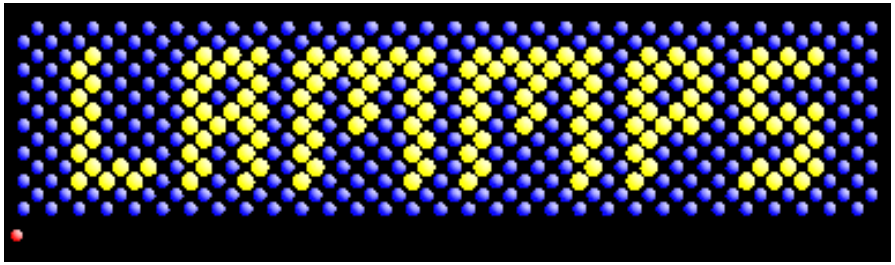
Exceptional service in the national interest

Sandia National Laboratories



LAMMPS and SPARTA: Performance Portability of Particle Methods Through Kokkos

Stan Moore

Sandia National Laboratories

CExA Fourth International Kokkos Tea-Time

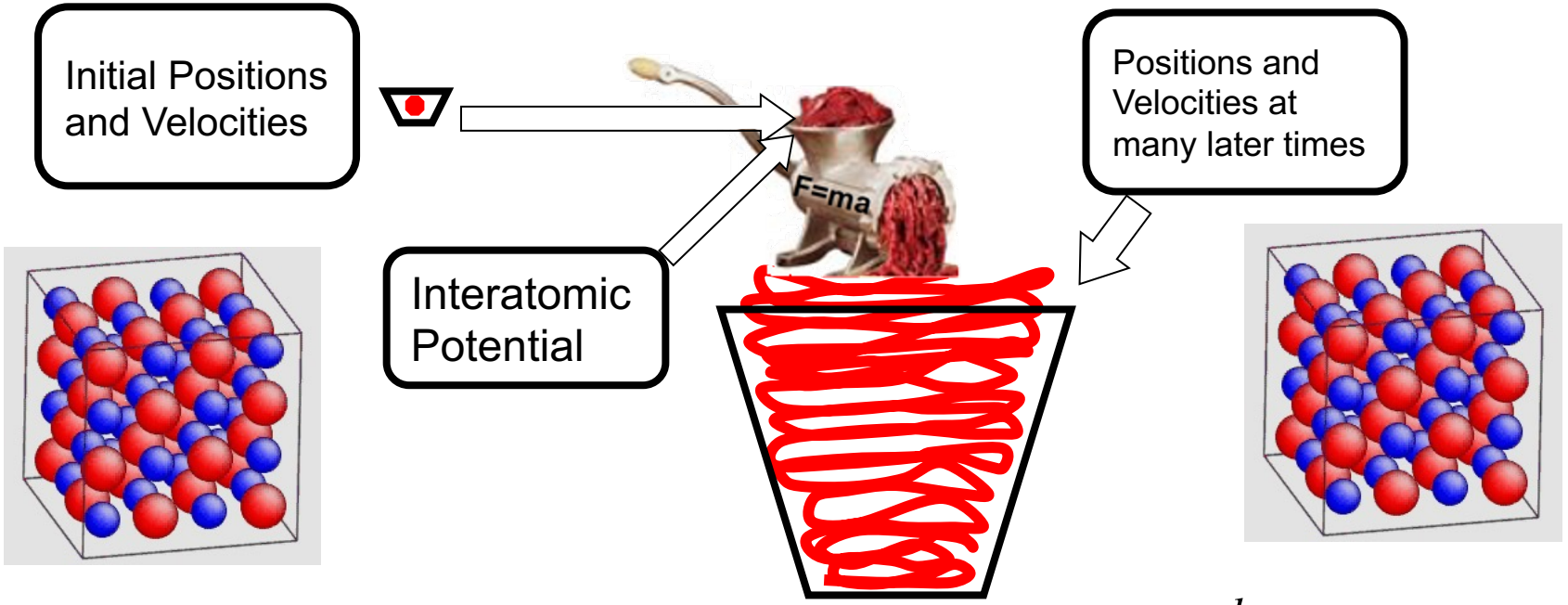
October 16, 2024

U.S. DEPARTMENT OF ENERGY

NNSA National Nuclear Security Administration

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525. SAND2024-14248C

Molecular Dynamics: What is it?



Mathematical Formulation

- Classical Mechanics
- Atoms are Point Masses: r_1, r_2, ... r_N
- Positions, Velocities, Forces: r_i, v_i, F_i
- Potential Energy Function = $V(r^N)$
- $6N$ coupled ODEs

Newton's Equations:

$$\frac{d\mathbf{r}_i}{dt} = \mathbf{v}_i$$

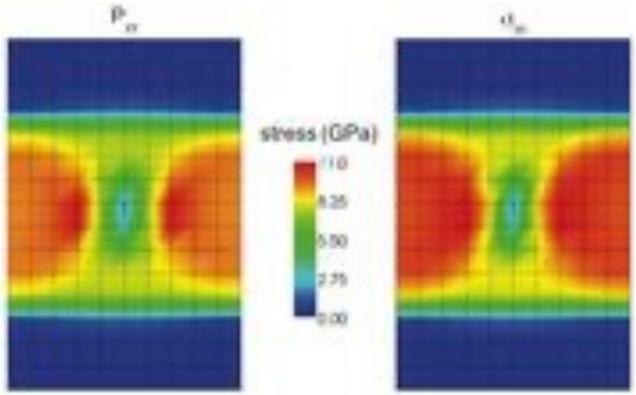
$$\frac{d\mathbf{v}_i}{dt} = \frac{\mathbf{F}_i}{m_i}$$

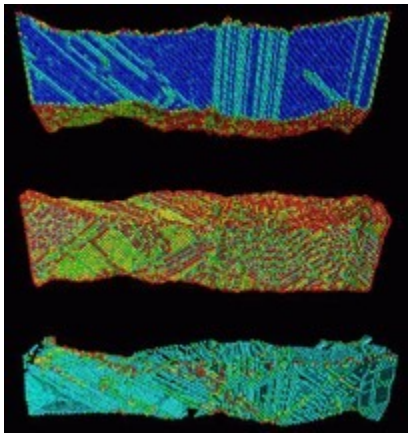
$$\mathbf{F}_i = -\frac{d}{d\mathbf{r}_i} V\left(\mathbf{r}^N\right)$$

MD Versatility

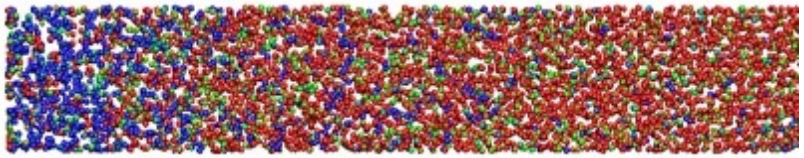
Sandia National Laboratories

Coupling to Solid Mechanics



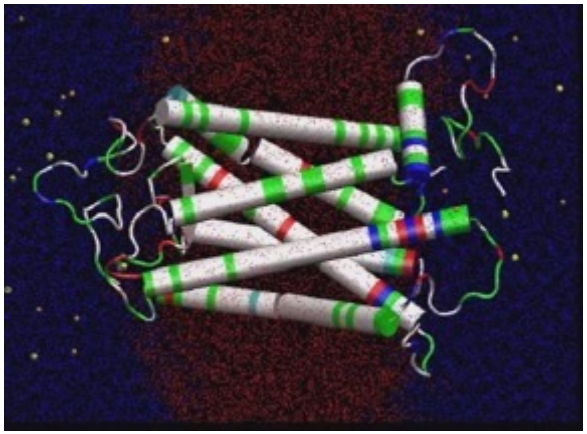


Materials Science: metals, polymers, etc.

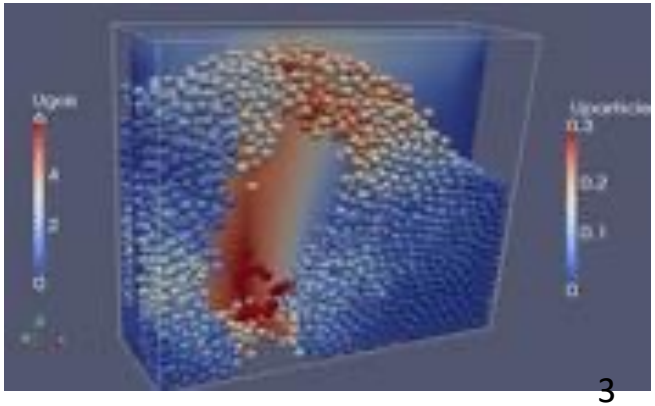


Chemistry

Biophysics



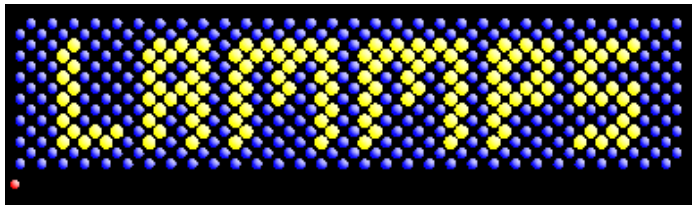
Granular Flow

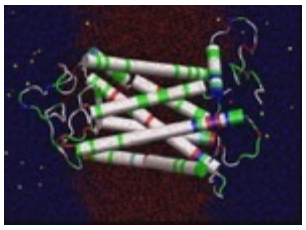


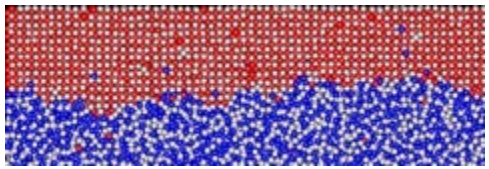
LAMMPS Code

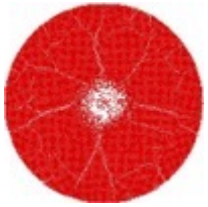
(**L**arge-scale **A**tomic/**M**olecular **M**assively **P**arallel **S**imulator)

- http://lammps.org
  - Open source, C++ code
  - Bio, materials, mesoscale









  - **Molecular Dynamics** (MD) focus, but more generally a **particle simulator** at varying length and time scales
    - Electrons → atomistic → coarse-grained → continuum
  - Spatial-decomposition of simulation domain for parallelism
  - Energy minimization, dynamics, non-equilibrium MD
  - Can be coupled to other scales: QM, kMC, FE, CFD, …

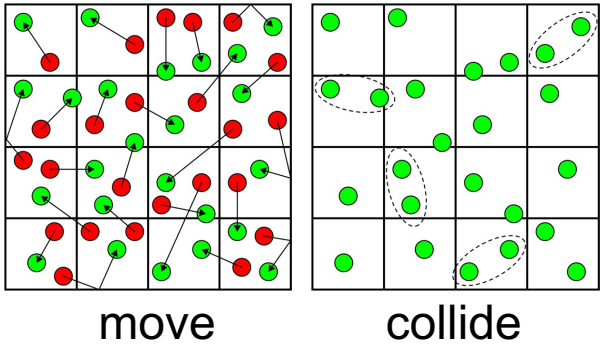
A. P. Thompson, H. M. Aktulga, R. Berger, D. S. Bolintineanu, W. M. Brown, P. S. Crozier, P. J. in 't Veld, A. Kohlmeyer, S. G. Moore, T. D. Nguyen, R. Shan, M. J. Stevens, J. Tranchida, C. Trott, S. J. Plimpton, Comp Phys Comm, 271 (2022) 10817

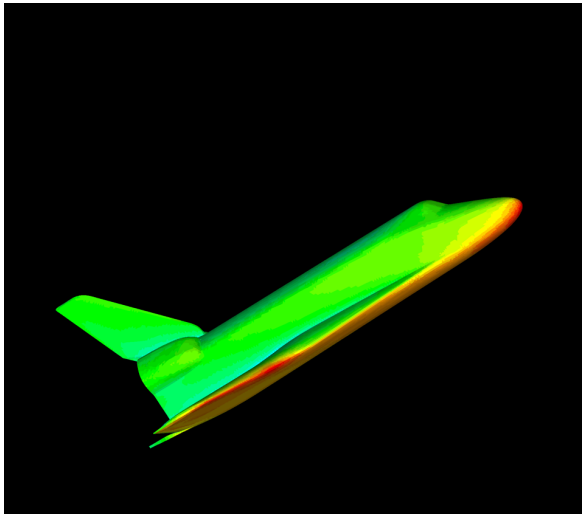
Direct Simulation Monte Carlo (DSMC)

Sandia National Laboratories

DSMC models fluids using molecules

- Solves the Boltzmann Equation stochastically
- Decouples particle advection and particle collisions
- Only viable approach for rarefied gases (mean-free path is large and continuum assumptions break down)
- Inherently includes physics usually not in CFD, valid for highly non-equilibrium flows

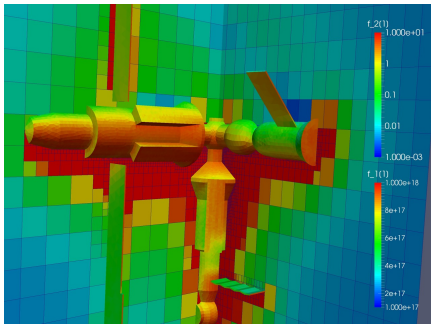




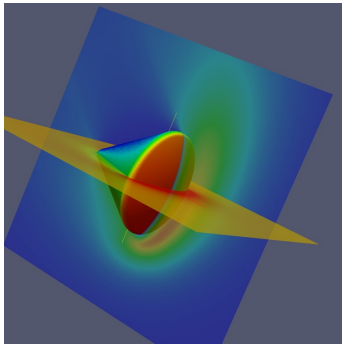
SPARTA Code

(**S**tochastic **PA**rallel **R**arefied-gas **T**ime-accurate **A**nalyzer)

- Direct Simulation Monte Carlo (DSMC) code (**rarefied gas flows**)
- Core developers are Stan Moore and Michael Gallis (SNL), and Steve Plimpton (retired SNL)
- Open-source, http://sparta.github.io
- Collaborators: ORNL, LANL, ANL, LBNL, NASA, ESA, academia

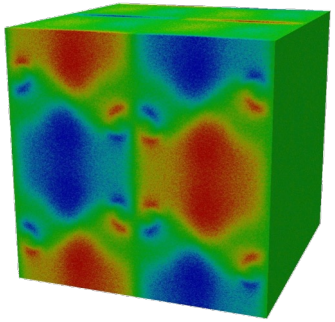


Spacecraft

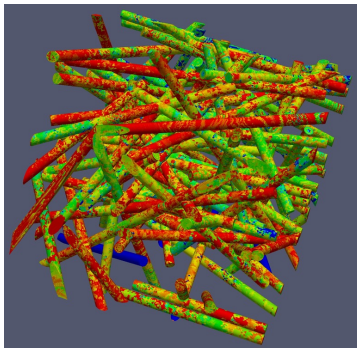


Re-entry

Instabilities



Turbulence

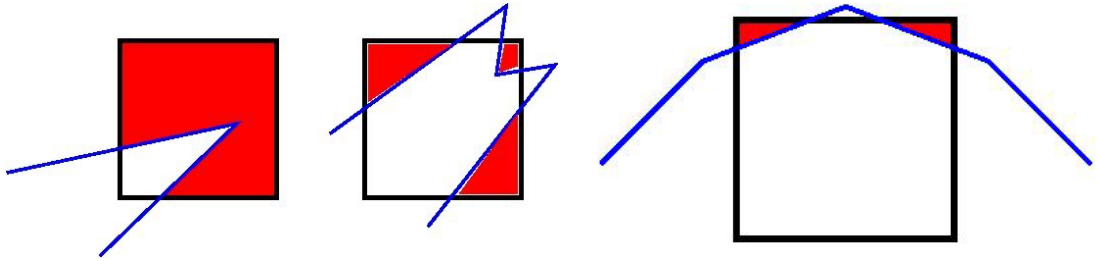


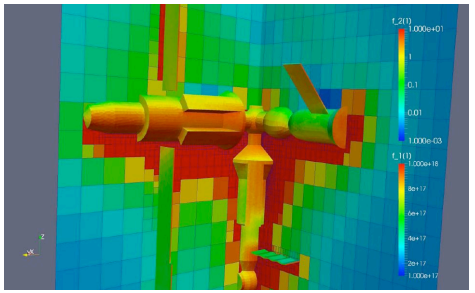
Porous Media

S. J. Plimpton, S. G. Moore, A. Borner, A. K. Stagg, T. P. Koehler, J. R. Torczynski, M. A. Gallis, Physics of Fluids, **31**, 086101 (2019)

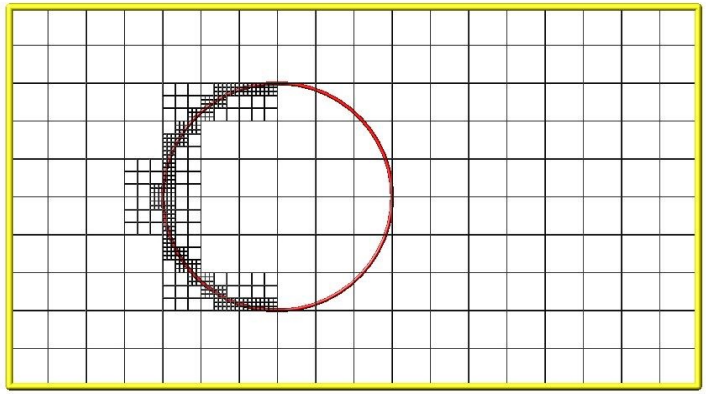
SPARTA Features

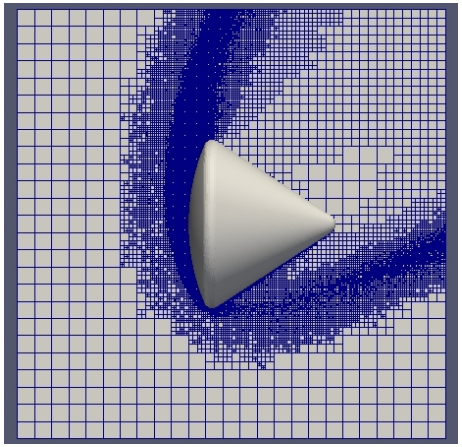
- Structured grids with complex surfaces via cut and split cells





- Hierarchal grids with adaptive mesh refinement

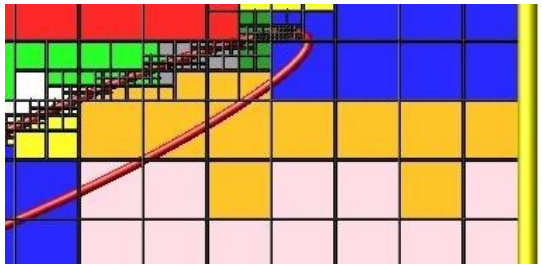


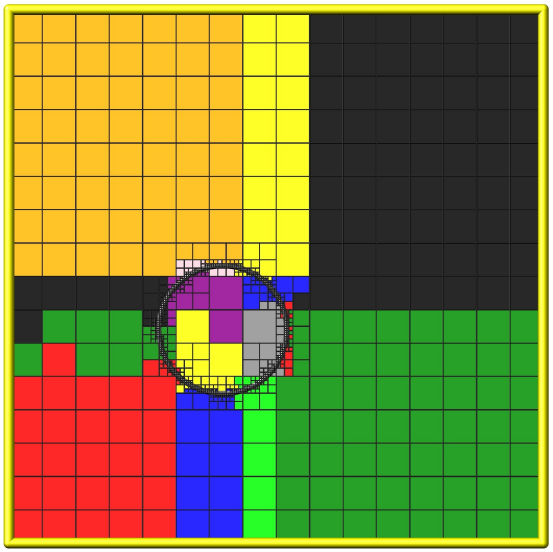


- MPI parallelism using highly scalable domain decomposition

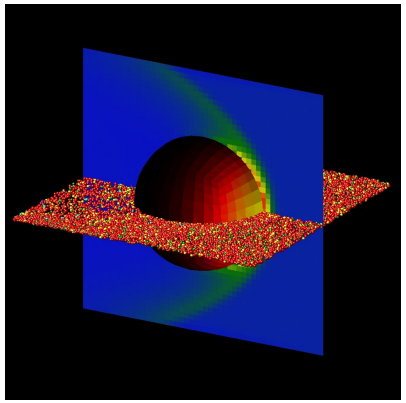
SPARTA Features (cont.)

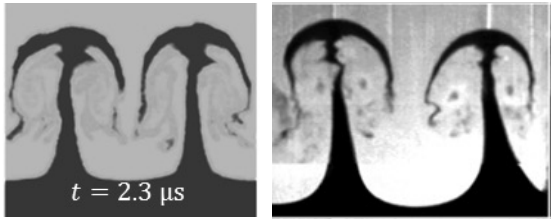
- Load balancing (static and dynamic)



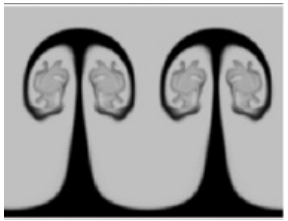


- In-Situ Visualization

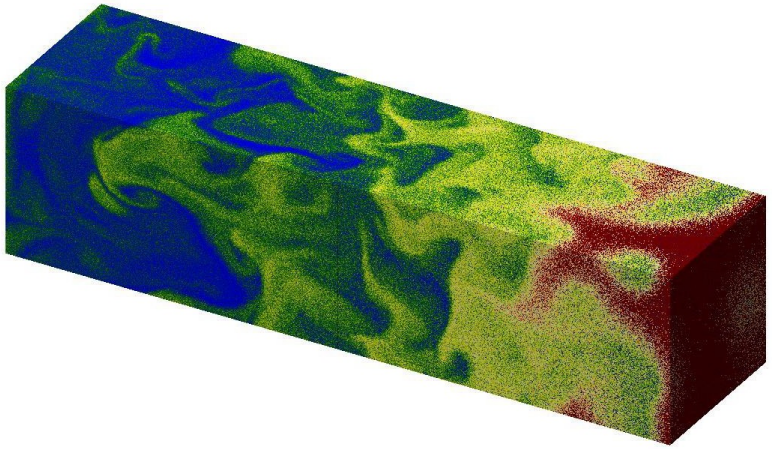




Experiment DSMC



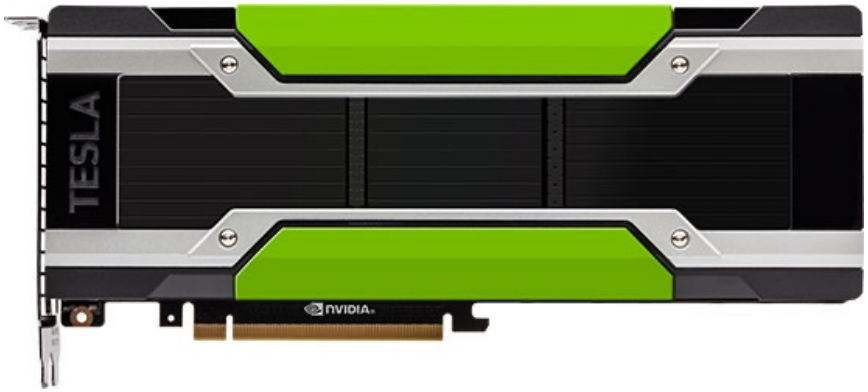
Navier-Stokes



- And more

HPC Hardware Trends

- Currently 9 out of the top 10 supercomputers use GPUs (Graphics Processing Units), according to the June 2024 Top500 List (https://www.top500.org)
- All 3 US exascale machines will have GPUs: OLCF Frontier (AMD), ALCF Aurora (Intel), NNSA El Capitan (AMD)
- Special code (beyond regular C++ and MPI) is required to run well on GPUs and many-core CPUs (e.g. CUDA, OpenMP)

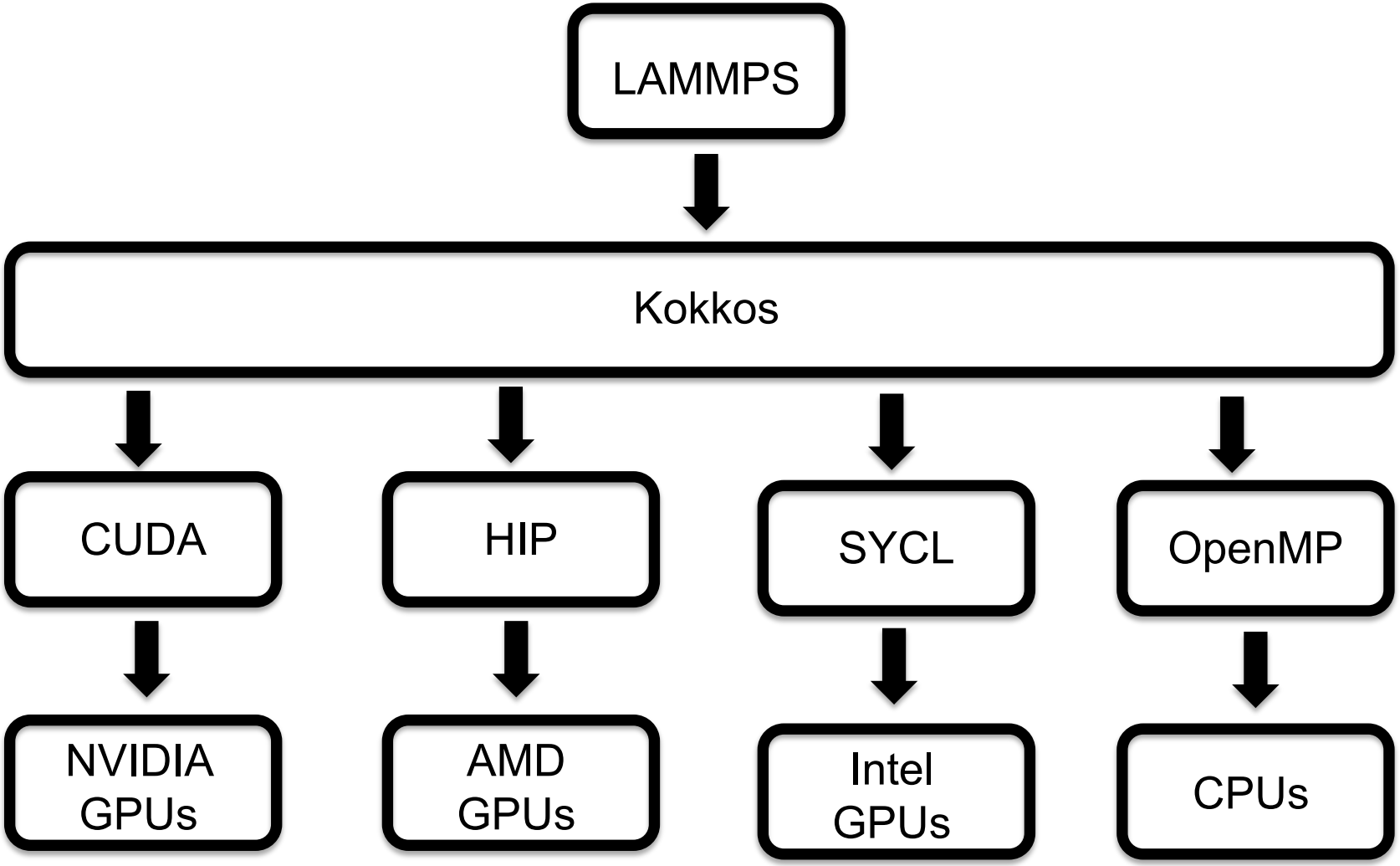


http://www.nvidia.com/object/tesla-p100.html

Sandia National Laboratories

- Abstraction layer between programmer and next-generation platforms
- Allows the same C++ code to run on multiple hardware (GPU, CPU)
- Kokkos consists of two main parts:
  1. Parallel dispatch—threaded kernels are launched and mapped onto backend languages such as CUDA or OpenMP
  2. Kokkos views—polymorphic memory layouts that can be optimized for a specific hardware
- Used on top of existing MPI parallelization (MPI + X)
- See https://kokkos.github.io/kokkos-core-wiki for more info

Kokkos LAMMPS



History

Sandia National Laboratories

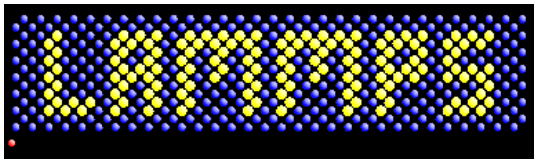
LAMMPS

- Fortran version created in **mid-1990s**, C++ MPI version released in **2004**
- Initial Kokkos implementation released in **2014**, supported CUDA and OpenMP backends
- Support for Kokkos HIP backend added in **2020**
- Support for SYCL and OpenMPTarget backends added in **2021**

SPARTA

- C++ MPI version released open source in **2014**
- Initial Kokkos implementation released in **2017** (CUDA and OpenMP)
- HIP, SYCL, and OpenMPTarget support added in **2021-2022**

Accelerator Packages in LAMMPS



- **Vanilla C++ version**

Accelerator packages:

- **OpenMP Package:** native OpenMP threading
- **INTEL Package**: native OpenMP threading, enhanced SIMD vectorization, uses hardware intrinsics, fast on CPUs but very complex code
- **GPU Package**: native CUDA and OpenCL support, only runs a few kernels (e.g. *pair* force calculation) on GPU, needs multiple MPI ranks per GPU to parallelize CPU calculations
- **KOKKOS Package**: implements Kokkos library abstractions, tries to run everything on device, supports CUDA (NVIDIA GPUs), HIP (AMD GPUs), SYCL (INTEL GPUs), and OpenMP (CPU) threading backends

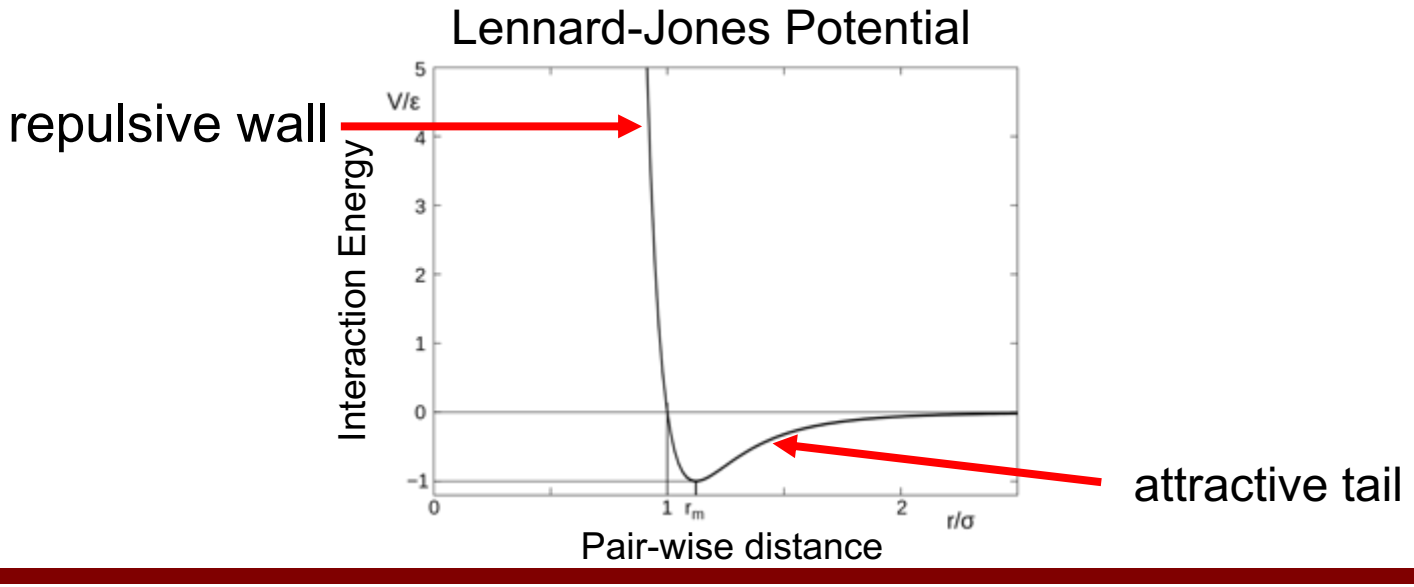
Legacy Code

Sandia National Laboratories

- Kokkos views are **aliased to legacy** C++ data structures in LAMMPS and SPARTA
- Allows incremental porting (good) but requires **LayoutRight and FP64 double precision** for view aliased to legacy
- Non-ported code runs non-threaded (MPI-only) on host CPU
- Non-ported code requires **data transfer between GPU and CPU**. Automated framework for most styles in LAMMPS and SPARTA, could be inefficient though
- Use *Kokkos::DualView* "sync" and "modify" for data transfer of aliased views: convenient and usually faster than managed memory, but prone to **hard-to-find bugs**

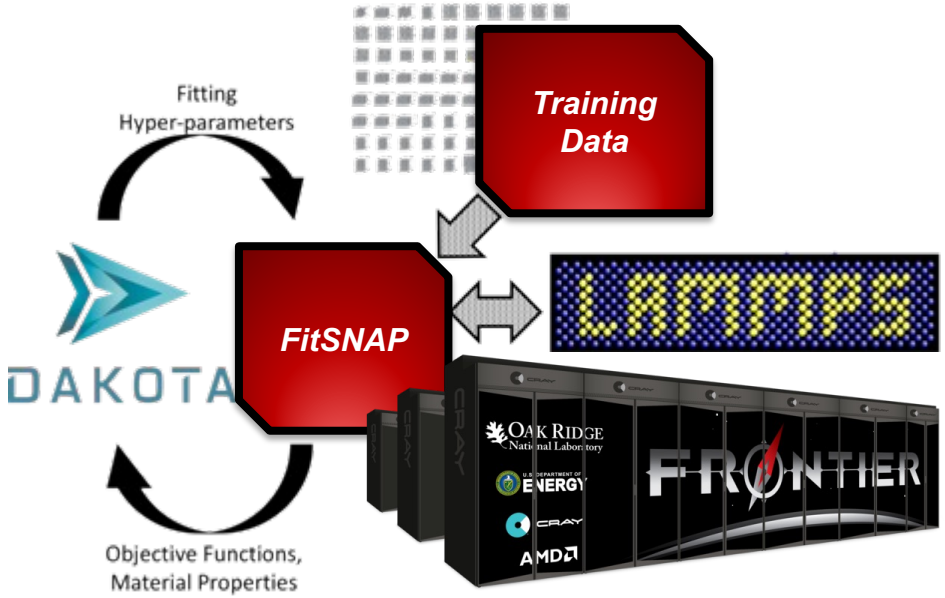
Interatomic Potentials

- Quantum chemistry (e.g. DFT): solves Schrödinger equation (electron interactions) to get forces on atoms. Accurate but very computationally expensive and only feasible for small systems: ~1000 atoms
- Molecular dynamics: uses empirical force fields, sometimes fit to quantum data. Not as accurate but **much** faster
- MD typically only considers pair-wise or three-body interactions, scales as *O*(N) (billion atom simulations are considered huge)



SNAP Machine Learning Potential

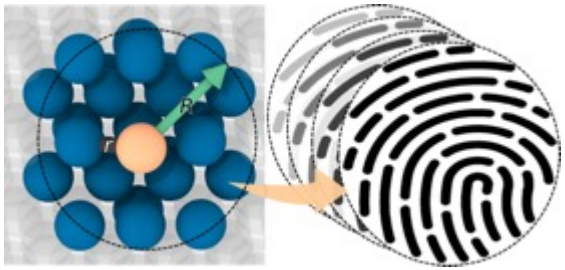
- Empirical models in LAMMPS are being replaced by machine learning (ML)
- ML interatomic potentials (IAPs) have three critical parts:
  1. Descriptors of the local environment
  2. Energy and force functions expressed in the descriptors
  3. Training on large amount of “ground truth” energies and forces from quantum chemistry calculations (e.g. DFT)



SNAP Bispectrum Components

- Neighbors of each atom are mapped onto unit sphere in 4D

$$3D\ Ball\colon (r,\theta,\phi), r < R_{cut} \implies 4D\ Sphere\colon (\theta_0,\theta,\phi), \theta_0 = \frac{r}{R_{cut}}\pi$$



- Expand density around each atom in a basis of 4D hyperspherical ***harmonics***,

$$\rho_i(\mathbf{r}) = \delta(\mathbf{0}) + \sum_{r_{i'}<R_{cut}} f_c(r_{i'}) w_{i'} \delta(\mathbf{r}_{i'})$$

- Deeply nested loops
- Loop structure not regular
- Loop sizes <= 14

- Bispectrum components of the 4D hyperspherical harmonic expansion are used as the geometric descriptors of the local environment
  - Preserves universal physical symmetries
  - Rotation, translation, permutation
  - Size-consistent (extensible)

$$u^j_{m,m'} = U^j_{m,m'}(0,0,0) + \sum_{r_{ii'}<R_{cut}} f_c(r_{ii'}) w_i U^j_{m,m'}(\theta_0,\theta,\phi)$$

$$B_{j_1,j_2,j} = \sum_{m_1,m_1'=-j_1}^{j_1} \sum_{m_2,m_2'=-j_2}^{j_2} \sum_{m,m'=-j}^{j} (u^j_{m,m'})^* H^{jmm'}_{\substack{j_1 m_1 m_1' \\ j_2 m_2 m_2'}} u^{j_1}_{m_1,m_1'} u^{j_2}_{m_2,m_2'}$$

Team for LAMMPS/SNAP GPU Optimizations

Sandia National Laboratories

- **Stan Moore** (SNL): LAMMPS Kokkos lead developer, integrated improvements into public LAMMPS, benchmarked LAMMPS on pre-exascale testbeds
- **Aidan Thompson** (SNL): created TestSNAP proxy app with built-in correctness check, algorithm redesign
- **Nick Lubbers** (LANL): algorithm redesign
- **Rahul Gayatri** (NERSC) and **Neil Mehta** (NERSC): performance improvements, support for TestSNAP and LAMMPS on pre-exascale testbeds, developing Kokkos OpenMPTarget backend
- **Evan Weinberg** (NVIDIA): Major performance improvements on GPUs
- **Nick Curtis** (AMD): Profiling SNAP on MI250X, Kokkos HIP backend improvements, investigating SNAP performance
- **Chris Knight** (ALCF) and **Yasi Ghadar** (ALCF): support for TestSNAP and LAMMPS on pre-Aurora testbeds
- **Daniel Arndt** (ORNL): developing Kokkos SYCL backend, helped tune TestSNAP performance on Arcticus
- **Mike Brown** (Intel): performance improvements on Aurora

SNAP Improvements

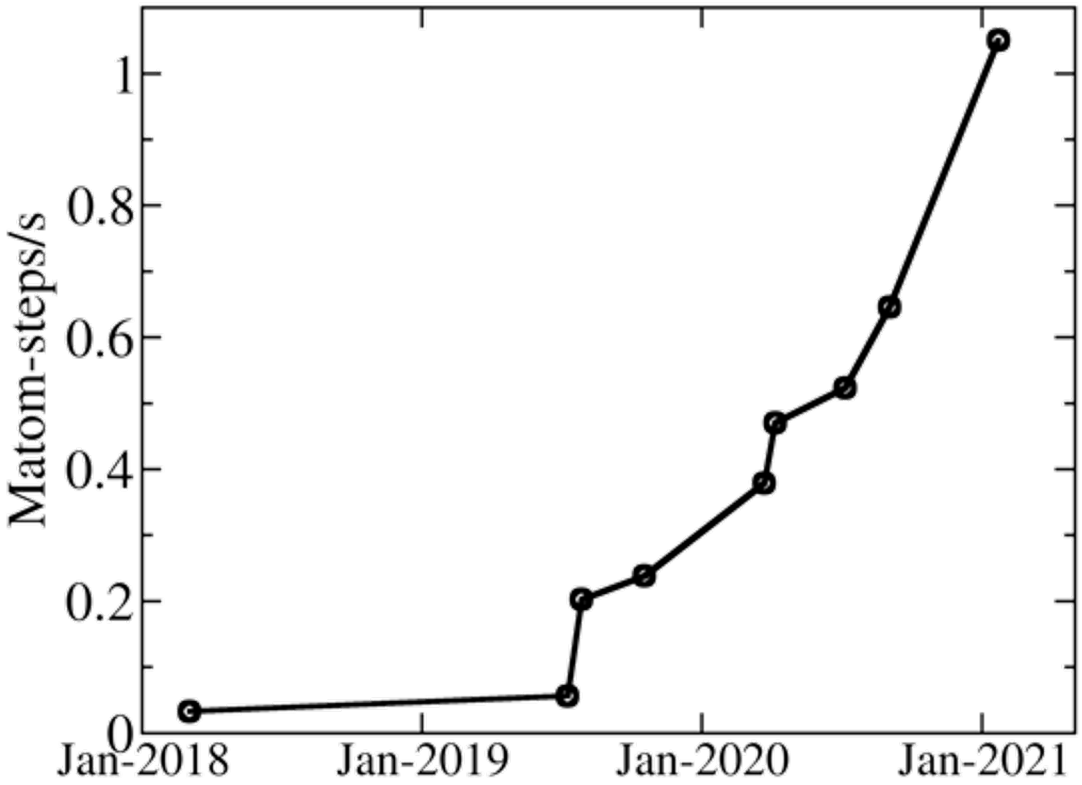
Sandia National Laboratories

- **Adjoint refactor**: algorithmic redesign that reduced the computational complexity and memory footprint by large factor
- **Flattened jagged multi-dimensional arrays:** reduced memory use
- **Major kernel refactor:** Broke one large kernel into many smaller kernels, reordered loop structure
- **Changed the memory data layout** of an array between kernels via transpose operations
- **Refactored loop indices and data structures** to use complex numbers and multi-dimensional arrays instead of arrays of structs
- Refactored some kernels to **avoid thread atomics** and use of **global memory**
- Judiciously used **Kokkos hierarchical parallelism** and **GPU shared memory**
- **Fused** a few selected **kernels**, which helped eliminate intermediate data structures and reduced memory use
- Added an AoSoA **memory data layout** inspired by Cabana code, which enforced perfect coalescing and load balancing in one of the kernels
- **Symmetrized data layouts** of certain matrices, which reduced memory overhead and use of thread atomics on GPUs (also improved CPU performance)
- Large refactor of Wigner matrices + derivatives to **use AoSoA data layout**
- **Pack several 32-bit integers** for Clebsch-Gordon coefficient lookup tables **into 128-bit int4 structs** and use 128-bit load/store to **reduce memory transactions**

SNAP Performance on V100

- Over 30x speedup since 2018!

better



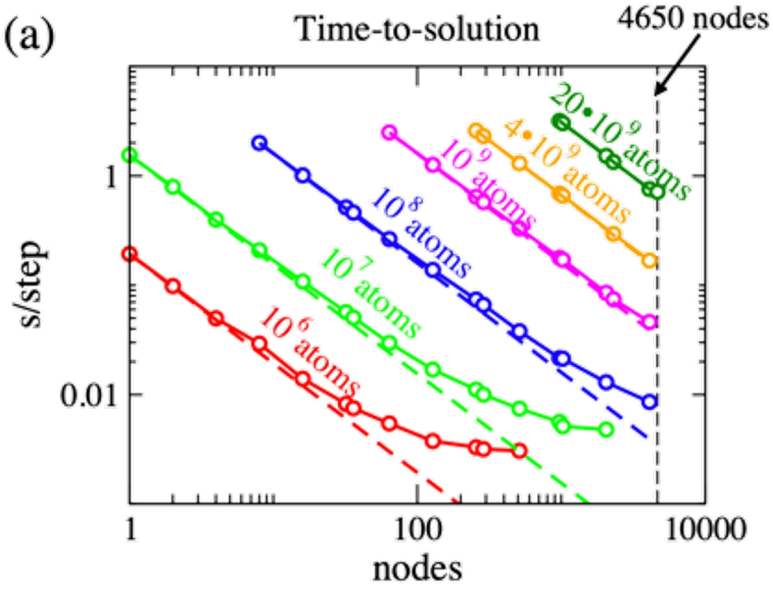
Lessons Learned

Sandia National Laboratories

- Kokkos overheads are generally small. Large performance gains by optimizing both **algorithm** and **GPU implementation**
- Newly ported kernels often require **optimization and tuning** by hand, naïve port from CPU code may not be sufficient
- If something is missing or not performant in Kokkos, it can usually be added or fixed!
- Different strategy for heavy vs lightweight kernels
- For lightweight kernels:
  - Reduce launch latency (fuse kernels)
  - Reduce host ←→ device data transfer (port all kernels)
  - Watch for unnecessary view initialization (on by default in Kokkos)
  - Use subview array instead of multiple scalar views

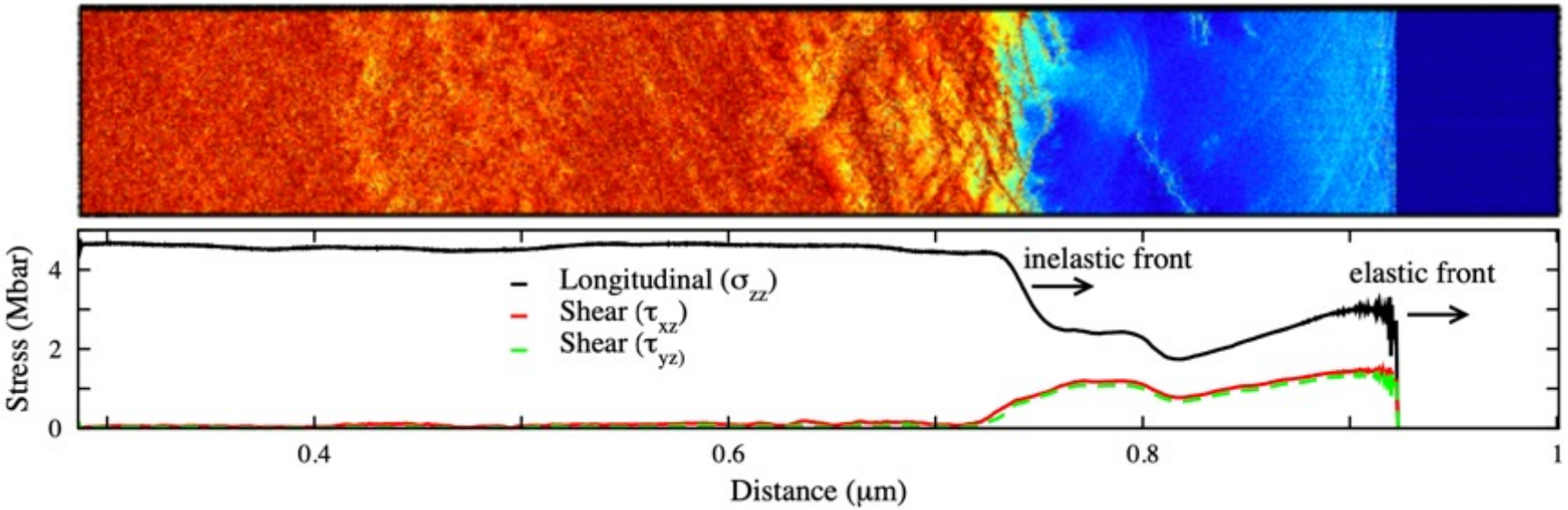
2021 ACM Gordon-Bell Award Finalist

- **“Billion atom molecular dynamics simulations of carbon at extreme conditions and experimental time and length scales”**
- SNAP model of carbon
- Team members from Sandia, U of S. Florida, NVIDIA, NERSC, and KTH
- Ran SNAP carbon model on full OLCF Summit (27,900 GPUs)
- Achieved **50.0 PFLOPs: 24.9% of Summit theoretical peak**, 33.6% of measured LINPACK benchmark
- SNAP MD simulation rate **22.9x higher** than DeepMD (2020 Gordon-Bell award for quantum-accurate MD)



K. N. Cong, J. T. Willman, S. G. Moore, A. B. Belonoshko, R. Gayatri, E. Weinberg, M. A. Wood, A. P. Thompson, I. I. Oleynik, In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC '21), Article 4, 1–12, 2022.

Shock Wave in Diamond Crystal



SNAP machine learning potential for carbon fit to quantum chemistry (DFT) calculations

Billion+ carbon atom simulation of split elastic-inelastic shock wave propagating in single crystal diamond (dark blue)

The elastic precursor (light blue) is followed by an inelastic wave (red), which exhibits an unexpected stress relaxation mechanism

Towards Exascale

- Ran older versions of SNAP on OLCF Crusher MI250X: NVIDIA improvements also helped MI250X (V100 was an excellent proxy for MI250X)
- ~25x performance improvement comparing original vs latest code for both V100 and MI250X, different benchmark than earlier slide
- MI250X has some disadvantages versus V100: smaller L1 cache, no overlapping integer operations with floating point compute

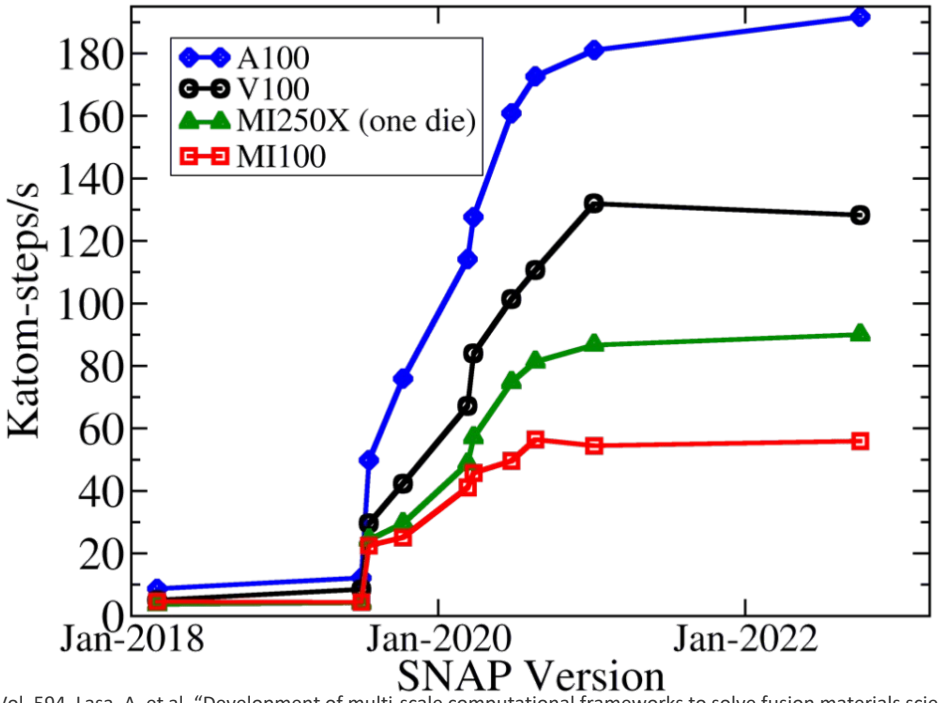


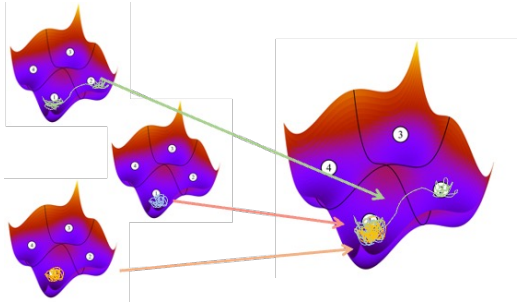
Figure from Journal of Nuclear Materials, Vol. 594. Lasa, A. et al, "Development of multi-scale computational frameworks to solve fusion materials science challenges," 155011, 2024,

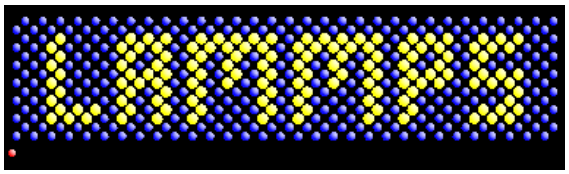
Code Changes Required for HIP

- Some raw CUDA code in neighborlist build was ported to Kokkos
- A few CUDA typedefs, template specializations, and macros were copied for HIP
- `#ifdef KOKKOS_ENABLE_CUDA` → `LMP_KOKKOS_GPU`
- A few parameters (e.g. team size, tile size, vector length) were tuned for HIP
- Added support for hipFFT (similar to cuFFT) for 1D FFTs
- **Overall, code changes very minimal**
- Similar case for SYCL and OpenMPTarget backends

EXAALT KPP1 on Frontier

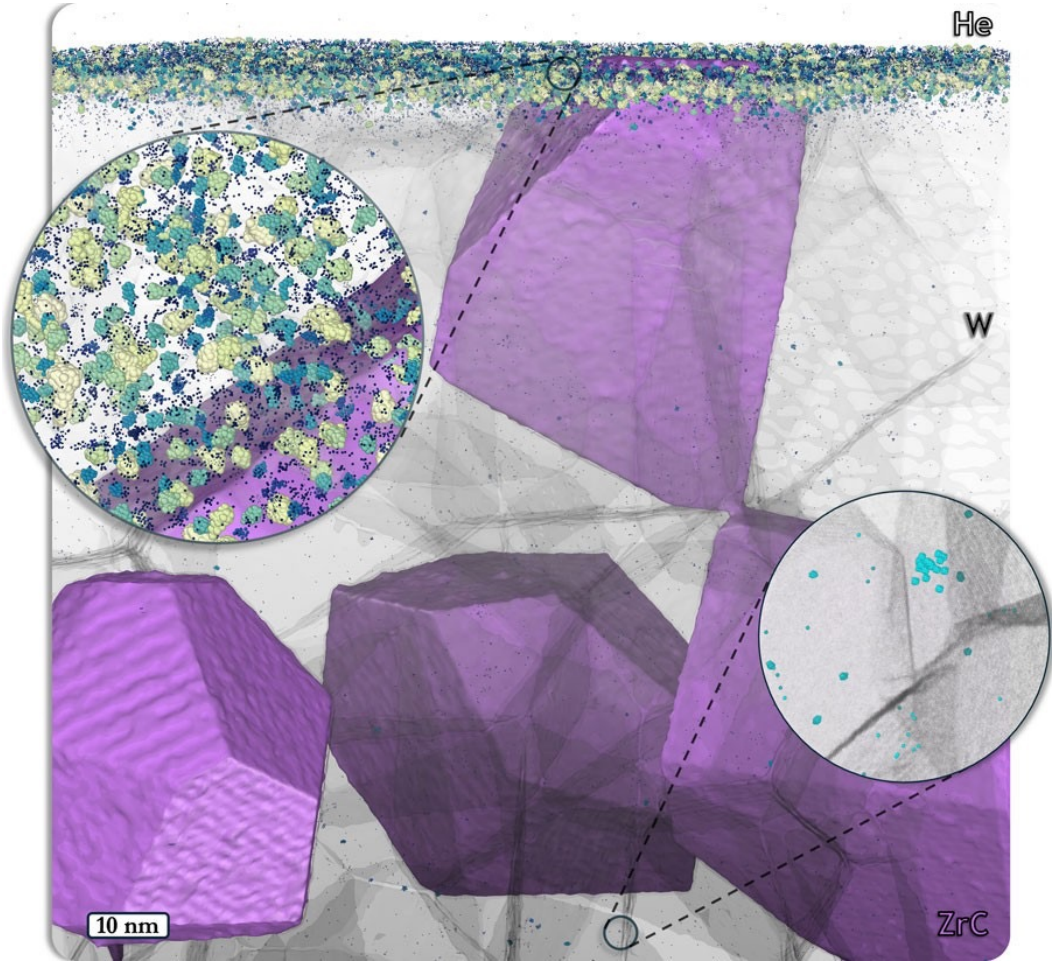
- EXAALT exascale computing project seeks to extend **accuracy, length, and time** scales of materials simulations
- EXAALT takes many replicas of LAMMPS and stitches them together to enable long-timescale simulations
- EXAALT team successfully executed their KPP1 simulation on 7000 Frontier nodes (~75% of full Frontier) using Kokkos HIP backend in LAMMPS
- Measured performance = **398.5x speedup over the Mira baseline**
- Extrapolation to full Frontier → **~530x** over Mira (ECP target was 50x)
- EXAALT also starting large-scale benchmarking on ALCF Aurora, successfully ran small test problem on 2048 nodes using SYCL backend



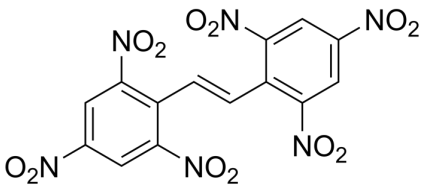


OLCF Frontier Science Simulation

- LAMMPS simulation of helium and neutron damage in tungsten zirconium carbide, modeled using SNAP
- Helium clusters form at the top surface of the polycrystalline tungsten
- Simulation used 46,656 AMD MI250X GCDs for approximately one (aggregate) Frontier-day
- The scale of this representative microstructure was enabled by SNAP performance improvements
- Work by Mitch Wood, SNL



Reactive Forcefield (ReaxFF)



Sandia National Laboratories

- LAMMPS ReaxFF is a Tier-1 CORAL-2 benchmark[1] (used for **OLCF Frontier acceptance testing**)
- ReaxFF was featured in AMD's MI200 public release[2,3]
- AMD (Nick Curtis, Leopold Grinberd, and Gina Sitaraman) contributed several Kokkos ReaxFF improvements to public LAMMPS
- AMD's work spurred additional improvements from Evan Weinberg (NVIDIA) and myself
- Since Feb. 2022: **1.54x** cumulative speedup on MI250X (one die) and **1.67x** speedup on V100 for 668K atom benchmark
- Great example of performance portability synergy via Kokkos

[1] https://asc.llnl.gov/coral-2-benchmarks
[2] https://www.amd.com/en/graphics/server-accelerators-benchmarks
[3] https://www.youtube.com/watch?v=zEKooI1UXYA

Production Simulations on NNSA Machines

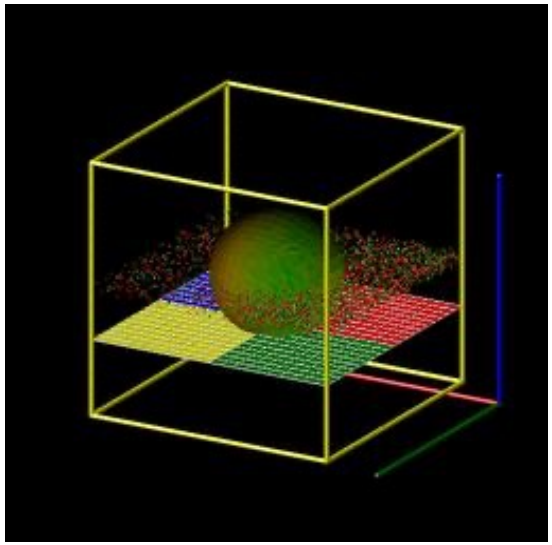
Sandia National Laboratories

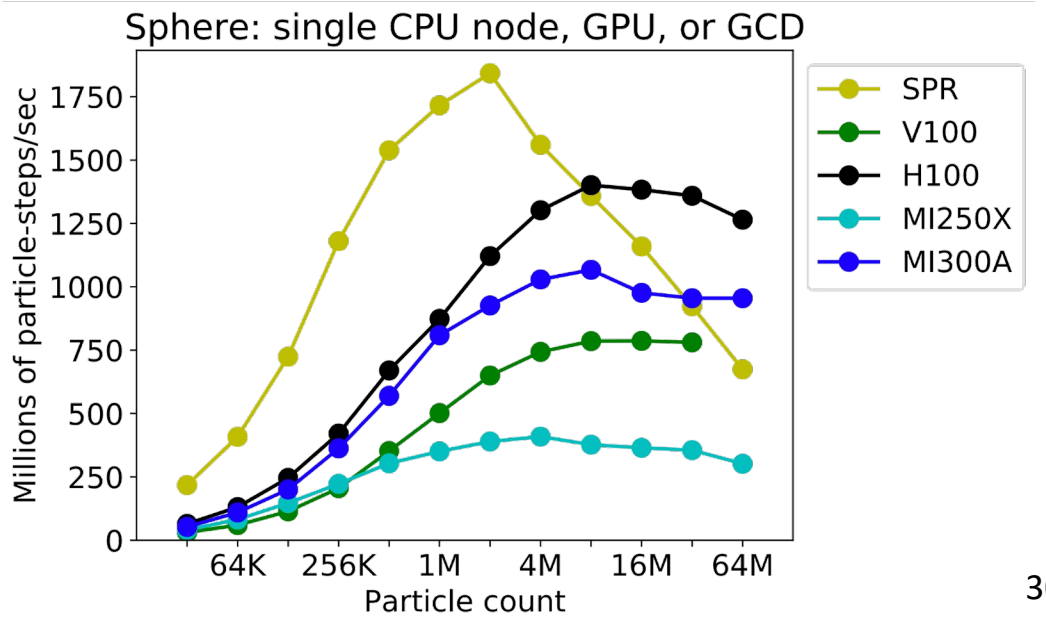
- SPARTA has run on entire NNSA **ATS-1** Trinity, on both Intel KNL and Haswell partitions simultaneously: ~19,000 nodes, over 1.2 million MPI ranks! Used Kokkos OpenMP and Serial backends
- LAMMPS ran ~1.5 billion atoms on 8192 NVIDIA V100 GPUs on NNSA's **ATS-2** Sierra (~47% of the full machine), using Kokkos CUDA backend. Now running 4x more atoms (~6 billion atoms)
- SPARTA and LAMMPS also running at large scale on **ATS-3** Crossroads (Intel Sapphire Rapid CPUs)
- Actively preparing for **ATS-4** El Capitan (AMD MI300A APUs)
- SPARTA and LAMMPS both part of future **ATS-5** procurement[1]

[1] https://mission.lanl.gov/advanced-simulation-and-computing/platforms/ats-5/

SPARTA Performance Portability

- Benchmark: particles flowing around a sphere
- Performance measured using Kokkos (GPUs) or MPI-only (CPU)
- Large cache effect for medium problem sizes on Intel Sapphire Rapids (SPR) CPU. Has high bandwidth memory (HBM) which improves performance
- GPUs (NVIDIA V100 & H100, AMD MI250X & MI300A) need large number of particles to saturate threads
- Some results are preliminary, more profiling/tuning work will be done in future





My Kokkos Wishlist

Sandia National Laboratories

- Better way(s) to debug missing *Kokkos::DualView* sync/modify. I have lost **days** of my life tracking down these issues. One tool I use (other than *printf*) is now deprecated... (UVM)
- Lower GPU kernel launch latency, which hurts performance of small simulations (e.g. low particle counts). May be some opportunities for improvement with recent CUDA versions
- Better vectorization on CPUs. Currently Kokkos OpenMP and Serial backends cannot compete with alternative LAMMPS INTEL package performance. Want to try out Kokkos SIMD, but requires significant code changes

Conclusions

- Kokkos performance portability abstractions greatly benefited LAMMPS and SPARTA (**huge success**)
- Significant performance gains for SNAP machine learning potential came from both algorithmic and GPU implementation improvements
- Optimizing SNAP for NVIDIA V100 gave large speedup on AMD MI250X and Intel PVC GPUs
- **Minimal code changes** required to switch from CUDA backend to HIP, SYCL, and OpenMPTarget backends
- More benefit to come on ALCF Aurora and ATS-4 El Capitan
- Looking forward to a performance portable future with Kokkos!

Sandia National Laboratories

Thank You
Questions?