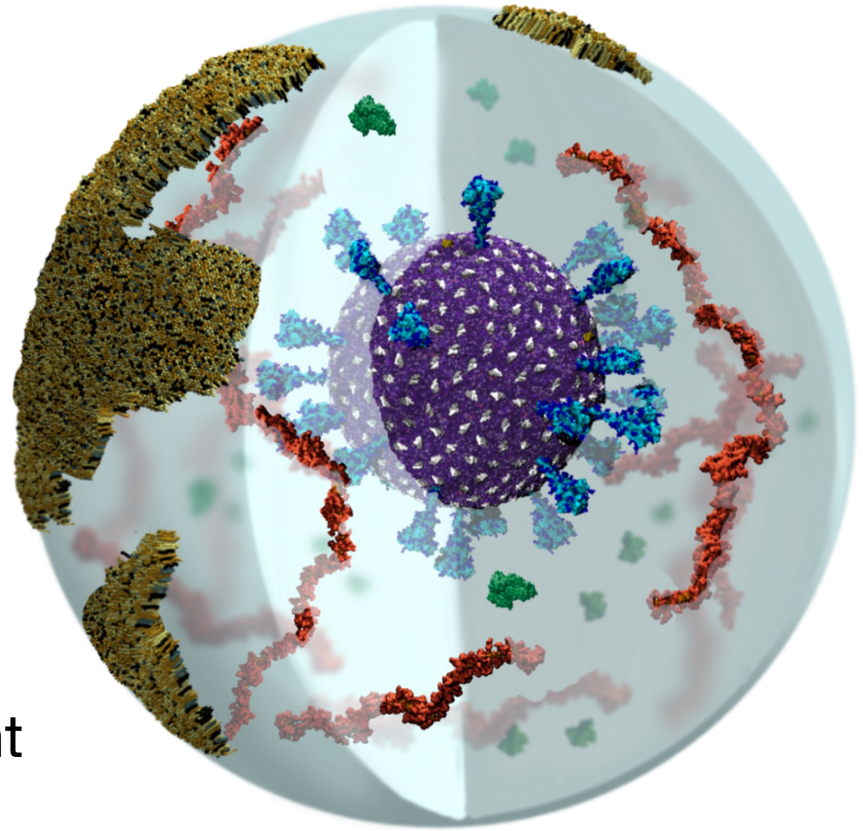
Preparing NAMD for the Aurora Supercomputer

David J. Hardy, University Of Illinois at Urbana-Champaign
Ke Yue, Intel Corporation
Wei Jiang, Argonne National Laboratory

NIH National Institutes of Health

Molecular Dynamics Combats Diseases Like COVID-19

- Molecular dynamics (MD) simulation software and HPC resources provide access to spatial and temporal scales not available to physical experiments
- Atomistic dynamics can reveal the molecular basis for diseases
- By studying viruses and other diseases with MD and related methods, researchers can inform the development of new treatments and therapies



Delta variant of coronavirus (SARS-CoV-2) in aerosol droplet.
Credit: A. Dommer, L. Casalino, F. Kearns, R. Amaro (UCSD).
Simulations (1B atoms) with NAMD, renderings with VMD.

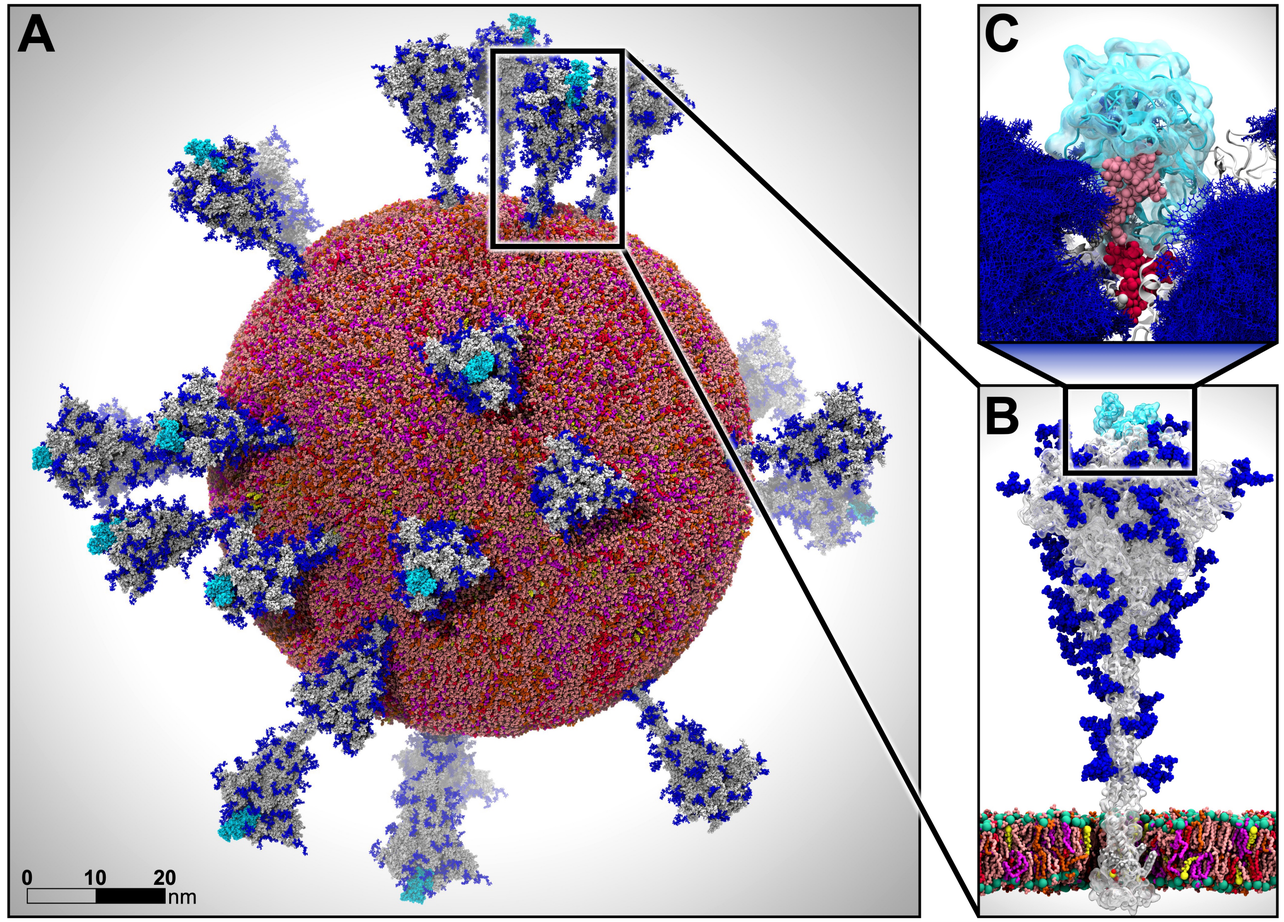
NIH National Institutes of Health

NAMD Simulating SARS-CoV-2 on Frontera and Summit

Collaboration with Amaro Lab at UCSD, images rendered by VMD

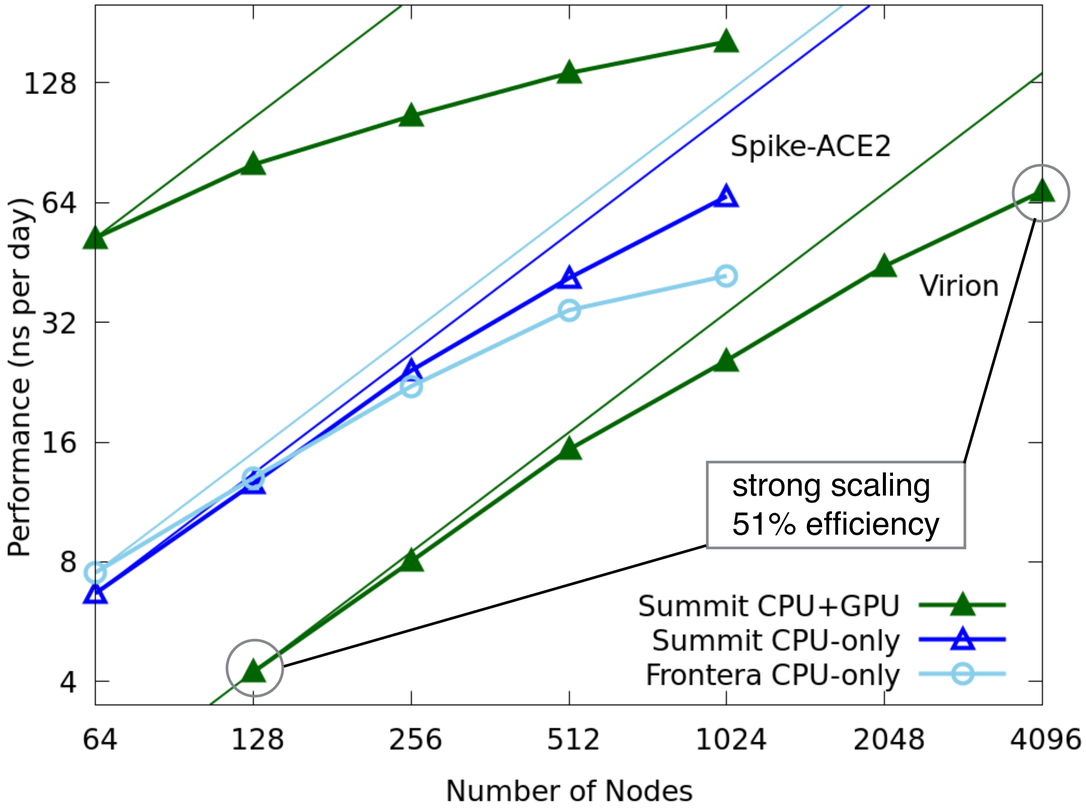
Winner of Gordon Bell Special Prize at SC20, project involved overall 1.13 Zettaflops of NAMD simulation

(A) Virion, (B) Spike, (C) Glycan shield conformations



Scaling performance:

- ~305M atom virion
- ~8.5M atom spike



Casalino, et al. *bioRxiv* (2020) https://doi.org/10.1101/2020.11.19.390187

NIH National Institutes of Health

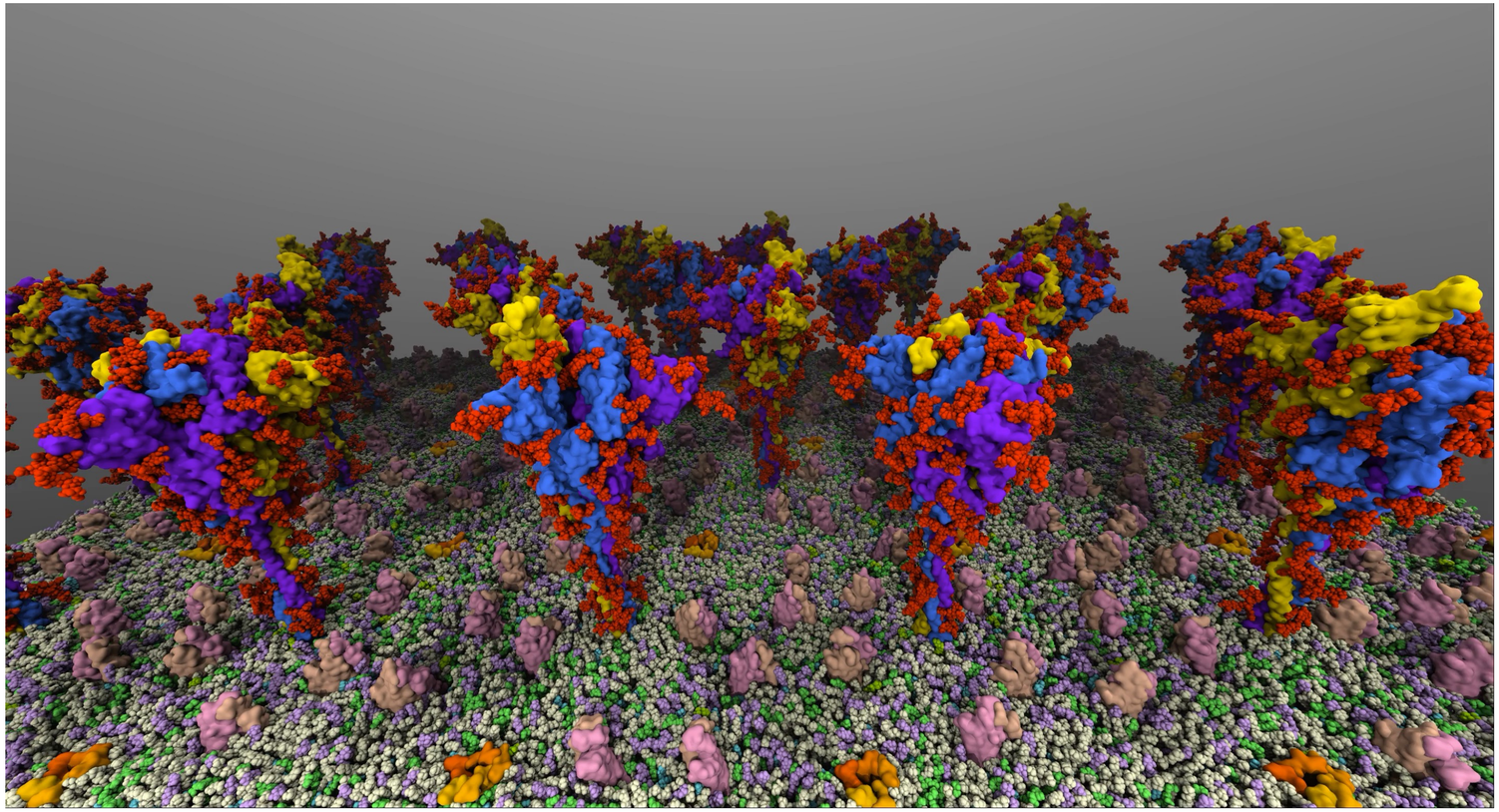
NAMD: Scalable Molecular Dynamics

- Code written in C++ with Charm++ parallel objects
  - CUDA for NVIDIA devices
  - HIP (via Hipify) for AMD devices
  - **oneAPI SYCL for all devices**
- Simulate movements of biomolecules over time
- Enable parallel scaling
  - **Large systems (single-copy scaling)**
  - **Enhanced sampling (multi-copy scaling)**
- Over 25,000 registered users, over 16,000 citations

https://www.ks.uiuc.edu/Research/namd/

Phillips, et al. *J. Comput. Chem.* 26, 1781-1802 (2005)

Phillips, et al. *J. Chem. Phys.* 153, 044130 (2020)

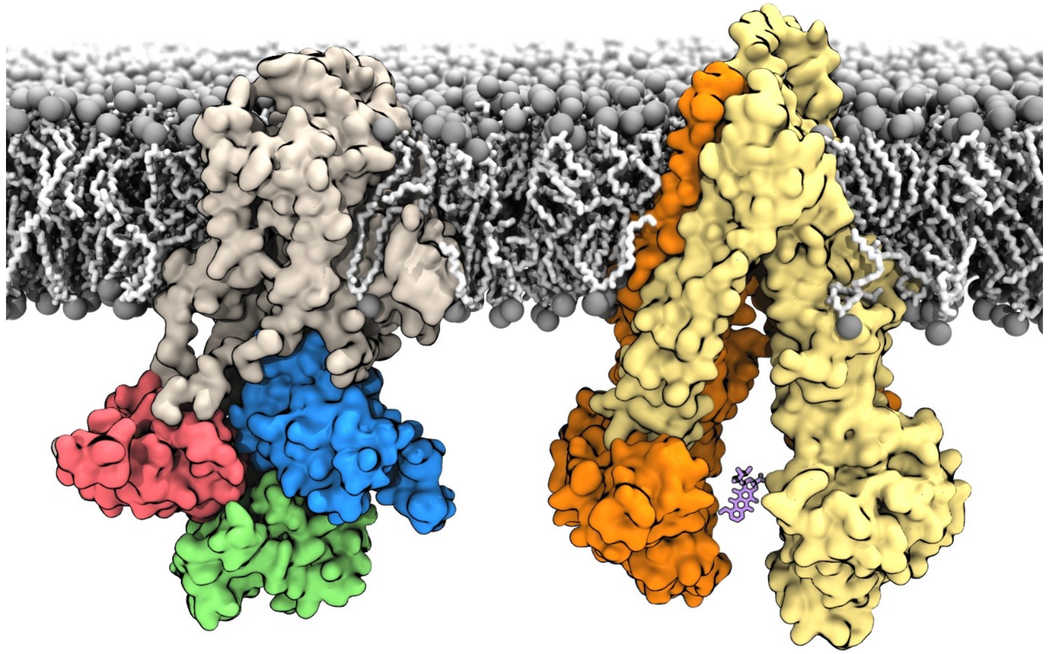


Investigations of coronavirus (SARS-CoV-2) spike dynamics.
Credit: Tianle Chen, Karanpal Kapoor, Emad Tajkhorshid (UIUC).
Simulations with NAMD, movie created with VMD.

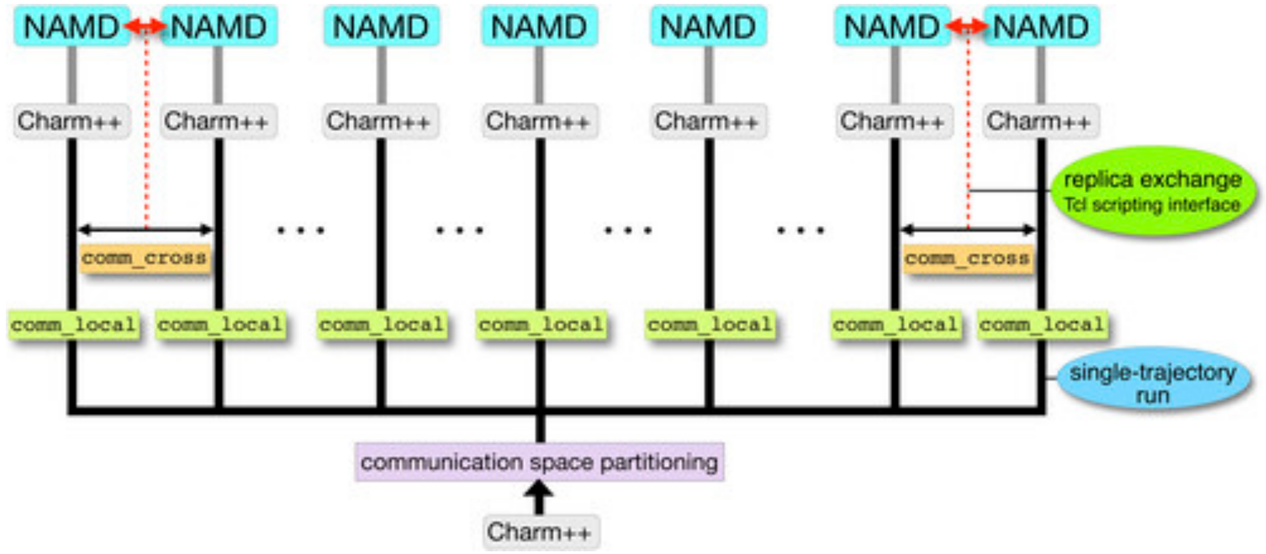
NIH National Institutes of Health

Early Science on Aurora: NAMD Free Energy Calculations

- Understand the function of large membrane transporters:
  - Calcium ATPase (SERCA) pump
  - P-glycoprotein (PGP) multidrug-resistance transporter
- Benefits to human health:
  - Both proteins use ATP hydrolysis as an energy source
  - Knowledge relating to multidrug resistance in cancer
- Free energy calculations involve thousands of weakly coupled "replicas" to cover thermodynamic reaction path
  - 20K-30K atom systems, one per GPU
  - Simulate using GPU-resident version of NAMD



SERCA PGP



Molecular Dynamics Simulation

Integrate Newton's equations of motion:

$$m_i \frac{d^2 \vec{r}_i}{dt^2} = \vec{F}_i = -\vec{\nabla} U(\vec{R})$$

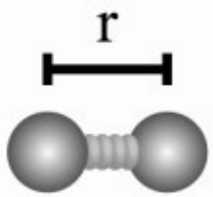
Integrate for millions of time steps

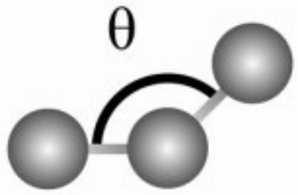
Bonded

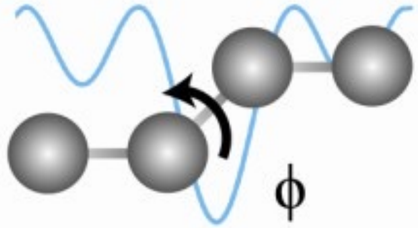
$$E_{total} = \sum_{bonds} K_r(r - r_{eq})^2 + \sum_{angles} K_\theta(\theta - \theta_{eq})^2 + \sum_{dihedrals} \frac{V_n}{2}[1 + \cos(n\phi - \gamma)]$$

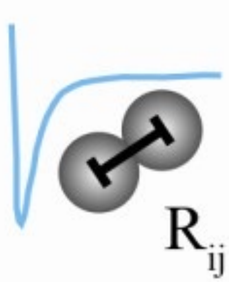
Non-bonded

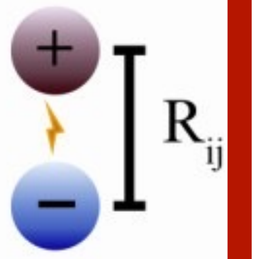
$$+ \sum_{i<j} \left[\frac{A_{ij}}{R_{ij}^{12}} - \frac{B_{ij}}{R_{ij}^{6}} + \frac{q_i q_j}{\epsilon R_{ij}} \right]$$







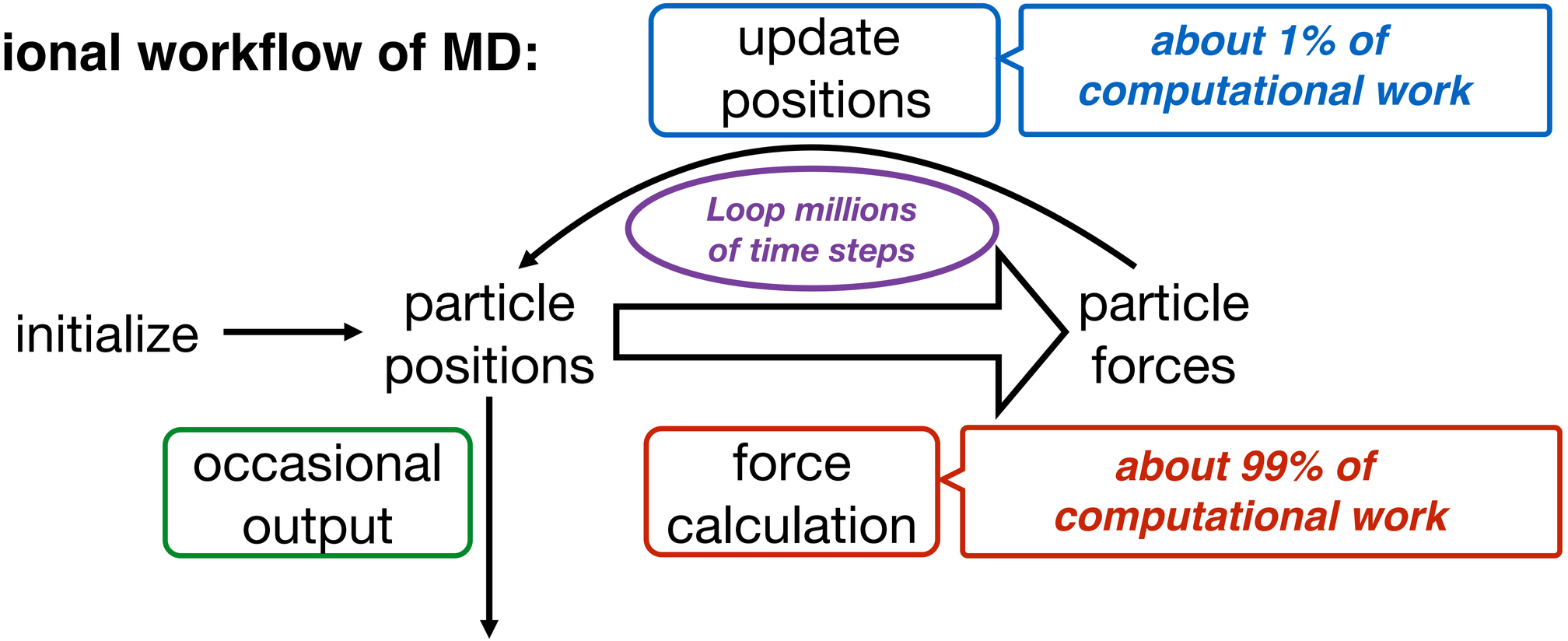




Most computationally intensive part

Parallelism for MD Simulation Limited to Each Time Step

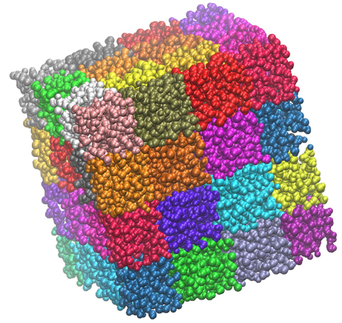
Computational workflow of MD:



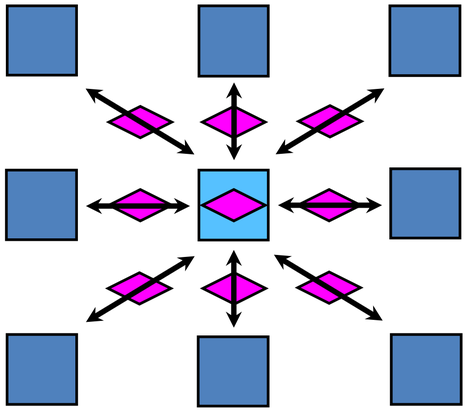
NAMD Parallelizes Domain and Interaction Space

- Decompose atoms into equal volume ***patches***
- Calculate pairwise forces between atoms, treat as interactions between neighboring patches
- Decompose patch-patch interaction ***compute objects***
- Moving atoms: update spatial decomposition by ***migrating atoms*** between adjacent patches
- Load balancing: update work decomposition by ***migrating compute objects*** to keep processors consistently occupied

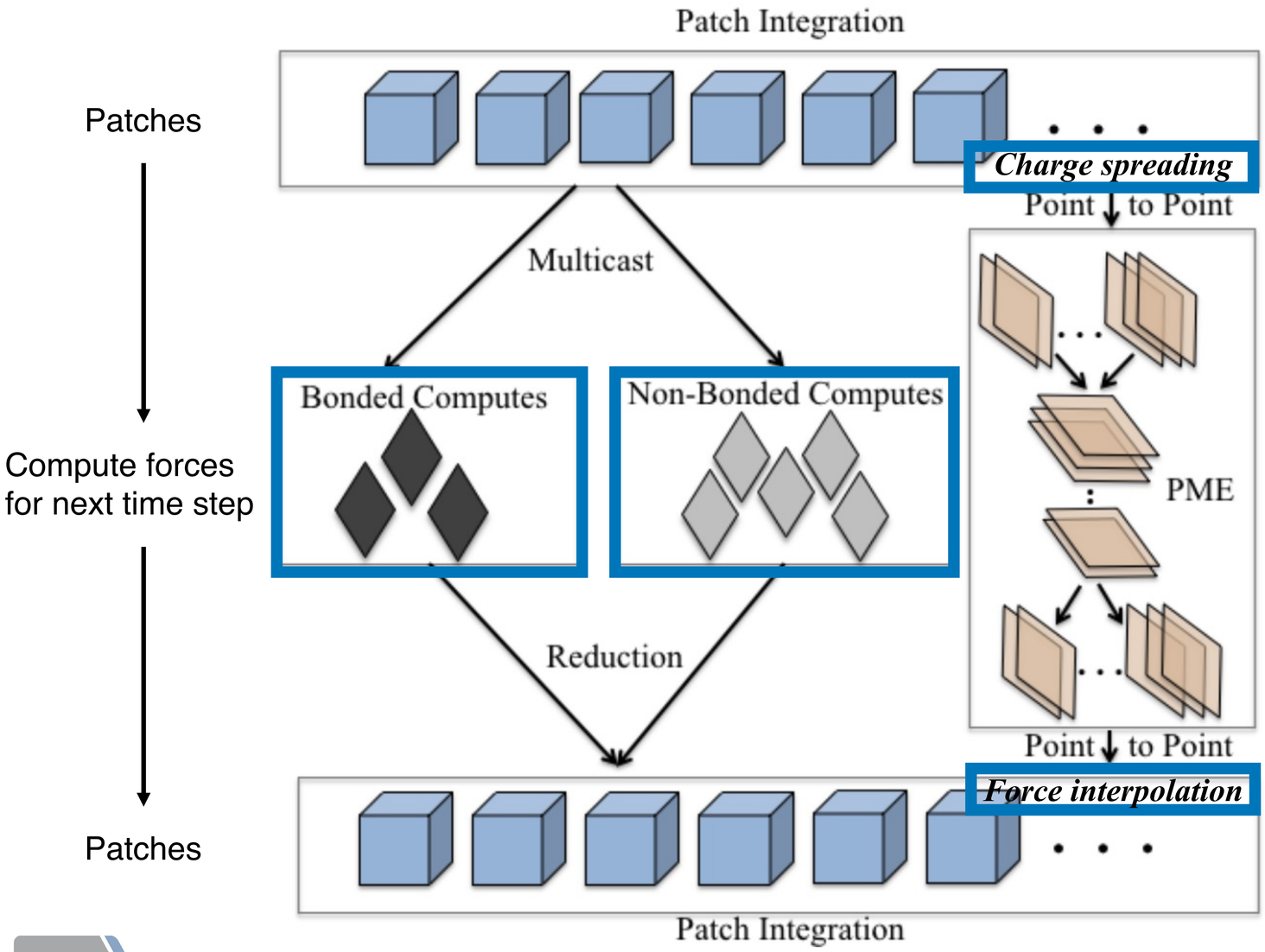
Spatial decomposition of atoms into patches



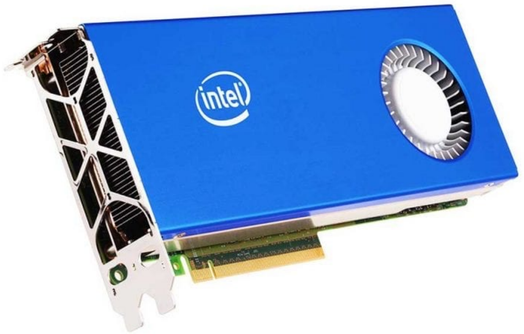
Work decomposition of patch-patch interactions into migratable compute objects



NAMD Parallel Workflow Incorporates GPUs



Offload force compute to GPU



Must aggregate positions

Multi-Node NAMD Uses GPU-Offload Scheme

Partition work between CPU and GPU

Showing approximate percentage of total work per step:

force calculation on GPUs

- **Short-range non-bonded forces (90%)**
- **Long-range PME electrostatics (5%)**
- **Bonded forces (2%)**
- **Corrections for excluded interactions (2%)**

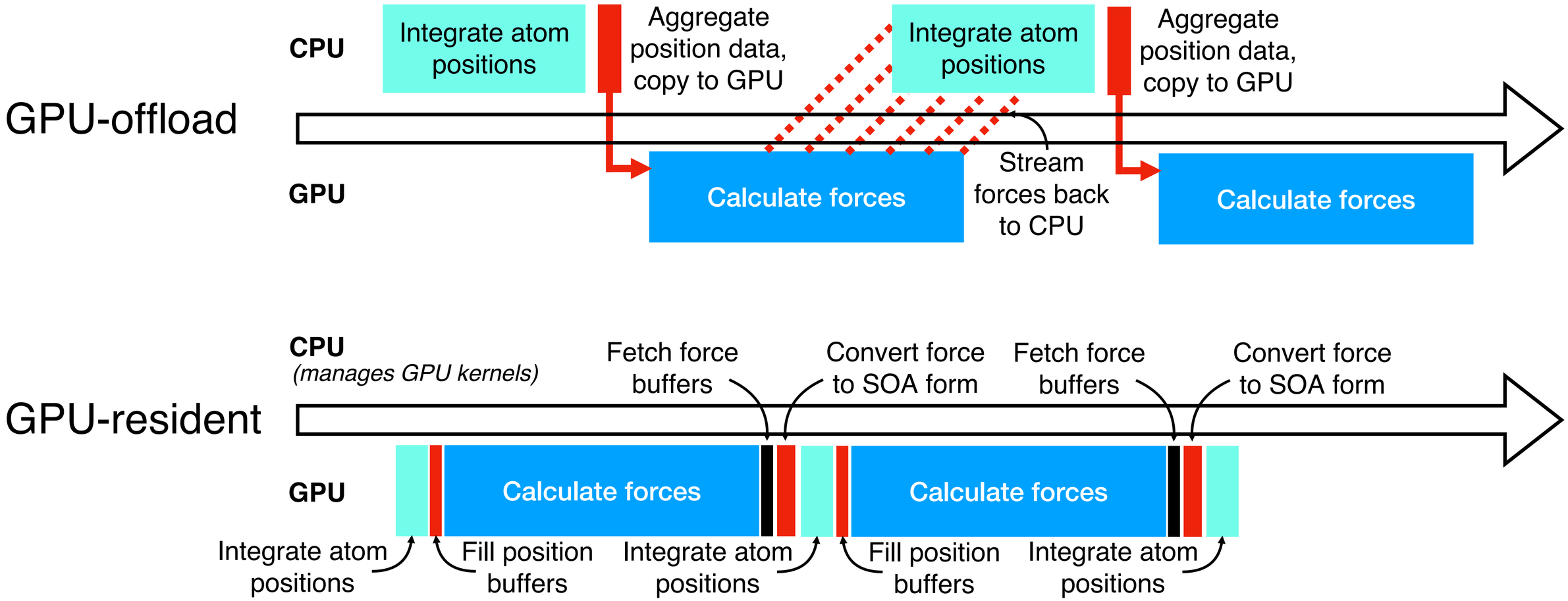
update coordinates on CPUs

- **Integrator, rigid bond constraints (1%)**

Enhanced sampling methods: additional forces, grid potentials, collective variables

New GPU-Resident Scheme Doubles GPU Performance

Move integrator to GPU and maintain data between time steps



Improve NAMD Code Longevity by Adopting SYCL

- Support upcoming exascale computers: ANL Aurora (Intel)
- SYCL provides advantages:
  - Modern C++ interface to GPU devices
  - Host-side code is much simpler than OpenCL
  - Same data structure definitions for both host and device
  - Single source and single compiler for host and device code
  - Vendor-neutral open standards language and library solution

How does SYCL differ from CUDA?

- Use of modern C++
  - Kernels defined as unnamed lambda expressions
  - Error-handling with try–catch block
- Design decisions in SYCL and OpenCL
  - SYCL work queue is analogous to CUDA stream, but defaults to out-of-order execution
  - Must specify accessor functions to enable SYCL kernels to access device buffers
  - Permit flexible vector width for performance portability across different hardware

Design Decisions for Porting NAMD

- **Extend NAMD without disrupting current GPU support**
  - Use preprocessor switches to isolate SYCL/DPC++ extensions from existing code
- **Leverage existing GPU kernels and data structures**
  - Translate CUDA kernels to SYCL and copy supporting data structures and kernel management infrastructure into SYCL/DPC++ versions
- **Add support incrementally, guided by Amdahl's Law, to accelerate most computationally expensive parts first**
  - Begin by porting short-range non-bonded force kernels
  - Continue with particle-mesh Ewald (PME) and bonded force kernels

Mirrors order of original CUDA development

NAMD Has a **LOT** of CUDA Code

- Eventually develop SYCL support for everything, including GPU-resident version
- Start porting from **stable code base** with GPU-offload (version 2.14)

Component	# of C/h files	# of cu files	# of kernels	src line count
Non-bonded force	6	2	20	5.8k
Bonded force	3	1	2	3.9k
PME - single node	6	1	5	4.1k
PME - scalable	6	1	3	3.3k
Utilities	8	1	1	1.7k
Total	29	6	31	18.8k

Overall Porting Strategy

- We employed a divide-and-conquer strategy, using preprocessor switches to decouple components in the CUDA code
  - Significantly reduces development and debugging complexity
- Separated components
  - Short-range non-bonded force & device utilities
  - Bonded force
  - PME (Particle-Mesh Ewald) — requires FFT
- Utilized supplemental libraries from oneAPI
  - oneDPL for C++17 parallel STL *reduce, shuffle, atomic_ref, sort* and *scan* replaces CUB library
  - oneMKL FFT replaces cuFFT library

Faster Porting with Code Conversion Tool

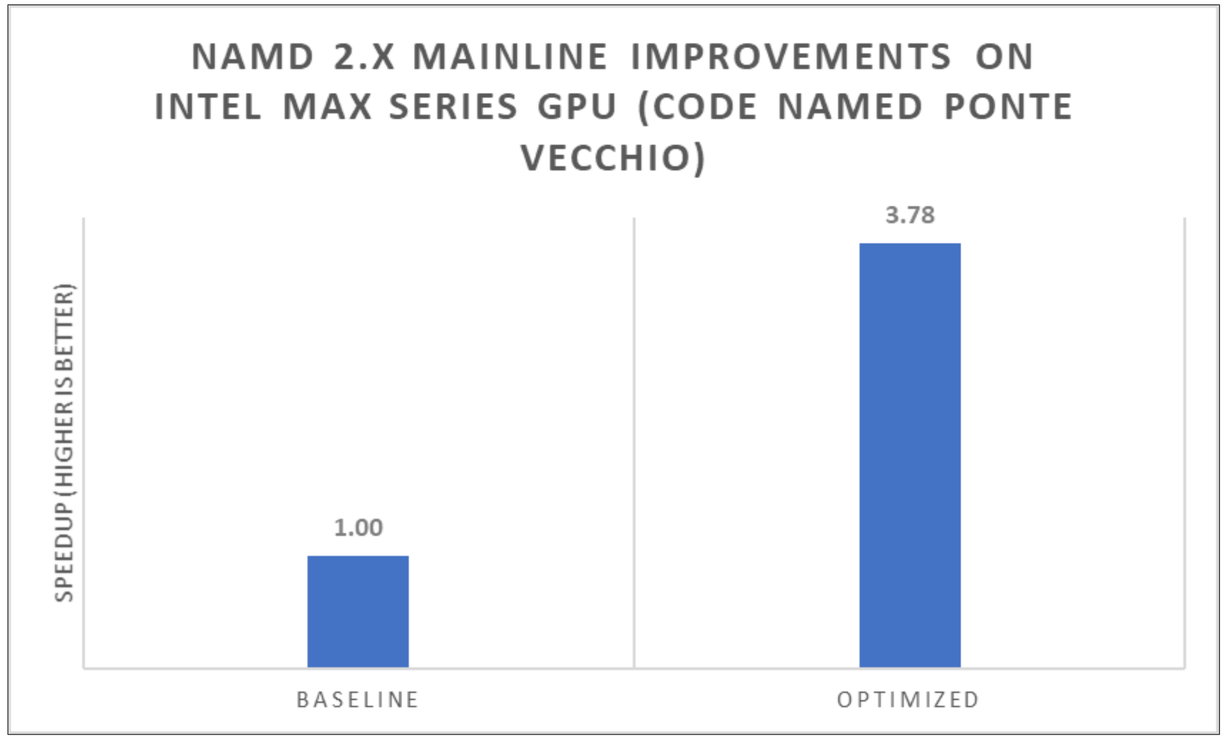
- Utilized SYCLomatic Migration Tool for faster development
- Started with converting the CUDA implementation
  - Saves > 80% of code porting effort
  - For example, threadIdx.x → ndItem.get_local_id(2)
- Provides a good guide to practice SYCL syntax

Validating SYCL Port of GPU-Offload Kernels

- Maximum relative error in total energy for 500-step constant energy simulation
- CPU-only run shows lower error due to double precision compared to mixed-precision GPU runs

architecture	ApoA1 (92K atoms)	STMV (1M atoms)
CPU-only (2nd run)	4.35841E-08	3.95857E-07
A5000 (CUDA)	3.17476E-06	4.97212E-06
CPU (DPC++)	2.89769E-06	3.29257E-06
Gen9 (DPC++)	2.82174E-06	3.84310E-06
ATS/Xe-HP (DPC++)	2.09291E-06	3.39862E-06

NAMD 2.x Performance Improvements on Intel Max Series GPU

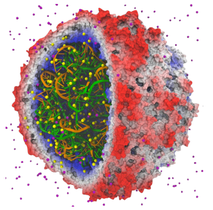


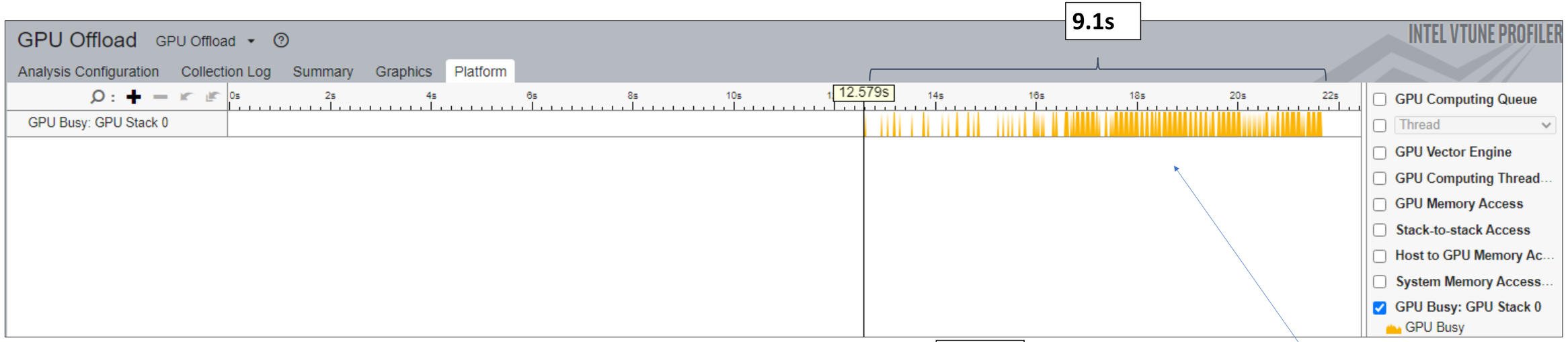
- Non-bonded force + PME offload
- Based on improvements from
  - Firmware, driver, oneAPI, SYCL optimizations
- NAMD used heavily for hardening of Intel oneAPI and other software components
  - oneAPI, Intel VTune, etc.

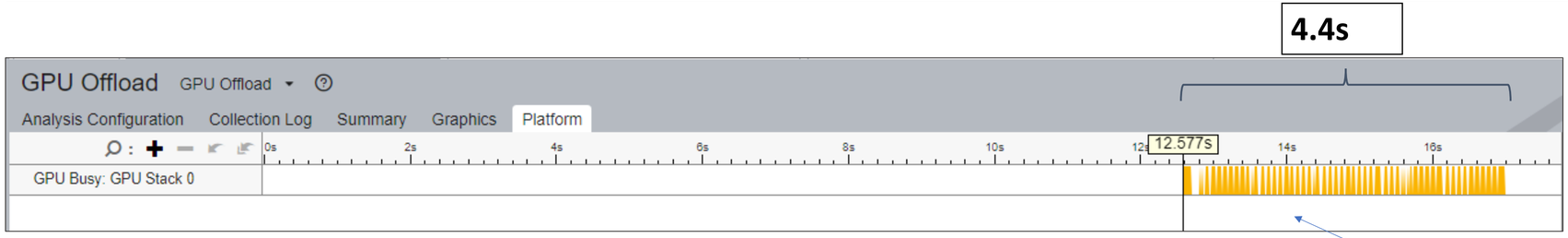
Configuration: 2-socket 3rd generation Intel Xeon® Scalable processor (code named Icelake) 8360Y, Intel Max Series GPU

Intel VTune™ Profile of NAMD on Intel MAX Series GPU

Simulating STMV (1M-atom benchmark system)







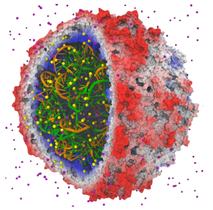
Before Optimization

After Optimization

Configuration: 1-socket 3rd generation Intel Xeon® Scalable processor (code named Icelake) 8360Y/32 NAMD host threads; GPU Stack 0 of Intel Max Series GPU; Non-Bonded Force offload; 100 steps

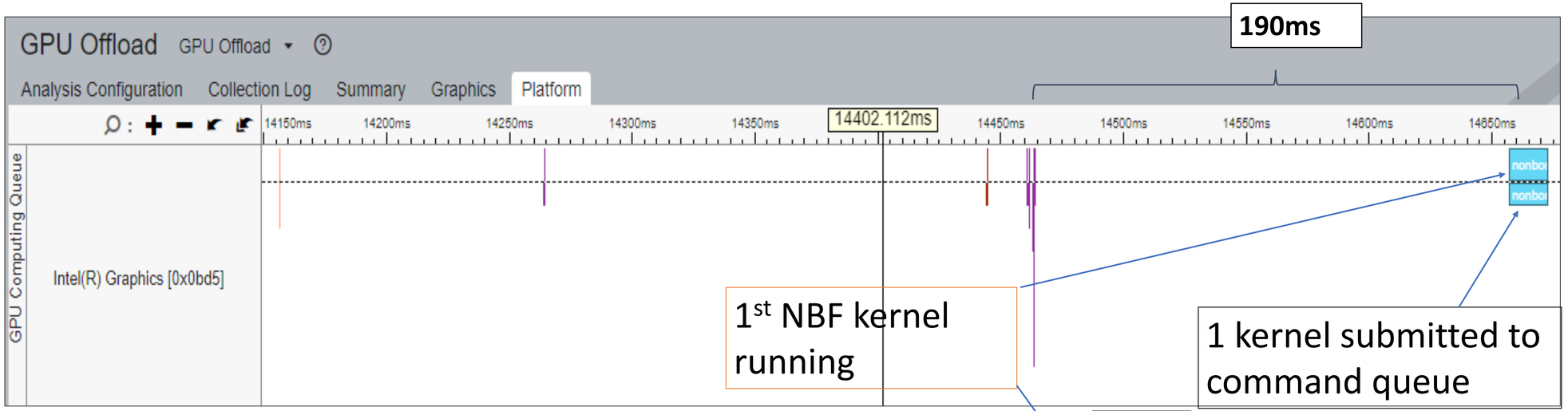
2x reduction in NBF GPU execution time

Intel VTune™ Profile of NAMD on Intel MAX Series GPU

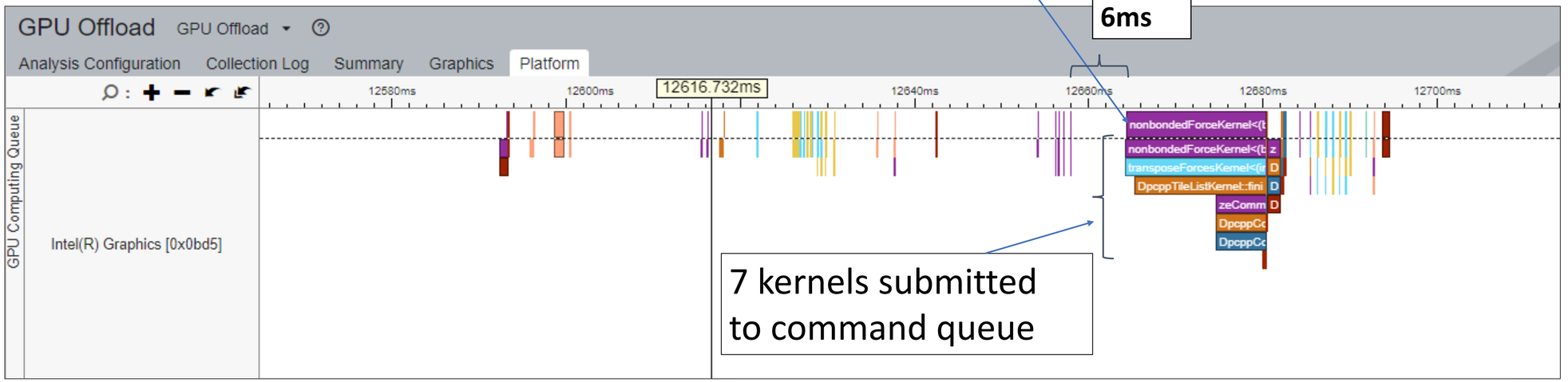


Simulating STMV (1M-atom benchmark system)

Before Optimization



After Optimization



Configuration: 1-socket 3rd generation Intel Xeon® Scalable processor (code named Icelake) 8360Y/32 NAMD host threads; GPU Stack 0 of Intel Max Series GPU; Non-Bonded Force offload; 100 steps

Improved SYCL kernel submission efficiency

Conclusions and Future Work

- Experience shows that Intel oneAPI tools are usable and capable
- Demonstrates SYCL porting for large complicated codebase
- Continue to improve NAMD performance
  - Improve SYCL kernel performance and further reduce synchronization latencies
  - Work on multi-node scaling as nodes of Aurora become available
- Port GPU-resident version of NAMD
  - Makes use of GPU-offload force compute kernels

Acknowledgments

- SYCL porting credits: Tareq Malas (formerly Intel), Jaemin Choi (formerly UIUC), Mike Brown (Intel)
- ANL Aurora Early Science Program of the Argonne Leadership Computing Facility
- Intel funding through UIUC oneAPI Center of Excellence
- NIH Grant P41-GM104601

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

NIH National Institutes of Health