = 224M atoms ~ “Influenza virus”

• Experiment by disabling optimizations
  – Only disable one at a time, not cumulatively.
Benchmarking Caution

• Cray XE/XK performance varies due to:
  – Compactness of nodes assigned to job
  – Other jobs running on machine (cross-traffic)
  – I/O activity (more Blue Waters than Titan)

• To test performance impact of changes, run old and new back-to-back in same job.
NAMD Topology Mapping on Titan Cray XK7

Phillips et al., SC14
Other NAMD Optimizations on Titan Cray XK7

(2fs timestep)

Performance (ns per day)

Number of Nodes

With All Optimizations
- Without Reduced Barostat Synchronization
- Without PME Interpolation Offloaded to GPU
- Without 2A PME Grid and 8th-Order Interpolation

Phillips et al., SC14
NAMD on Torus and Non-torus Networks

Performance (ns per day)

(2fs timestep)

Number of Nodes

21M atoms

224M atoms

Titan XK7
Stampede CPU+Phi
Edison XC30
Blue Waters XE6
Stampede CPU only

Phillips et al., SC14
Streaming CPU Results to CPU

- Allows incremental results from a single grid to be processed on CPU before grid finishes on GPU
- Allows merging and prioritizing of remote and local work
- GPU side:
  - Write results to host-mapped memory (also without streaming)
  - __threadfence_system() and __syncthreads()
  - Atomic increment for next output queue location
  - Write result index to output queue
- CPU side:
  - Poll end of output queue (int array) in host memory
Streaming on GPU:

```c
if ( force_ready_queue ) {
    __threadfence_system();
    __syncthreads();
    if (threadIdx.x == 0) {
        int old = atomicInc(force_list_counters,force_lists_size-1);
        force_ready_queue[old] = myPatchPair.patch1_force_list_index;
        __threadfence_system();
    }
}

Polling on host:

while ( -1 != (flindex = force_ready_queue[force_ready_queue_next]) ) {
    force_ready_queue[force_ready_queue_next] = -1;
    ++force_ready_queue_next;
    ...process output flindex...
}
```
Controlling Output Order

- Blocks have widely varying runtimes
  - Input order is not output order

<table>
<thead>
<tr>
<th>1</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6</td>
<td></td>
</tr>
</tbody>
</table>
Output: 2, 3, 4, 5, 1, 7, 8, 6, 9

- Non-streaming was simple, just sort large to small

<table>
<thead>
<tr>
<th>6</th>
<th>7</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>1</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Savings
Controlling Output Order

- First use reversed priorities as input order

\[
\begin{array}{cccc}
9 & 3 & 2 & 1 \\
8 & 7 & 6 & 5 \\
4 & & & \\
\end{array}
\]

Output: 8, 7, 6, 5, 9, 3, 2, 4, 1

- Then reverse output order to use as input

\[
\begin{array}{ccccc}
1 & 2 & 3 & 5 & 6 \\
4 & 7 & 8 & & \\
\end{array}
\]

- Provides good ordering and near-ideal compactness
Controlling Output Order

• Requires very little code to save order

```cpp
if (threadIdx.x == 0 && block_order) {
    int old = atomicInc(force_list_counters + 1, total_block_count - 1);
    block_order[old] = block_begin + blockIdx.x;
}
```

• Does not require measuring block runtimes
• Better than old heuristic ordering
• Streaming wins even on single node!
Controlling Output Order

• But what is optimal output order?
  • Remote before local (same as before)
  • Distribute local across threads
    • Slight preference for GPU host thread
  • Local without remote proxies last
    • Not yet implemented
Non-Streaming Kernel
Streaming Kernel
Non-bonded kernels

PME kernel polling

PME kernel submissions
Non-Streaming Kernel Pairlist Step
Streaming Kernel Pairlist Step
New Streaming Kernel Performance

Number of Nodes

Performance (ns per day)

Blue Waters XK7 (new streaming)
Blue Waters XK7 (no streaming)

21M atoms

(2fs timestep)
Parallelize PME Within Node

(2fs timestep)

Number of Nodes

Performance (ns per day)

Blue Waters XK7 (new streaming, tuned PME)
Blue Waters XK7 (new streaming)
Blue Waters XK7 (no streaming)

21M atoms

2.9 ms/step
60 ns/day

+5%
+10-30%
Blue Waters vs Titan

Number of Nodes

Performance (ns per day)

Blue Waters XK7 (new streaming, tuned PME)
Titan XK7 (new streaming, tuned PME)
Titan XK7 (SC14, old streaming)
Blue Waters XK7 (no streaming)

(2fs timestep)
Topology-Aware Scheduling on Blue Waters

• Map jobs to convex sets to avoid network interference
• NCSA, Cray, Adaptive
• Just enabled January 13
• Most likely explanation for Blue Waters performance advantage over Titan
• See Enos et al., CUG 2014
Blue Waters vs Titan

Blue Waters XK7 (GTC15)  
Titan XK7 (GTC15)  
Titan XK7 (SC14)

(2fs timestep)

Number of Nodes

0.25 0.5 1 2 4 8 16 32 64 256 512 1024 2048 4096 8192 16384

Performance (ns per day)

21M atoms

224M atoms

7 ms/step  
25 ns/day
Comparison with CPU-only Machines

(2fs timestep)

<table>
<thead>
<tr>
<th>Number of Nodes</th>
<th>21M atoms</th>
<th>224M atoms</th>
</tr>
</thead>
</table>

- **Blue Waters XK7 (GTC15)**: +50-200%
- **Titan XK7 (GTC15)**: +100-200%
- **Edison XC30 (SC14)**: +70%
- **Blue Waters XE6 (SC14)**: +50-200%
Conclusions

• In biology chemical detail is critical.
• Remote visualization will be necessary.
• Replica exchange enables long timescales.
• Map decomposition to network topology.
• Stream results from GPU in priority order.
• Bad scheduling harms performance.
Thanks to: NIH, NSF, DOE, NCSA, NVIDIA (Sarah Tariq, Patric Zhao, Sky Wu, Justin Luitjens, Nikolai Sakharnykh), Cray (Sarah Anderson, Ryan Olson), NCSA (Robert Brunner), PPL (Eric Bohm, Yanhua Sun, Gengbin Zheng, Nikhil Jain) and 19 years of NAMD and Charm++ developers and users.

James Phillips
Beckman Institute, University of Illinois
http://www.ks.uiuc.edu/Research/namd/