Billion atom molecular dynamics simulations of carbon at extreme conditions and experimental time and length scales

Kien Nguyen-Cong*
nguyencong@usf.edu
University of South Florida
Tampa, FL, USA

Jonathan T. Willman*
jwillma2@usf.edu
University of South Florida
Tampa, FL, USA

Stan G. Moore
stamoor@sandia.gov
Sandia National Laboratories
Albuquerque, NM, USA

Anatoly B. Belonoshko
anatoly@kth.se
Royal Institute of Technology (KTH)
Stockholm, Sweden

Rahulkumar Gayatri
rgayatri@lbl.gov
NERSC
Berkeley, CA, USA

Evan Weinberg
eweinberg@nvidia.com
NVIDIA Corporation
Santa Clara, CA, USA

Mitchell A. Wood
mitwood@sandia.gov
Sandia National Laboratories
Albuquerque, NM, USA

Aidan P. Thompson
athomps@sandia.gov
Sandia National Laboratories
Albuquerque, NM, USA

Ivan I. Oleynik
oleynik@usf.edu
University of South Florida
Tampa, FL, USA

ABSTRACT

Billion atom molecular dynamics (MD) using quantum-accurate machine-learning Spectral Neighbor Analysis Potential (SNAP) observed long-sought high pressure BC8 phase of carbon at extreme pressure (12 Mbar) and temperature (5,000 K). 24-hour, 4650 node production simulation on OLCF Summit demonstrated an unprecedented scaling and unmatched real-world performance of SNAP MD while sampling 1 nanosecond of physical time. Efficient implementation of SNAP force kernel in LAMMPS using the Kokkos CUDA backend on NVIDIA GPUs combined with excellent strong scaling (better than 97% parallel efficiency) enabled a peak computing rate of 50.0 PFLOPs (24.9% of theoretical peak) for a 20 billion atom MD simulation on the full Summit machine (27,900 GPUs). The peak MD performance of 6.21 Matom-steps/node-s is 22.9 times greater than a previous record for quantum-accurate MD. Near perfect weak scaling of SNAP MD highlights its excellent potential to advance the frontier of quantum-accurate MD to trillion atom simulations on upcoming exascale platforms.

KEYWORDS

molecular dynamics, machine-learning interatomic potentials, carbon, extreme conditions

1 JUSTIFICATION FOR ACM GORDON BELL PRIZE

Peak 50.0 PFLOPS rate in quantum-accurate 20 billion atom molecular dynamics simulation, 6.21 Matom-steps/node-s MD performance - 22.9x improvement over previous record for quantum-accurate MD. Sustained real-world simulation of 1 billion carbon atoms for 1 nanosecond of physical time on 4,650 nodes of Summit during 24 hours of wall clock time.

2 PERFORMANCE ATTRIBUTES

Performance Attribute	Our Submission
Category of achievement	Time to solution, scalability
Type of method used	SNAP/Kokkos via LAMMPS MD
Results reported on basis of	Whole application including I/O
Precision reported	Double precision
System scale	Measured on full system
Measurement mechanism	Timers, FLOP count

3 OVERVIEW OF THE PROBLEM: CLASSICAL SIMULATIONS OF MATERIALS AT EXTREME CONDITIONS WITH QUANTUM ACCURACY

Recent exciting discoveries of thousands of exoplanets beyond our solar system has advanced the research on planetary materials at extreme pressures and temperatures to the forefront of physical sciences [1, 2]. A fundamental requirement for understanding the composition and the structure of exoplanetary interiors is an accurate knowledge of crystal structure, high pressure-temperature (PT) equations of state (EOS) and melting behavior of key geological materials. The advent of powerful laser [3] and pulsed-power [4] compressions, and in-situ X-ray free electron laser diffraction experiments [5] have made it possible to recreate and probe the high-PT environment of exoplanetary cores in the laboratory. However, a lack of theoretical and simulation guidance of experimental efforts, including comprehensive atomic-scale understanding of the complex physics of a material's response to extreme PT conditions, substantially limits the science return from these sophisticated but very expensive experiments. Such meaningful predictions require billion-atom MD simulations at experimental nanosecond and micrometer time and length scales.

*K. Nguyen-Cong and J. T. Willman contributed equally to this work.

SC '21, November 14–19, 2021, St. Louis, MO, USA

ACM ISBN 978-1-4503-8442-1/21/11…$15.00
https://doi.org/10.1145/3458817.3487400

Traditionally, quantum molecular dynamics (QMD) using density functional theory (DFT) is used to simulate matter at extreme PT conditions [6, 7]. Due to substantial computational costs, such simulations are limited to samples up to a thousand atoms and up to tens of picoseconds of simulation time, thus acquiring mostly equilibrium properties of the materials (e.g. EOS and static phase diagrams). These equilibrium models have been found to fail in the regimes where non-equilibrium time-dependent physics becomes important [8].

In principle, the intrinsic time and length scale limitations of QMD can be overcome by classical MD simulations which employ empirical interatomic potentials [9]. However, their intrinsic inability to cover a wide range of pressures and temperatures with sufficient accuracy is a serious roadblock towards high fidelity predictive simulations of planetary materials aimed at theoretical guidance of experiments.

Very recently, new exciting opportunities in atomistic simulations have emerged with the advent of machine-learning interatomic potentials (ML-IAPs) [10–13], which aim to provide a classical description of underlying interatomic interactions with DFT accuracy. Although several applications of ML-IAPs in materials modeling have already emerged [14–17], their exceptional potential in describing diverse and complex atomic environments occurring in materials subjected to extreme PT conditions has yet to be realized.

Our team has recently made a significant advance towards solving this extremely challenging but fundamentally important problem: predictive atomic-scale simulations of materials at extreme PT conditions at experimental time and length scales. In particular, we designed a quantum-accurate Spectral Neighbor Analysis Potential (SNAP) ML-IAP for carbon [18], which describes its properties at extreme conditions spanning pressures from 0 to 50 Mbars and temperatures up to 20,000-K, including the phase diagram, melting curves of diamond, BC8 and simple cubic phases of carbon, with accuracy systematically within 5% of very rigorous QMD results. Although such impressive results come with substantial computational cost, SNAP's linear scaling with number of atoms and its efficient implementation within the LAMMPS MD simulation package [19] allows us to run massively parallel billion atom simulations on leadership class DOE Summit, DOE Perlmutter, and NSF Frontera systems.

The current plan of work was specifically designed with the overarching goal of highlighting important algorithmic innovations in implementing SNAP MD on heterogeneous multi-core processors and many-core GPU accelerators in production simulations of carbon at extreme conditions. Carbon, existing in the form of graphite and diamond at ambient conditions, is expected to transform to a new crystalline form, so-called BC8 structure, at the extreme pressures of tens of millions of atmospheres (Mbars) and temperatures of tens of thousands of kelvins in the interiors of recently discovered carbon-rich exoplanets [20]. Multiple attempts to experimentally discover the new form of carbon at extreme conditions in laboratory have been so far unsuccessful [21–23].

This scientific challenge is a perfect candidate for demonstrating the transformative impact of quantum-accurate MD simulations at experimental time and length scales. Such accurate simulations involving billion of atoms, that have never been attempted before, became possible due to a recent breakthrough of our team in a very efficient implementation of SNAP on HPC platforms, utilizing GPU accelerators as well as an exclusive access to OLCF's Summit - one of the most powerful HPC systems in the world [24]. By efficiently utilizing all its 4650 nodes (27,900 GPUs), in this grand simulation we were able to uncover novel pathways towards synthesis of the elusive BC8 phase of carbon, while demonstrating unprecedented scaling and unmatched real-world performance of SNAP MD.

4 CURRENT STATE OF THE ART

With a suitable expression for forces, taken from derivatives of an interatomic potential (IAP) energy function, the classical equations of motion can be numerically integrated with a finite time step $\delta t \approx 1 \cdot 10^{-15}$ s to simulate the equilibrium and non-equilibrium dynamics of any biological/chemical/material system. Conventionally, empirical IAP derived from simplified descriptions of covalent/metallic/ionic bonding were used to obtain the atomic forces due to interactions with nearby atoms. Machine-learning potentials (ML-IAP) are driven by the need for accuracy approaching that of quantum electronic structure methods, while retaining the computational cost, linear scaling, and parallel efficiency of the empirical potentials. In a relatively short amount of time since their inception [25, 26] ML-IAP have been demonstrated to achieve accuracy comparable to that of electronic structure methods such as density functional theory (DFT).

In general, ML-IAP are constructed from three unique, but not completely separable parts; a descriptor set, regression technique, and model form. The descriptors (synonymous with features) encode the bonding environment around each atom (see Fig. 1) and play a critical role in both the absolute accuracy (w.r.t. DFT) and computational cost of the model. The most successful ML potentials developed to date can be divided into two classes according to the choice of model form, which are either kernel-based or neural network (NN) models. Examples of the latter include the Behler-Parrinello NNP [25], ANI [27, 28], HIP-NN [29] and DeepMD [30], each of which are subtly different based on the choices of descriptors and neural network architecture. The present work on SNAP falls into the class of kernel-based ML-IAP which also includes GAP [26], ChIMES [31] and MTP [32], where differences are predominately defined by the choice of descriptors. Kernel-based methods compute a similarity metric between each feature vector from MD and the database of ground-truth training structures. The most widely used similarity kernel is the squared exponential kernel used in Gaussian Process Regression models such as GAP [26]. It can be shown that SNAP, MTP, ChIMES and other linear models are equivalent to GAP when the squared exponential kernel is replaced with a dot product kernel [33].

Recent theoretical work by Drautz [34] has shown that SNAP, MTP, and several other successful descriptors are part of the Atomic Cluster Expansion (ACE) family of descriptors each with a particular choice of radial basis [35]. A recent comparison of the leading ML-IAP methodologies (including both NN and kernel-based methods) by an independent group showed that SNAP, GAP, and MTP (i.e. all kernel-based methods) provided the best balance between computational cost and accuracy [12].

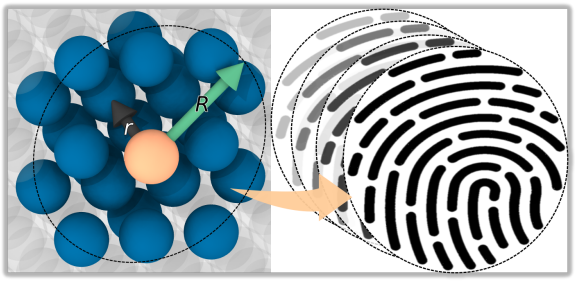

Figure 1: Schematic representation of ML descriptors encoding the local environment of an atom. All atoms within the radial cutoff (dashed line, *R*) are used to generate the descriptors, represented as fingerprints here. The atomic energy is expressed as a linear or nonlinear function of the descriptors, with parameters that are adjusted during training to minimize error w.r.t. DFT data.

The Spectral Neighborhood Analysis Potential (SNAP) approach pioneered by our team uses bispectrum components of the local neighbor density projected onto a basis of hyperspherical harmonics in four dimensions as descriptors, pictorially captured in Fig. 1. We use the quadratic form of SNAP for carbon, in which the atomic energy E^i_{SNAP} for an atom i is expressed as a sum of the bispectrum components $\mathbf{B}^i$ for that atom (see Section 5) and quadratic products of these descriptors, weighted by regression coefficients

$$E^i_{SNAP}(\mathbf{r}^N) \quad = \quad \boldsymbol{\beta} \cdot \mathbf{B}^i + \frac{1}{2}\mathbf{B}^i \cdot \boldsymbol{\alpha} \cdot \mathbf{B}^i \tag{1}$$

where the symmetric matrix $\boldsymbol{\alpha}$ and the vector $\boldsymbol{\beta}$ are constant linear coefficients whose values are trained to reproduce energies and forces obtained from DFT training structures. Similarly, the forces on each atom k are expressed in terms of the derivative of atomic energies with respect to the position of atom k, where N is the total number of atoms in the structure

$$\mathbf{F}^k_{SNAP} = -\nabla_k \sum_{i=1}^{N} E^i_{SNAP} = -\sum_{i=1}^{N} \left(\boldsymbol{\beta} + \mathbf{B}^i \cdot \boldsymbol{\alpha}\right) \cdot \frac{\partial \mathbf{B}^i}{\partial \mathbf{r}_k} \tag{2}$$

Training of the SNAP ML-IAP for carbon was performed iteratively utilizing the DAKOTA optimization package [36], wherein SNAP prediction errors were minimized with respect to DFT data. The $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ coefficients were determined by weighted linear regression minimizing the SNAP predicted energies and atomic forces relative to a database of DFT calculations. This resulted in a robust IAP over an astounding pressure and temperature range (0-50 Mbars and 300-20,000 K), far exceeding the capability of any empirical IAP.

The computational bottleneck in any MD simulation is the evaluation of the forces. In the case of SNAP (Eq. 2), this cost is dominated by the evaluation of the bispectrum components $\mathbf{B}^i$ for each atom, as well as the associated derivatives w.r.t. the positions of neighbor atoms $\partial \mathbf{B}^i / \partial \mathbf{r}_k$. In comparison to empirical IAP, nearly all ML-IAP are more computationally expensive given the complexity in the descriptor definitions, thus total atom counts and simulation times are sacrificed for the improved accuracy.

In previous work, we have demonstrated that kernel-based methods such as SNAP can uniquely take advantage of accelerator devices by exposing multiple levels of parallelism in the computational kernel that evaluates the gradients of descriptors needed for the MD force calculation. Trott *et al.* developed an early CUDA implementation of SNAP that achieved good computational efficiency on the NVIDIA K20x GPU. That work also demonstrated the excellent scalability of machine-learning potentials, allowing an MD simulation to run on the full Titan machine (18,688 GPUs) with only 13 atoms/GPU [37].

For comparison with other state of the art ML-IAP, in addition to the FLOP rate, a *universal normalized metric* for MD simulation throughput must be used. Namely, the performance of an MD simulation consisting of N_{atoms} simulated using N_{nodes} and completing N_{steps} within T_{sim} seconds is

$$\frac{N_{atoms}}{10^6} \frac{N_{steps}}{N_{nodes} \times T_{sim}} \tag{3}$$

and is reported in units of Matom-steps/node-s herein. Having both of these metrics, one for computational intensity and the other for simulation performance, is important when comparing MD simulations across various system sizes, hardware types, varied number of nodes, simulation time and disparate IAPs used.

Recently DeepMD [38], a NN based ML-IAP, reported $8.1 \cdot 10^{-10}$ s/atom-step for 100 timesteps with ~127 million Cu atoms on 4560 Summit nodes. The equivalent MD performance as defined in Eq. 3 is 0.271 Matom-steps/node-s, which currently stands as the best computational performance of any NN ML-IAP at this scale [39]. By comparison, our SNAP MD simulations reported in this paper have achieved an MD performance of 6.21 Matom-steps/node-s while simulating ~20 billion carbon atoms on the full Summit machine (4650 nodes), **which is 22.9x higher than the MD performance of DeepMD**. The next section details the algorithmic improvements that provided this performance gain.

5 INNOVATIONS REALIZED

Here we present the algorithmic and architecture specific optimizations that were made to SNAP in order to improve the throughput on newer generation CPUs and GPUs. The SNAP energy and forces are expressed as a basis expansion in bispectrum components (Eq. 1, 2) up to an upper limit in the angular momentum quantum number J, defined below. Exploitation of a symmetry relation in the bispectrum components reduced the computational complexity from $O(J^7)$ to $O(J^5)$, giving an order of magnitude speedup on CPUs [13]. This version of SNAP was ported to run on GPUs in LAMMPS and is the version that we took as our starting point [40].

As shown in the equations below, SNAP consists of many irregularly structured, deeply nested loops with small, varying loop sizes, increasing the challenges of optimization compared to regularly structured linear algebra kernels (e.g. GEMM).

The evaluation of the SNAP potential and derived forces follows the following pattern:

- `ComputeUi`: Evaluate the local neighbor density of an atom i in terms of a four-dimensional hyperspherical harmonic

basis,

$$\mathbf{U}_j = \sum_{r_{ik}<R} f_c(r_{ik})\mathbf{u}_j(a,b), \quad (4)$$

where $\mathbf{u}_j$ are Wigner U-matrices, each rank $2j+1$, and a, b are the Cayley-Klein parameters, mappings of $\mathbf{r}_{ik}$ to the 3-sphere, and the index j takes half-integer values $\{0, \frac{1}{2}, 1, \frac{3}{2}, \ldots\}$. The $\mathbf{u}_j$ are efficiently calculated by a recursion relation

$$\mathbf{u}_j = \mathcal{F}(\mathbf{u}_{j-\frac{1}{2}}), \quad (5)$$

where $\mathcal{F}$ is a linear operator mapping two adjacent elements of $\mathbf{u}_{j-1/2}$ to each element of $\mathbf{u}_j$. $f_c(r_{ik})$ is a smooth cutoff function.

- `ComputeBi` and `ComputeZi`: The $\mathbf{U}_j$ are not basis invariant and thus not useful as descriptors. We form real, scalar, basis-invariant triple-products [26]:

$$B_{j_1 j_2 j} = \mathbf{U}_{j_1} \otimes^{j}_{j_1 j_2} \mathbf{U}_{j_2} : \mathbf{U}^*_j \quad (6)$$

$$= \mathbf{Z}^{j}_{j_1 j_2} : \mathbf{U}^*_j. \quad (7)$$

  The symbol $\otimes^{j}_{j_1 j_2}$ indicates a Clebsch-Gordan product of matrices, an $O(j^4)$ operation. The : corresponds to an element-wise scalar product of two matrices of equal rank, an $O(j^2)$ operation. The vector of descriptors $\mathbf{B}^i$ for atom i introduced in Eq. 1 is a flattened list of elements $B_{j_1 j_2 j}$ restricted to $0 \le 2j_2 \le 2j_1 \le 2j \le 2J$, so that the number of unique bispectrum components scales as $O(J^3)$. In the current work, $2J$ is set to 8, yielding a descriptor vector $\mathbf{B}^i$ of length 55.
- `ComputeDuidrj` and `ComputeDeidrj`: Compute derivatives of the descriptors,

$$\frac{\partial B_{j_1 j_2 j}}{\partial \mathbf{r}_k} = \mathbf{Z}^{j}_{j_1 j_2} : \frac{\partial \mathbf{U}^*_j}{\partial \mathbf{r}_k} + \mathbf{Z}^{j_1}_{j j_2} : \frac{\partial \mathbf{U}^*_{j_1}}{\partial \mathbf{r}_k} + \mathbf{Z}^{j_2}_{j j_1} : \frac{\partial \mathbf{U}^*_{j_2}}{\partial \mathbf{r}_k}, \quad (8)$$

  and accumulate into the force via Eq. 2.

This section describes the implementation and optimizations of the quadratic SNAP ML-IAP given above that uses the Kokkos performance-portability library [41]. Kokkos provides a framework for decomposing work into discrete, independent pieces that are written in C++ and then mapped onto backend languages (such as CUDA) and dispatched in parallel, hiding the architecture-specific details of executing work. Kokkos provides constructs to exploit hierarchical parallelism. The most relevant here are multi-dimensional, tiled launches, which conceptually map onto cache blocking on the CPU and multi-dimensional thread and block indices on the GPU. Of special note, Kokkos provides an abstraction of "scratchpad memory", which conceptually maps onto small memory segments on the CPU which stay resident in cache, and maps onto shared memory on the GPU.

The first set of optimizations below describe the systematic extraction of hierarchies of parallelism in the SNAP ML-IAP. These are complimented by optimizations to memory layouts enabled by the Kokkos performance portable framework "view" abstraction for multi-dimensional data structures. The latter set of optimizations describe where the ideals of performance portability break down, and we diverge the implementations for the CPU and GPU. This is necessary because GPUs, compared to CPUs, require a far higher *arithmetic intensity*, or ratio of FLOPS to memory transactions, to take full advantage of hardware accelerators.

The implementation of the SNAP potential described here is publicly available[1] with the LAMMPS molecular dynamics package [19, 42] . The work we describe below was been performed over the past ~3 years, starting with the baseline GPU implementation [40] in LAMMPS.

5.1 Kernel Fission and Reduction of Computational Complexity

Despite taking advantage of the Kokkos features of hierarchical parallelism and scratchpad memory, the initial implementation of the SNAP potential had lackluster performance on GPUs. The original implementation mirrored the baseline CPU version by using one large, fused kernel, which caused high register usage, throttling occupancy.

Our first change was *kernel fission*, splitting the large kernel into multiple small kernels. This reduced register pressure across separate kernels, but greatly increased memory usage since intermediate quantities for all pairs of atom-neighbors needed to be explicitly stored between kernel launches.

These memory overheads became prohibitive and motivated several important optimizations. First, it motivated index flattening in both $\mathbf{U}_j$ and $\mathbf{Z}^{j}_{j_1 j_2}$, replacing jagged arrays with compressed indices. This innovation reduced the memory for $\mathbf{U}_j$ by a factor of 1/3 and considerably more for $\mathbf{Z}^{j}_{j_1 j_2}$.

More importantly, this motivated the development of the *adjoint refactorization*, which combines Eqs. 1 and 8 to define a new quantity $\mathbf{Y}$,

$$\mathbf{Y}_j = \sum_{j_1 j_2} (\boldsymbol{\beta} + \mathbf{B}\cdot\boldsymbol{\alpha})^{j}_{j_1 j_2} \mathbf{Z}^{j}_{j_1 j_2}. \quad (9)$$

This adjoint refactorization simplifies the final force evaluation to

$$\mathbf{F}^k_{SNAP} = -\sum_{i=1}^{N}\sum_{j=0}^{J} \mathbf{Y}_j : \frac{\partial \mathbf{U}^*_j}{\partial \mathbf{r}_k}. \quad (10)$$

$\mathbf{Y}$ can be identified as the adjoint of $\mathbf{dB}$ with respect to $\mathbf{dU}$. **This reduces the computational complexity from $O(J^5)$ to $O(J^3)$ by removing a factor of $O(J^2)$ computation from the evaluation of Eq. 8 compared to Eq. 10.** This reformulation also enables a factor of 3 reduction of flops due to a $j \leftrightarrow j_1 \leftrightarrow j_2$ symmetry in $\mathbf{Z}^{j}_{j_1 j_2}$. As part of the development of this method, we optimized away a factor of $O(N_{neigh})$ storage in $\mathbf{Z}$. The calculation of $\mathbf{Y}$ was implemented in a new kernel `ComputeYi`.

5.2 Extraction of Parallelism and Data Layout Optimizations

The acts of kernel fission and implementing the adjoint refactorization simplified identifying the parallelism available in each kernel. All four kernels noted below have trivial atom parallelism. Of further note:

[1] Production simulations in this work used the version of SNAP in [42], while scaling simulations were rerun using a slightly modified version of [42] optimized for large atom counts per GPU. These new optimizations have been recently released publicly in LAMMPS [43].

- `ComputeUi`: Eq. 4 offers additional neighbor parallelism if the sum over neighbors is performed atomically.
- `ComputeYi`: Eq. 9 offers additional quantum number parallelism if the sum over j_1, j_2 is performed atomically.
- `ComputeDuidrj`: Trivial neighbor parallelism across independent $\frac{\partial \mathbf{U}_j}{\partial r_k}$.
- `ComputeDeidrj`: Additional neighbor parallelism if force accumulations are performed atomically, exploiting Newton's Laws.

Each source of parallelism obeys *linearity*, meaning we are free to reorder the per-atom, per-neighbor, and per-quantum-number parallelism as appropriate to maximize performance. In the evaluation of $\mathbf{U}_j$, for example, we can choose to evaluate the contribution from all atoms one neighbor at a time, or compute the contribution from all neighbors one atom at a time. The hierarchical parallelism abstractions in Kokkos makes it easy to rearrange the order of parallelism. It also simplifies changing data layouts to promote good memory access on the CPU and GPU: Array-of-Structures on the CPU to promote spatial/temporal cache locality, Structure-of-Arrays on the GPU to promote memory alignment and coalescing.

The *trivial* parallelism over atom number across all kernels enables a perfect SIMD implementation. To exploit this we use an Array-of-Structures-of-Arrays (AoSoA) data layout, where the inner-most array dimension is a generic "`vector_length`" which can take a different interpretation on different architectures. On Intel CPUs, a vector length of 8 corresponds to one AVX512 SIMD register on which vector intrinsics can be performed. On NVIDIA GPUs, the number of *threads within a warp*, 32, ensures *warp convergence* as well as perfect memory alignment and coalescing for memory I/O, both necessary for the optimal use of GPU hardware.

One key benefit of the AoSoA data layout is ideal cache reuse across architectures because the data for each vectorized set of atoms is contiguous in memory, ensuring full cache pages are utilized. The AoSoA data layout naturally extends to hierarchical "chunking" of work, where the SNAP force evaluation can be grouped in batches of atoms large enough to saturate the GPU but small enough to avoid large memory allocations.

We note there are a few additional short bandwidth-bound kernels for data initialization and staging purposes.

In its current state, the SNAP calculation is performant on the GPU. It is not yet optimal on the CPU due to difficulties with autovectorization; we are considering the use of the Kokkos SIMD type or the Cabana library [44] as a point of future work.

5.3 Increasing Arithmetic Intensity in Recursive Polynomial Evaluations

Our next significant performance improvement came from optimizing the calculation of the Wigner U-matrices and derivatives thereof. The core observation can be summarized in one sentence: *evaluating recursive polynomials is an inherently compute intensive process.* For the Wigner U-matrices themselves, there are only two inputs: the complex Cayley-Klein parameters a and b. All subsequent outputs are recursive linear combinations thereof.

The original implementation of `ComputeUi`, the kernel responsible for evaluating Eq. 4, took a *breadth-first* approach to the recursive evaluation of each $\mathbf{u}_j$. Each iteration computed half of $\mathbf{u}_j$ from $\mathbf{u}_{j-\frac{1}{2}}$ and applied the symmetry properties of the Wigner U-matrices to compute the other half. After all $\mathbf{u}_j$ were computed, a second pass through all $\mathbf{u}_j$ performed the rescaling by the cutoff function and atomic accumulation into $\mathbf{U}_j$. Each step of this process was staged through global memory.

With this access pattern, we cannot expect L1 cache reuse to circumvent the memory costs due to the quadratic scaling of the number of elements in $\mathbf{u}_j$ with j. Minimizing bandwidth overheads required four separate optimizations:

(1) Re-write the recursive polynomial evaluation as a hybrid *depth/breadth* evaluation, depth first in rows of $\mathbf{u}_j$. This reduces intermediate state overheads by a factor of j, promoting cache reuse.
(2) Inline the atomic accumulation into $\mathbf{U}_j$ with the recursive polynomial calculation, eliminating a later reload.
(3) Explicitly cache intermediate values in shared memory.
(4) Remove all global memory staging by introducing a redundant work model, re-computing some rows of $\mathbf{u}_{j'<j}$ in the process of computing rows of $\mathbf{u}_j$.

The changes described above translate well to the derivative of the Wigner U-matrix in the routine `ComputeDuidrj`, albeit with subtle changes. We split the evaluation of the x, y, and z kernels into three separate kernels to avoid a prohibitive increase in shared memory overheads relative to `ComputeUi`. In addition, we fused the accumulation of the force, Eq. 10, into the recursive polynomial evaluation, eliminating the kernel `ComputeDeidrj`. The read of $\mathbf{Y}$ can be partially hidden behind the heavy computational intensity of the recursive polynomial calculation. This new, fused kernel is renamed to `ComputeFusedDeidrj`. **These changes realize the compute-bound nature of recursive polynomial evaluation.**

Last, we precompute the Cayley-Klein parameters once in a new kernel to offset unnecessary redundant work in `ComputeUi` and each of three invocations of `ComputeFusedDeidrj`.

The improvements described in this subsection reduce the overall kernel runtime for recursive polynomial evaluation by a factor of 12.3x relative to [45] on a V100 GPU, even as it increases discrete FLOPS by 1.73x due to the redundant work in the hybrid depth/breadth approach. In its optimized form, `ComputeUi` and `ComputeFusedDeidrj` sustain 1.63 and 2.47 TFLOPS/sec, respectively, on a V100. This is a direct consequence of our novel approach which eliminates all global memory staging.

We note that we cannot directly compare the costs of individual kernels due to the combination of kernel fusion and fission.

5.4 Kernel Optimizations for the Adjoint Representation in Quadratic SNAP

The last speedups came from kernel optimizations within the calculation of the adjoint matrix $\mathbf{Y}$. In the case of linear SNAP, we can bypass the intermediate storage of $\mathbf{Z}$ by direct atomic addition into $\mathbf{Y}$ via Eq. 9. In the case of *quadratic* SNAP, we need to precompute $B_{j_1 j_2 j}$ before accumulating $\mathbf{Y}$. This necessitates computing each value of $\mathbf{Z}$ independent of the `ComputeYi` kernel.

We note that we can store the pre-computed values of $\mathbf{Z}$ for reuse in the subsequent calculation of $\mathbf{Y}$, saving the $O(J^2)$ overheads of computing each component of $\mathbf{Z}$ twice. This is implemented in

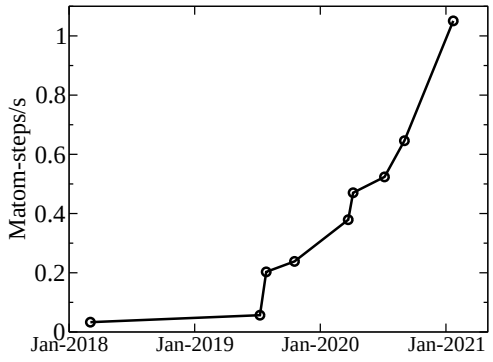

Figure 2: Time progression of performance for successive optimizations of the SNAP Kokkos implementation in LAMMPS running an MD benchmark on a single NVIDIA V100 GPU. The current implementation is now over 30 times faster than the baseline Kokkos implementation in 2018.

a new kernel `ComputeYiFromZlist`, which has a 5.2x faster runtime than the original `ComputeYi` in [45]. The `ComputeZi` kernel is already highly optimized, sustaining 1.89 TFLOPS/sec on a V100.

5.5 Performance Summary

The overall effect of the optimizations described in this section [42] (and those described in Footnote 1) was a **35.4x increase in performance compared to the original Kokkos implementation** [40] on a single V100 GPU for a carbon benchmark, measured on Summit using CUDA 11.2.0. A benchmark for the SNAP tungsten model [46] had similar performance improvements over time as shown in Fig. 2.

The analysis of single GPU performance limiters is complicated by the diverse algorithmic motifs within the SNAP calculation. The most expensive kernel, the recursive evaluation of $\partial \mathbf{U}_j / \partial r_k$, is compute limited. The next-most expensive kernel, calculating components of $\mathbf{Z}^j_{j_1 j_2}$, is bound by L1 bandwidth (as opposed to L2 or global memory) due to optimal cache reuse patterns. The calculation of $\mathbf{U}_j$ is bound by double-precision atomic performance, nonetheless it is optimal because the alternative is the original, far less performant approach of global memory staging.

6 HOW PERFORMANCE WAS MEASURED

6.1 Science application used to measure performance in our production simulations

We have carefully designed our production simulations to demonstrate the real-world performance of our quantum-accurate SNAP MD simulations in ground-breaking production runs with the aim of delivering important scientific results. Our goal was to perform a production simulation that runs on the entire Summit HPC system (4,650 nodes) continuously for 24 hours (with possible restarts if simulation is aborted due to hardware failures) using the largest possible system ensuring maximum resource utilization, while measuring in-situ computational performance of the simulation.

Our science goal is to observe the phase transformation of amorphous carbon (a-C) to the high pressure BC8 phase of carbon at 12 Mbar and temperatures between 5,000-5,500 K. We are part of a joint simulation-experiment discovery science collaborative team, which aims to synthesize the elusive BC8 carbon phase using powerful lasers at National Ignition Facility (NIF) at Lawrence Livermore National Laboratory. We performed preliminary simulations using small samples containing 0.3 million atoms to narrow the pressure-temperature (P-T) range where this transformation occurs. Therefore, the major objective for the production run was to push the time and length scales of our simulations to nanoseconds and hundreds of nanometers, the scales characteristic of ultrafast laser compression experiments. By confirming a-C→BC8 transformation, our simulation results help minimize a probability of failure in these very expensive (~ $ million per shot) NIF experiments.

Our project was planned in two stages. In the first preparation stage we determined the maximum possible system size (that attains the maximum computational efficiency) during a production run of a a-C sample compressed to 12 Mbar and heated to ~5,000 K, while still giving 1 ns/day of simulation throughput. In addition to preparing for full-scale production runs at stage 2, this study allowed us to uncover the true scaling potential of innovative implementation of SNAP MD on GPU platforms.

The scaling simulations were performed for a-C at 12 Mbars and 5,000 K, using a constant number of particles (N), volume (V) and temperature (T) (NVT ensemble) with the Langevin thermostat to control temperature. The a-C sample was computationally "synthesized" by rapid quenching liquid carbon at 6,000 K to 300 K in 100 ps, followed by a series of annealing cycles (heating to 2,000 K and cooling to 300 K). The quality of our computational synthesis of a-C sample is judged by reproducing experimentally observed fraction of sp^3-coordinated C atoms in tetrahedral a-C - 86.4% at zero pressure. Once the a-C sample is prepared at ambient conditions, it is compressed to a target pressure (12 Mbars) and several annealing cycles are repeated to allow the atomic structure to adjust to that of high-pressure a-C phase. To generate the samples of varying sizes, the original 0.3 million atom sample is replicated in each Cartesian direction and additional NVT MD simulations were run at the initial temperature (~5,000 K) for 20 ps to break the artificial periodicity due to cell replication.

An MD timestep size of 0.5 fs was chosen to ensure conservation of energy, as verified in a separate NVE test run that approximately matched the same pressure and temperature of the production run. The number of steps for the scaling studies performed is 100, which is sufficient to get a consistent measure of average MD performance. The measured performance for both weak and strong scaling tests does not include infrequent I/O such as writing binary checkpoint files and per-atom quantities to "dump" text files. However, this cost is accurately measured during our production run during stage 2 of the project.

Once the scaling studies were finished and the optimal size of the sample to achieve maximum performance on all 4,650 nodes was determined, we proceeded with the second stage – production simulations of the a-C→BC8 transition at the exact same PT

Table 1: Properties of supercomputers utilized [24]

Machine	Top500 Rank	Nodes	GPU Accel.	FP64 PFLOPS Peak	FP64 PFLOPS LINPACK
Summit	2	4,672	Y	200.8	148.6
Perlmutter	5	1,536	Y	89.8	64.6
Selene	6	560	Y	79.2	63.5
Frontera	10	8,008	N	38.7	23.5

Table 2: FLOP count for different kernels.

Kernel	FLOP/atom-step
SNAP force	1.731×10^6
Verlet time integrator	40
Langevin thermostat	19
Total	1.731×10^6

conditions as in the previous scaling studies. We designed our simulation workflow to maximize the science return from this expensive, ~111,600 node-hour simulation by sampling a-C→BC8 transitions at various temperatures to gain an insight into kinetics of the phase transition at large scale (Fig. 7). If transition happened at a particular temperature, the simulation continued until the entire sample was converted to BC8 as judged by monitoring changes in potential energy per atom. Then the temperature was set to another value, the initial sample was read in from restart file and a new NVT simulation was performed at the new temperature. Within our production runs, consisting of roughly 24 hours (cumulative), we managed to sample three temperatures, 5,000, 5,300 and 5,500 K (Fig. 7).

6.2 HPC Systems and Environment

The majority of our scaling and production runs were performed on Summit, the leadership-class HPC system at Oak Ridge National Laboratory. Summit consists of 4,672 nodes featuring dual-socket IBM Power9 CPUs and 6 NVIDIA V100-16GB GPUs connected together using dual-rail Mellanox EDR 100G InfiniBand, non-blocking Fat Tree interconnect topology.

In order to compare GPU and CPU performance, we also ran scaling tests on the TACC Frontera leadership class HPC (CPU-only) system at University of Texas. Frontera consists of 8,008 nodes featuring dual-socket 28-core Intel Xeon Platinum 8280 (Cascade Lake) CPUs connected by Mellanox Infiniband HDR-100 interconnect.

Finally we ran scaling tests on Selene and the latest addition to the Top500 list, the Perlmutter machine. Selene consists of 560 nodes featuring dual-socket AMD EPYC 7742 CPUs and 8 NVIDIA A100-80GB GPUs connected by octorail Mellanox HDR 200GB Infiniband interconnect. Perlmutter is comprised of 1536 nodes where each node has 4 NVIDIA A100-40GB GPUs and a single AMD EPYC 7763 (Milan) CPUs. The nodes are connected by HPE Cray Slingshot network switches and network interface cards (NICs).

Further details on each machine are given in Table 1.

The LAMMPS code including the SNAP ML-IAP was compiled on Summit with CUDA 11.2.0, GNU 7.4.0, and IBM Spectrum MPI 10.3.1.2-20200121. The Kokkos library with the CUDA backend was used for GPU acceleration. All runs were performed using one MPI rank per GPU. LAMMPS at Frontera was compiled with Intel 19.1.1 and Intel MPI 19.0.9. The original (non-Kokkos) version of SNAP was used on Frontera. LAMMPS on Selene was compiled with CUDA 11.2.1, GNU 9.3.0, and OpenMPI 4.1, consistently using the Kokkos library with the CUDA backend, with one MPI rank per GPU. LAMMPS on Perlmutter was compiled with CUDA 11.0, GNU 9.3.0 and Cray MPICH.

On Summit we used CUDA-aware MPI, but explicitly turned off the GPU-direct driver-level communication mode using the "-gpu -disable_gdr" flags for IBM Spectrum MPI. We found turning off GPU-direct gave better performance for 1 billion atoms on the full machine than the default settings (not shown here). On Selene we also used CUDA-aware MPI, and on Perlmutter we used CUDA-aware MPI that is available with the latest Cray MPICH.

6.3 Measurement metrics

In our performance measurements we used two major metrics - (1) the total number of floating point operations per second (FLOPS) and (2) normalized MD throughput, referred to as "MD performance". Although the first metric is self-explanatory, the second is not. In this work, MD performance is measured in millions of atom steps per node per second (Matom-steps/node-s), which is an ideal normalized performance metric for comparing the performance of large-scale MD simulations on HPC platforms, even between different IAP types, atom and node counts. Eq. 3 details how this MD specific performance metric is calculated. FLOPS were measured using the NVIDIA CUDA *nvprof* tool. All the scaling and production simulations were performed using double precision floating point arithmetic.

The number of floating point operations in the SNAP force calculation depends on the number of atoms as well as the number of neighbors, which in turn depends on the density of the sample. The average number of neighbors for the initial configuration of the small amorphous sample (used for benchmarking) was 27.1. For the carbon model we fixed the other parameters that affect the FLOP count, such as the radial cutoff distance of 2.7 Angstrom, 55 bispectrum coefficients, and the use of quadratic as opposed to linear SNAP.

The number of double precision floating point operations for a 46,656 atom amorphous carbon sample run for 100 timesteps was measured on a single GPU. At the beginning and end of the 100 timestep run, energy and pressure were computed and thermodynamic output was written to the log file. The *flops_dp* metric in nvprof captures non-atomic double precision floating point adds and multiplies. Double precision atomic additions as motivated in Section 5 were counted by hand and verified using the NVIDIA Nsight Compute tool's instruction counters. The SNAP force computation, the Verlet time integrator, and the Langevin thermostat kernels were profiled and the results are shown in Table 2. Other kernels such as packing/unpacking MPI communication buffers and the occasional building of the neighbor list contribute negligible FLOPS. The total average number of floating point operations (including atomic-adds) for this sample was found to be 1.731×10^6 per atom-step.

7 PERFORMANCE RESULTS

The small amorphous sample profiled previously was replicated in three dimensions to give ~20 billion (19,683M) atoms. Due to this periodic repetition we expect the number of floating point operations measured previously to scale with the number of atoms for this larger benchmark. We then ran the 20 billion atom simulation on 27,900 GPUs for 100 timesteps, and the timers in the LAMMPS log file reported an average performance of 6.21 Matom-steps/node-s, or equivalently 1.47 timesteps per second. We note that the LAMMPS timers only measure the MD timestep loop, so time setting up the run and MPI initialization/finalization are not included. **Combining this performance with the previously measured FLOP count gives 50.0 PFLOPS (double precision) on the full Summit machine, or 24.9% of the theoretical peak computing rate.** Our SNAP implementation has no regular linear algebra kernels yet achieves one-third of the measured LINPACK performance on Summit, highlighting the extent of our optimizations for GPUs.

The DeepMD NN-based ML-IAP [30] recently reported a double precision time-to-solution of 8.1×10^{-10} s/step/atom for ~127 million copper atoms on 4560 Summit nodes [39], which is equivalently an MD performance of 0.271 Matom-steps/node-s. For 20 billion carbon atoms on 4650 Summit nodes, **our MD performance of 6.21 Matom-steps/node-s is 22.9x higher than what DeepMD reported, meaning our SNAP ML-IAP is significantly more efficient than DeepMD while still achieving *ab initio* accuracy.**

A maximum of 4,662 nodes can be requested per job in the "batch" queue on Summit. We chose to run on 4,650 nodes (27,900 GPUs) for two reasons. The first is that 27,900 MPI ranks factor into a 3D grid of nearly equal values: 30×30×31, minimizing the surface-to-volume ratio of the communication halo exchange regions for our cubic simulation box. The second is that this provides a small buffer of extra nodes in case one or more nodes go down or are running at sub-optimal performance.

To minimize variation and maximize performance in both scaling and production runs, we ran a small LAMMPS test job on every GPU independently. The average runtime was calculated, and any GPUs that were significantly slower than the average were recorded. While running our scaling and production runs, several (~15) GPUs outside this criterion were detected and the corresponding nodes were reported to the system administrators. To mitigate the effect of these unhealthy nodes on the performance of our full system runs, we reserved 4,660 nodes (10 nodes more than we use in production 4,650 node runs) and used the LAMMPS test job to generate a list of slow nodes on-the-fly. These nodes were then automatically excluded at runtime when the MPI driver "jsrun" was executed in the submission script.

A strong scaling study was performed on the amorphous carbon sample running for 100 MD timesteps, as shown in Fig. 3. We used several sample sizes: 1, 10, and 100 million, and 1, 4 and 20 billion atom a-C samples while varying number of nodes from the minimum possible to all 4,650 nodes, the former being the minimum number of nodes a particular sample size can fit into, e.g. 64 nodes for 1 billion atom a-C sample and 972 nodes for 20 billion atom a-C sample.

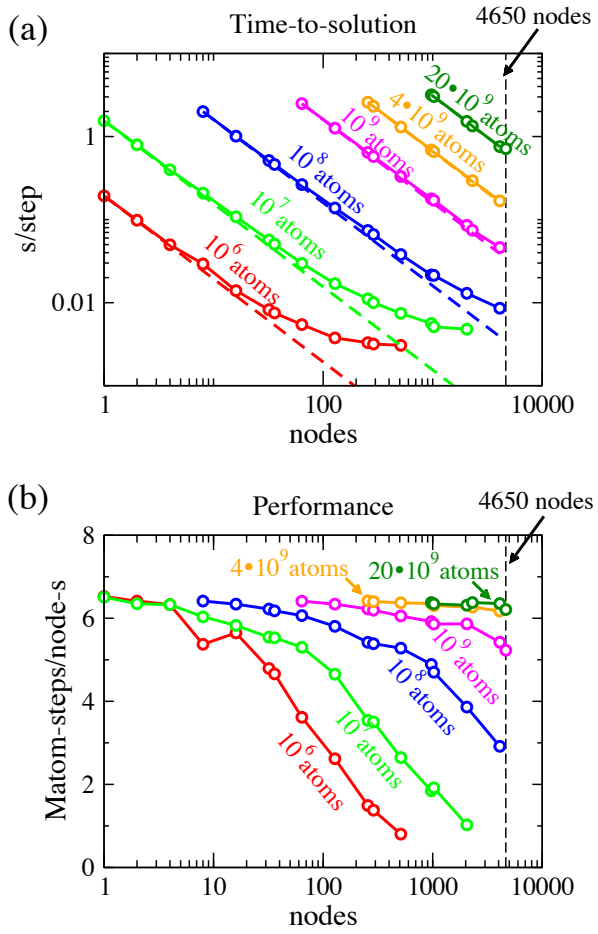

Figure 3: Strong scaling: (a) time to solution in seconds per step and (b) MD performance for amorphous carbon samples with 1,259,712; 10,077,696; 102,503,232; 1,024,192,512; 4,251,528,000 and 19,683,000,000 atoms. Total loop time was measured for 100 MD steps. Perfect scaling is shown in (a) as dashed lines. Perfect scaling in (b) would be a horizontal line (not shown).

Fig. 3 shows excellent strong scaling behavior up to the full machine for samples with billions of atoms. For example, the 20 billion atom simulation has 97% parallel efficiency when comparing the performance of 4,650 nodes to 972 nodes. The 1 billion atom simulation has 82% parallel efficiency when comparing 4,650 nodes to 64 nodes. The 10 million atom simulation has 41% parallel efficiency when comparing 512 nodes to 1 node.

Fig. 4 shows a breakdown of the timings as reported by the LAMMPS log file for different sample sizes on the full machine. The relative percentage of communication grows as the computational load decreases, hence increasing the atom count per GPU increases the efficiency at full scale.

Fig. 5 shows weak scaling behavior using 373,248 atoms/node (62k atoms/GPU), scaling from 1 to 4096 nodes and run for 100 MD timesteps. There is a small drop in performance going from 8 to 64

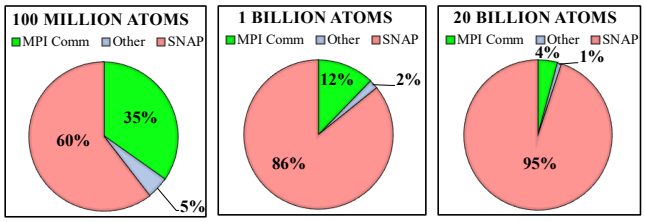

Figure 4: A breakdown of the time spent for different atom counts on the full machine as measured by the timers in LAMMPS. "SNAP" indicates time spent in the force computation, "MPI Comm" indicates time spent in communication, and "Other" indicates time spent in I/O, the Langevin thermostat, Verlet time integration, and other services.

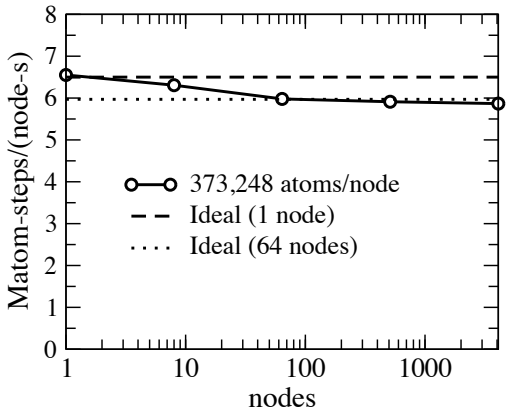

Figure 5: Weak scaling for amorphous carbon samples measured as MD performance vs node count. Sample sizes range from 373,248 atoms to 1,528,823,808 atoms and correspond to 373,248 atoms/node. Ideal scaling comparing to 1 and 64 nodes is indicated by dashed and dotted horizontal lines, respectively.

nodes which is to be expected: Summit nodes are grouped in racks of 18 and there is an associated inter-rack communications penalty when crossing this threshold. Beyond this threshold excellent weak scaling behavior resumes, giving 90% parallel efficiency on 4096 nodes compared to 1 node. This experiment affirmed that 373,248 atoms/node at full machine scale would provide a simulation rate of 1 ns per day.

Fig. 6 compares the performance of the amorphous carbon benchmark on 4 out of the top 10 fastest HPC machines (as of June 2021), strong scaling the 1 billion atom amorphous benchmark. Performance on Summit is approximately 52 times faster per node than on Frontera. Performance on Selene is about 1.9x faster than Summit per node. For example, with a 20 billion atom run on 512 nodes of Selene, we achieved 12.72 Matom-steps/node-s, which corresponds to 11.14 PFLOPS or 14% of peak for the full machine. **A similar 20 billion atom run on 1024 nodes of Perlmutter achieved a 6.42 Matom-steps/node-s, corresponding to 11.24 PFLOPS on just two-thirds of the machine.**

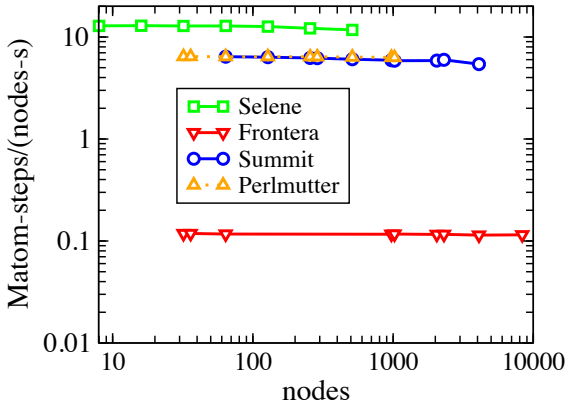

Figure 6: Comparison of performance between TACC Frontera, OLCF's Summit, NERSC's Perlmutter, and NVIDIA's Selene supercomputers for an a-C sample containing 1,024,192,512 atoms.

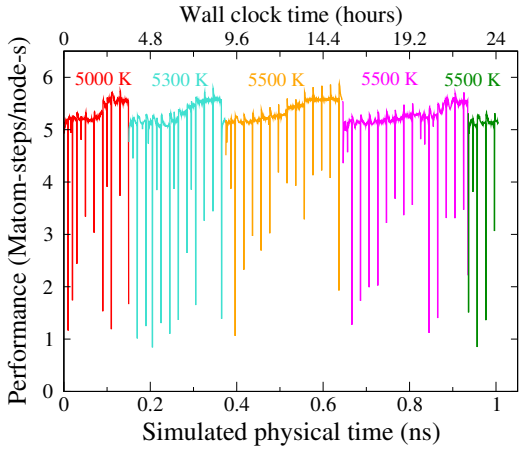

Figure 7: Sustained performance for realistic production science runs of a-C→BC8 transformation. Simulation consisted of 1,024,192,512 atoms on 4650 nodes. Loop time was measured every 1000 MD steps. Large dips in performance show file I/O (e.g. writing binary checkpoint files). The small rise in average performance in each simulation is associated with the emergence of the ordered BC8 phase.

We note that the FP64 peak FLOPS on Selene and Perlmutter includes the usage of FP64 tensor cores on the NVIDIA A100, which cannot be utilized with SNAP because the evaluation does not map onto matrix multiplication. It is not surprising that Selene has twice the Matom-steps/node-s of Perlmutter because it has two times the number of NVIDIA A100 GPUs. We also see a rough performance parity between one Summit node and one Perlmutter node despite Perlmutter having two fewer GPUs, owing to the generational improvements between the two machines.

Fig. 7 shows the performance of our science production run, investigating the a-C→BC8 phase transition at different temperatures. Color represents restarts at different temperatures. Significant

I/O was performed periodically during the runs, corresponding to dips in the performance in Fig. 7. Atom coordinates, velocities, and potential energy were written to text files (~70 GB) every 40k timesteps (20k during equilibration) for subsequent analysis. Binary checkpoint files (~85 GB) were also written at the same frequency. In order to greatly improve file I/O performance, each MPI rank wrote to its own separate file in parallel on the file system. Thermodynamic output and timing was written to the log file every 1000 steps. An increase in performance can also be seen as the disordered amorphous phase transitions to the ordered BC8 phase in different parts of the production run.

Another periodic performance overhead is rebuilding the neighbor list. The neighbor list used an extra "skin" distance of 2.0 Angstroms. Atom coordinates were checked every 10 timesteps and the list was rebuilt only if one or more atoms had moved past half of the skin distance.

The part of the 1 billion atom production represented by the orange curve in Fig. 7 had an average sustained performance (including I/O) of 5.24 Matom-steps/node-s for over 6 hours, which corresponds to a 1.03 ns/day simulation rate. Other parts of the science run shown by different colors in Fig. 7 had similar performance.

As noted previously, the LAMMPS timers only measure inside the MD timestep loop (including I/O). However, the total LAMMPS setup time (including reading a checkpoint file) for the 1 billion production run was less than 4 minutes, which is small compared to the 6 hours of sustained performance mentioned previously. The MPI initialization/finalization time on 4650 nodes is also small: the entire time to initialize, run, and finalize the 1 billion amorphous benchmark for 100 timesteps is only ~3 minutes. Therefore, we consider the 5.24 Matom-steps/node-s as performance of the "whole application including I/O". **The performance of the 1 billion atom production run is only 16% lower than that of the 20 billion atom amorphous benchmark (6.21 Matom-steps/node-s) which achieved 50.0 PFLOPS.** Neglecting the cost of I/O, the performance of the production run is expected to be slightly higher than a similar amorphous benchmark since the production run includes simulating the ordered BC8 crystal, which runs faster than the amorphous phase as shown in Fig. 7.

A point of future work is the use of reduced precision within the SNAP calculation. At this time the numerical and thermodynamic stability of such an implementation over a long simulation time is an open question outside of the scope of this work, though there is a long-standing precedent for the viability of reduced precision approaches in many, but not all, IAP calculations [47, 48]. We do have a pure single precision implementation of the SNAP potential (results not shown); the literature on other IAP suggests that a hybrid single/double or single/fixed precision implementation would be necessary for simulation stability.

8 IMPLICATIONS/IMPACT

Our simulation project was specifically designed to showcase the transformative science advance enabled by a combination of algorithmic innovations in the implementation of quantum-accurate, machine-learning SNAP molecular dynamics on GPUs and an exclusive access for 24 hours to the entire Summit machine, the

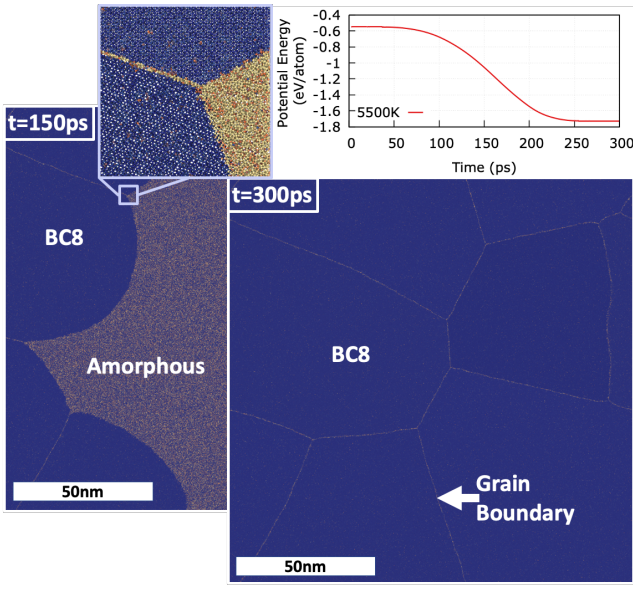

Figure 8: 1 billion carbon atom MD simulation of the a-C → BC8 phase transition (Top) Average potential energy per atom versus time showing a global phase transition from the higher energy a-C precursor to the more stable BC8 crystalline form. (Bottom) Visualizations of the partially and fully transformed material, with atoms colored by potential energy. The darker regions are the lower energy BC8 phase. Note the thin lines of higher energy indicating locations where multiple BC8 nuclei have grown into one another forming nascent grain boundaries.

second most powerful HPC system in the world. We have successfully achieved our overarching objective – to demonstrate record-breaking sustained performance (5.24 Matom-steps/node-s) of SNAP MD during the 111,600 node-hour production run on all 4,650 Summit nodes, while simulating a billion atom carbon sample at experimentally relevant hundreds of nanometers and nanosecond length and time scales.

Our production run "computationally synthesized" the long-sought BC8 phase of carbon emerging from the amorphous carbon (a-C) precursor at extreme conditions of density and temperature. Our simulation uncovered the fundamental mechanism for the a-C→BC8 phase transformation which involves nucleation and growth of polycrystalline grains of BC8 phase in amorphous matrix, see Fig. 8. The 0.2 nanosecond time scale over which the simulated transition occurs is accessible to in-situ dynamic X-ray diffraction imaging during ultra-fast laser-driven dynamic compression experiments at the DOE/NNSA National Ignition Facility (NIF).

The broader science impact of our project is the demonstration of the unprecedented power of extreme-scale quantum-accurate MD simulations to predict novel physical phenomena and guide experiments towards observing them, thus avoiding time consuming trial and error experimentation. Our simulation team is part of a joint computational-experimental collaboration that aims to perform experiments at NIF and also Sandia National Laboratories'

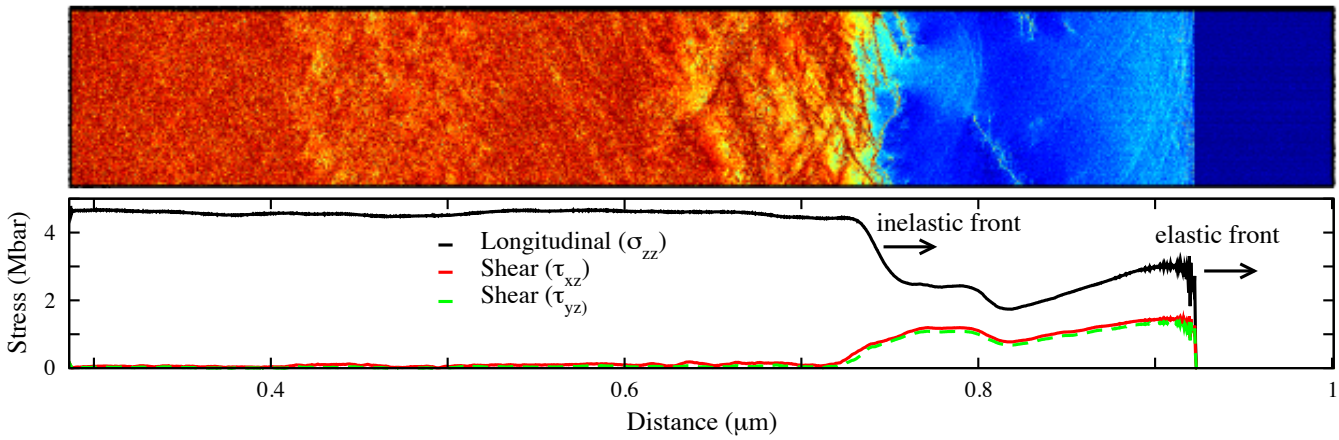

Figure 9: SNAP MD simulation of split elastic-inelastic shock wave propagating along <110> crystallographic direction in single crystal diamond. The compression is driven by a piston moving with constant velocity $v_p = 7.0$ km/s. The sample contains 1.756 billion atoms with cross-section 0.1×0.1 μm^2 and length 1 μm. The elastic precursor propagates with the velocity 22.3 km/s. The second inelastic wave, colored red in the top image, propagates with the velocity 18.3 km/s, while displaying an unexpected mechanism of stress relaxation in brittle diamond.

Z pulsed power facilities, both of which are planning future experiments based on our predictive simulations. Another illustrative example of the unprecedented insight provided by accurate MD simulations at this scale are the simulations of shock wave propagation in a micron-thick diamond sample, which have uncovered a novel atomic-scale mechanism of inelastic deformation rendered in Fig. 9. *Thus billion atom MD simulations are critical to make direct connection with experiment.* By measuring the time-dependent velocity of the sample's rear surface due to the arrival of the compression wave with very fine spatial (up to 0.1 nm) and time (less than 2 ps) resolution in experiment, directly comparing with that obtained in MD simulations complemented by atomically-resolved, frame-by-frame images of non-equilibrium processes behind compressive wave from MD (see, for example, Fig. 9), fundamental mechanisms of phase transitions under dynamic loading will be uncovered.

The excellent HPC performance of the SNAP MD force kernel in the LAMMPS molecular dynamics package enabled by the Kokkos CUDA backend will be directly transferable to planned leadership computing platforms. In the case of the Intel GPU Aurora exascale machine planned for Argonne National Laboratory, the DPC++/SYCL (OneAPI) and/or OpenMPTarget Kokkos backends will allow us to port our code with minimal changes [49]. Similarly in the case of the AMD GPU Frontier exascale machine planned for Oak Ridge National Laboratory, the Kokkos HIP backend will complement the CUDA backend that we used on Summit [50]. On these machines, the code optimizations described here will allow us to advance the frontiers of quantum-accurate MD simulations towards simulating trillion atom samples. Moreover, these algorithmic and performance improvements are extendable to future emerging heterogeneous node architectures containing multi-core processors and many-core accelerators.

The performance achievements demonstrated here are also not limited to this particular application or material system. For example, the same excellent SNAP performance has also been demonstrated in many other computational materials science applications, including simulations of plasma-material interactions [46, 51], and radiation damage [52] on leadership platforms that make use of NVIDIA GPUs (ORNL Summit, LLNL Sierra). Because the code we have developed here has been merged into the public distribution of LAMMPS, it is available to the growing number of independent research groups that have adopted SNAP for developing and running their own machine-learning potentials.

ACKNOWLEDGMENTS

The work at USF is supported by DOE/NNSA (grant # DE-NA-0003910). The SNAP/Kokkos code optimization was supported by the CoPA and EXAALT projects within the Exascale Computing Project (ECP, No. 17-SC-20-SC). ECP is a collaborative effort of the U.S. Department of Energy Office of Science and the National Nuclear Security Administration. Computations were performed using leadership-class HPC systems: OLCF Summit at Oak Ridge National Laboratory (ALCC and INCITE awards MAT198) and TACC Frontera at University of Texas at Austin (LRAC award #DMR21006). 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. The authors thank OLCF team (Don Maxwell, Jens Glaser, and Bronson Messer) and TACC team (John Cazes, Tim Cockerill and Dan Stanzione) for providing excellent support of our production simulations on entire OLCF Summit and TACC Frontera HPC systems. This research used resources of the National Energy Research Scientific Computing Center (NERSC), which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.

REFERENCES

[1] Didier Queloz. Nobel Lecture: 51 Pegasi b and the exoplanet revolution. *Rev. Mod. Phys.*, 92:30503, 2020.

[2] Michel Mayor. Nobel Lecture: Plurality of worlds in the cosmos: A dream of antiquity, a modern reality of astrophysics. *Rev. Mod. Phys.*, 92:30502, 2020.

[3] J. R. Rygg et al. X-ray diffraction at the National Ignition Facility. *Rev. Sci. Instrum.*, 91, 2020.

[4] D. B. Sinars et al. Review of pulsed power-driven high energy density physics research on Z at Sandia. *Phys. Plasmas*, 27:070501, 2020.

[5] Malcolm I. McMahon. *Synchrotron and FEL Studies of Matter at High Pressures*, pages 1857–1896. Springer International Publishing, Cham, 2020.

[6] Martin French and Thomas R. Mattsson. Thermodynamically constrained correction to ab initio equations of state. *J. Appl. Phys.*, 116:013510, 2014.

[7] Lorin X. Benedict, Kevin P. Driver, Sebastien Hamel, Burkhard Militzer, Tingting Qi, Alfredo A. Correa, A. Saul, and Eric Schwegler. Multiphase equation of state for carbon addressing high pressures and temperatures. *Phys. Rev. B*, 89:224109, 2014.

[8] Thomas S. Duffy and Raymond F. Smith. Ultra-high pressure dynamic compression of geological materials. *Front. Earth Sci.*, 7:1–20, 2019.

[9] Steven J. Plimpton and Aidan P. Thompson. Computational aspects of many-body potentials. *MRS Bull.*, 37:513–521, 2012.

[10] Volker L. Deringer, Miguel A. Caro, and Gábor Csányi. Machine Learning Interatomic Potentials as Emerging Tools for Materials Science. *Advanced Materials*, 31:1902765, 2019.

[11] Jörg Behler. Perspective: Machine learning potentials for atomistic simulations. *J. Chem. Phys.*, 145, 2016.

[12] Yunxing Zuo, Chi Chen, Xiangguo Li, Zhi Deng, Yiming Chen, Jörg Behler, Gábor Csányi, Alexander V. Shapeev, Aidan P. Thompson, Mitchell A. Wood, and Shyue Ping Ong. Performance and cost assessment of machine learning interatomic potentials. *J. Phys. Chem. A*, 124:731–745, 2020.

[13] Aidan P Thompson, Laura P Swiler, Christian R Trott, Stephen M Foiles, and Garritt J Tucker. Spectral neighbor analysis method for automated generation of quantum-accurate interatomic potentials. *Journal of Computational Physics*, 285:316–330, 2015.

[14] Miguel A. Caro, Volker L. Deringer, Jari Koskinen, Tomi Laurila, and Gábor Csányi. Growth Mechanism and Origin of High sp3 Content in Tetrahedral Amorphous Carbon. *Phys. Rev. Lett.*, 120:166101, 2018.

[15] Volker L. Deringer, Noam Bernstein, Gábor Csányi, Chiheb Ben Mahmoud, Michele Ceriotti, Mark Wilson, David A. Drabold, and Stephen R. Elliott. Origins of structural and electronic transitions in disordered silicon. *Nature*, 589:59–64, 2021.

[16] M. A. Wood, M. A. Cusentino, B. D. Wirth, and A. P. Thompson. Data-driven material models for atomistic simulation. *Phys. Rev. B*, 99:1–12, 2019.

[17] Conrad W. Rosenbrock, Konstantin Gubaev, Alexander V. Shapeev, Livia B. Pártay, Noam Bernstein, Gábor Csányi, and Gus L. W. Hart. Machine-learned interatomic potentials for alloys and alloy phase diagrams. *npj Computational Materials*, 7:24, 2021.

[18] Jonathan T Willman, Ashley S Williams, Kien Nguyen-Cong, Aidan P Thompson, Mitchell A Wood, Anatoly B Belonoshko, and Ivan I Oleynik. Quantum accurate SNAP carbon potential for MD shock simulations. *AIP Conf. Proc.*, 2272:070055, 2020.

[19] S. Plimpton. Fast parallel algorithms for short-range molecular dynamics. *J. Comp. Phys.*, 117:1–19, 1995. https://lammps.org/.

[20] C. Hakim, K.; van Westrenen, W.;Dominik, Kaustubh Hakim, Rob Spaargaren, Damanveer S. Grewal, Arno Rohrbach, Jasper Berndt, Carsten Dominik, and Wim Van Westrenen. Mineralogy, Structure, and Habitability of Carbon-Enriched Rocky Exoplanets: A Laboratory Approach. *Astrobiology*, 19:867–884, 2019.

[21] J. H. Eggert, D. G. Hicks, P. M. Celliers, D. K. Bradley, R. S. McWilliams, R. Jeanloz, J. E. Miller, T. R. Boehly, and G. W. Collins. Melting temperature of diamond at ultrahigh pressure. *Nat. Phys.*, 6:40–43, 2010.

[22] R. F. Smith, J. H. Eggert, R. Jeanloz, T. S. Duffy, D. G. Braun, J. R. Patterson, R. E. Rudd, J. Biener, A. E. Lazicki, A. V. Hamza, J. Wang, T. Braun, L. X. Benedict, P. M. Celliers, and G. W. Collins. Ramp compression of diamond to five terapascals. *Nature*, 511:330–333, 2014.

[23] A. Lazicki, D. McGonegle, J. R. Rygg, D. G. Braun, D. C. Swift, M. G. Gorman, R. F. Smith, P. G. Heighway, A. Higginbotham, M. J. Suggit, D. E. Fratanduono, F. Coppari, C. E. Wehrenberg, R. G. Kraus, D. Erskine, J. V. Bernier, J. M. McNaney, R. E. Rudd, G. W. Collins, J. H. Eggert, and J. S. Wark. Metastability of diamond ramp-compressed to 2 terapascals. *Nature*, 589:532–535, 2021.

[24] Top500. https://www.top500.org/lists/2021/06/. [Online; accessed 23-July-2021].

[25] Jörg Behler. Atom-centered symmetry functions for constructing high-dimensional neural network potentials. *J. Chem. Phys.*, 134:074106, 2011.

[26] Albert P. Bartók, Mike C. Payne, Risi Kondor, and Gábor Csányi. Gaussian approximation potentials: The accuracy of quantum mechanics, without the electrons. *Phys. Rev. Lett.*, 104:136403, 2010.

[27] Justin S. Smith, Ben Nebgen, Nicholas Lubbers, Olexandr Isayev, and Adrian E. Roitberg. Less is more: Sampling chemical space with active learning. *The Journal of Chemical Physics*, 148:241733, 2018.

[28] Justin S Smith, Benjamin Nebgen, Nithin Mathew, Jie Chen, Nicholas Lubbers, Leonid Burakovsky, Sergei Tretiak, Hai Ah Nam, Timothy Germann, Saryu Fensin, et al. Automated discovery of a robust interatomic potential for aluminum. *Nature communications*, 12:1–13, 2021.

[29] Nicholas Lubbers, Justin S. Smith, and Kipton Barros. Hierarchical modeling of molecular energies using a deep neural network. *The Journal of Chemical Physics*, 148:241715, 2018.

[30] Linfeng Zhang, Jiequn Han, Han Wang, Roberto Car, and Weinan E. Deep potential molecular dynamics: A scalable model with the accuracy of quantum mechanics. *Phys. Rev. Lett.*, 120:143001, 2018.

[31] Rebecca K. Lindsey, Laurence E. Fried, and Nir Goldman. Chimes: A force matched potential with explicit three-body interactions for molten carbon. *Journal of Chemical Theory and Computation*, 13:6222–6229, 2017.

[32] Alexander V. Shapeev. Moment tensory potentials: a class of systematically improvable interatomic potentials. *Multiscale Model. Simul.*, 14:1153, 2016.

[33] Jonathan Vandermause, Steven B Torrisi, Simon Batzner, Yu Xie, Lixin Sun, Alexie M Kolpak, and Boris Kozinsky. On-the-fly active learning of interpretable bayesian force fields for atomistic rare events. *npj Computational Materials*, 6:1–11, 2020.

[34] Ralf Drautz. Atomic cluster expansion of scalar, vectorial, and tensorial properties including magnetism and charge transfer. *Phys. Rev. B*, 102:024104, 2020.

[35] Yury Lysogorskiy, Cas van der Oord, Anton Bochkarev, Sarath Menon, Matteo Rinaldi, Thomas Hammerschmidt, Matous Mrovec, Aidan Thompson, Gábor Csányi, Christoph Ortner, and Ralf Drautz. Performant implementation of the atomic cluster expansion (pace) and application to copper and silicon. *npj Computational Materials*, 7:97, 2021.

[36] B. M. Adams, K. R. Dalbey, M. S. Eldred, D. M. Gay, L. P. Swiler, W. J. Bohnhoff, J. P. Eddy, and K. Haskell. Dakota users manual version 5.1. Sandia Technical Report SAND2010-2183, Sandia National Laboratories, Albuquerque, NM, 2011.

[37] Christian R. Trott, Simon D. Hammond, and Aidan P. Thompson. SNAP: Strong scaling high fidelity molecular dynamics simulations on leadership-class computing platforms. In Julian Martin Kunkel, Thomas Ludwig, and Hans Werner Meuer, editors, *Supercomputing*, pages 19–34, Cham, 2014. Springer International Publishing.

[38] Weile Jia, Han Wang, Mohan Chen, Denghui Lu, Lin Lin, Roberto Car, E Weinan, and Linfeng Zhang. Pushing the Limit of Molecular Dynamics with Ab Initio Accuracy to 100 Million Atoms with Machine Learning. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–14. IEEE, 2020.

[39] Denghui Lu, Han Wang, Mohan Chen, Lin Lin, Roberto Car, Weinan E, Weile Jia, and Linfeng Zhang. 86 pflops deep potential molecular dynamics simulation of 100 million atoms with ab initio accuracy. *Computer Physics Communications*, 259:107624, 2021.

[40] LAMMPS. Github commit. https://github.com/lammps/lammps/commit/39a09d3, 2018.

[41] H. Carter Edwards, Christian R. Trott, and Daniel Sunderland. Kokkos: Enabling manycore performance portability through polymorphic memory access patterns. *Journal of Parallel and Distributed Computing*, 74:3202 – 3216, 2014. Domain-Specific Languages and High-Level Frameworks for High-Performance Computing.

[42] LAMMPS. Github commit. https://github.com/lammps/lammps/commit/c9742056d, 2021.

[43] LAMMPS. Github commit. https://github.com/lammps/lammps/commit/e363b4a, 2021.

[44] Stuart Slattery, Christoph Junghans, Damien Lebrun-Grandie, Shane Fogerty, Robert Bird, Sam Reeve, Guangye Chen, Rene Halver, Aaron Scheinberg, Cameron Smith, and Evan Weinberg. Ecp-copa/cabana: Version 0.3.0. https://doi.org/10.5281/zenodo.3785678, 2020.

[45] LAMMPS. Github commit. https://github.com/lammps/lammps/commit/fe9f7f4, 2019.

[46] Mitchell A Wood and Aidan P Thompson. Extending the accuracy of the SNAP interatomic potential form. *The Journal of Chemical Physics*, 148:241721, 2018.

[47] Andreas Götz, Mark Williamson, Dong Xu, Duncan Poole, Scott Grand, and Ross Walker. Routine microsecond molecular dynamics simulations with amber on gpus. 1. generalized born. *Journal of chemical theory and computation*, 8:1542–1555, 2012.

[48] M. Höhnerbach, A. E. Ismail, and P. Bientinesi. The vectorization of the tersoff multi-body potential: An exercise in performance portability. In *SC '16: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 69–81, 2016.

[49] B. Homerding. Preparing applications for aurora. Argonne technical report, Argonne National Laboratory, Chicago, IL, 2021. https://ecpannualmeeting.com/assets/overview/sessions/Preparing_Applications_for_Aurora.pdf.

[50] P. C. Roth. Developing software for olcf frontier. Oak ridge technical report, Oak Ridge National Laboratory, Oak Ridge,TN, 2021. https://ecpannualmeeting.com/assets/overview/sessions/Roth-Path-to-Frontier-20200206.pdf.

[51] M.A. Cusentino, M.A. Wood, and A.P. Thompson. Beryllium-driven structural evolution at the divertor surface. *Nuclear Fusion*, 61:046049, 2021.

[52] Mary Alice Cusentino, Mitchell A Wood, and Aidan P Thompson. Explicit multielement extension of the spectral neighbor analysis potential for chemically complex systems. *The Journal of Physical Chemistry A*, 124:5456–5464, 2020.