29-Billion Atoms Molecular Dynamics Simulation With *Ab Initio* Accuracy on 35 Million Cores of New Sunway Supercomputer

Xun Wang, Xiangyu Meng, Zhuoqiang Guo, Mingzhen Li, Lijun Liu, Mingfan Li, Qian Xiao, Tong Zhao, Ninghui Sun, Guangming Tan, *Senior Member, IEEE*, and Weile Jia

***Abstract*—Physical phenomena such as bond breaking and phase transitions require molecular dynamics (MD) with *ab initio* accuracy, involving up to billions of atoms and over nanosecond timescales. Previous state-of-the-art work has demonstrated that neural network molecular dynamics (NNMD) like deep potential molecular dynamics (DeePMD), can successfully extend the temporal and spatial scales of MD with *ab initio* accuracy on both ARM and GPU platforms. However, the DeePMD-kit package is currently unable to fully exploit the computational potential of the new Sunway supercomputer due to its unique many-core architecture, memory hierarchy, and low precision capability. In this paper, we re-design the DeePMD-kit to harness the massive computing power of the new Sunway, enabling the MD with over ten billion atoms. We first design a large-scale parallelization scheme to exploit the massive parallelism of the new Sunway. Then we devise specialized optimizations for the time-consuming operators. Finally, we design a novel mixed precision method for DeePMD-kit customized operators to leverage the low precision computing power of the new Sunway. The optimized DeePMD-kit achieves 67.6 / 56.5 × speedup for water / copper systems on the new Sunway. Meanwhile, it can perform 29 billion atoms simulation for the water system on 35 million cores (i.e., 90,000 computing nodes, around 84% of the whole supercomputer) with a peak performance of 57.1 PFLOPs, which is 7.9× bigger and 1.2× faster than state-of-the-art results. This paves the way for investigating more realistic scenarios, such as studying the mechanical properties of metals, semiconductor devices, batteries, and other materials and physical systems.**

***Index Terms*—High Performance Computing, Molecular Dynamics, DeePMD, Parallel Optimization, New Sunway Supercomputer.**

I. Introduction

PHYSICAL phenomena such as bond breaking [1], complex chemical process [2], quantum dot [3] require molecular dynamics (MD) simulations with ab initio accuracy, which can involve modeling millions of atoms over timescales of nanoseconds. This high level of precision is crucial for accurately capturing the intricate details of atomic interactions and the dynamics of complex systems. A major challenge in this field is managing the spatial and temporal scale of MD simulations. For instance, accurately simulating the pH level of water higher than 8 requires modeling on at least 10^8 water molecules with nanoseconds simulation.

Although molecular dynamics (MD) simulations based on empirical force fields (EFF) [4], [5], [6], can handle millions to trillions of atoms, their accuracy remains uncertain. Developing EFFs for systems with multiple elements or those involving bond formation and cleavage remains particularly challenging. Additionally, many practical problems lack suitable EFFs. Recently, force fields like the REAXFF method [4] and the Tersoff method [7] have gained significant attention for their ability to model chemical reactions. But they lack the versatility and predictive accuracy of density functional theory (DFT) [8], [9]. However, the spatial and temporal scales of traditional *ab initio* molecular dynamics (AIMD) simulations based on DFT are constrained to thousands of atoms and picoseconds. This limitation arises from the cubic computational complexity of solving the eigenvalue problem, a fundamental challenge even when utilizing GPUs for acceleration [10], [11], [12]. Low-scaling methods such as pole expansion and selected inversion [13], designed to reduce computational complexity, have enabled simulations involving millions of atoms [14], [15]. This progress, while significant, is still insufficient for simulating the

Received 13 August 2024; revised 21 November 2024; accepted 5 February 2025. Date of publication 11 February 2025; date of current version 9 April 2025. This work was supported in part by the National Science Foundation of China under Grant 62372435, Grant T2125013, Grant 62032023, Grant 61972377, Grant 92270206, Grant 61972380, and Grant 61972416; in part by the CAS Project for Young Scientists in Basic Research under Grant YSBR-005; in part by the China National Postdoctoral Program for Innovative Talents under Grant BX20240383; in part by Natural Science Foundation of Shandong Province under Grant ZR2022LZH009; in part by GHfund C under Grant 202407035455; and in part by National Key R&D Program of China under Grant 2021YFA1000103-3. Recommended for acceptance by J. Zhai. *(Xun Wang and Xiangyu Meng contributed equally to this work.) (Corresponding authors: Xun Wang; Mingzhen Li; Weile Jia.)*

Xun Wang and Xiangyu Meng are with the Department of Computer Science and Technology, Shandong Key Laboratory of Intelligent Oil & Gas Industrial Software, China University of Petroleum (East China), Qingdao, Shandong 266580, China (e-mail: wangsyun@upc.edu.cn; x_meng0420@163.com).

Zhuoqiang Guo, Mingzhen Li, Tong Zhao, Ninghui Sun, Guangming Tan, and Weile Jia are with the State Key Lab of Processors, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China, and also with the University of Chinese Academy of Sciences, Beijing 100190, China (e-mail: guozhuoqiang20z@ict.ac.cn; limingzhen@ict.ac.cn; zhaotong@ict.ac.cn; snh@ict.ac.cn; tgm@ict.ac.cn; jiaweile@ict.ac.cn).

Lijun Liu is with the Department of Mechanical Engineering, Osaka University, Suita, Osaka 565-0871, Japan (e-mail: liu@mech.eng.osaka-u.ac.jp).

Mingfan Li and Qian Xiao are with the School of Computer Science and Technology, University of Science and Technology of China, Hefei 230027, China (e-mail: mingfan@mail.ustc.edu.cn; xqbeida@mail.ustc.edu.cn).

Digital Object Identifier 10.1109/TC.2025.3540646

TABLE I
COMPARISON ACROSS *AB INITIO* MOLECULAR DYNAMICS APPLICATIONS, WHERE TTS DENOTES THE TIME TO SOLUTION, BP AND DP DENOTE THE BEHLER-PARRINELLO SCHEME AND DEEP POTENTIAL

Application	Year	Potential	System	#Atoms	#CPU Cores	#GPUs	Machine	Peak(FLOPS)	TtS(s/atom/step)
Simple-NN [19]	2019	BP	SiO_2	14K	80	–	Unknown	–	3.6×10^{-5}
Singraber el al. [36]	2019	BP	H_2O	9K	512	–	VSC	–	1.3×10^{-6}
SNAP ML-IAP [37]	2021	SNAP	C	1B	27.3K	27.3K	Summit	50P	3.4×10^{-11}
Allegro [30]	2023	Allegro	HIV capsid	100M	1280	5120	Perlmutter	–	7.9×10^{-11}
DeePMD-kit [31]	2022	DP	Cu	3.4B	27.3K	27.3K	Summit	43.7P	1.1×10^{-10}
DeePMD-kit [31]	2022	DP	Cu	1.1B	0.5M	–	Fugaku	7.4P	6.7×10^{-10}
DeePMD-kit [31]	2022	DP	H_2O	3.9B	27.3K	27.3K	Summit	46.3P	8.9×10^{-11}
DeePMD-kit [31]	2022	DP	H_2O	1.5B	0.5M	–	Fugaku	7.8P	4.6×10^{-10}
This work	2024	DP	Cu	13B	35M	–	new Sunway	41.4P	1.5×10^{-10}
This work	2024	DP	H_2O	29B	35M	–	new Sunway	57.1P	8.8×10^{-11}

pH level of 8 for the water system (10^8 molecules), and the time-to-solution is only tens to hundreds of femtoseconds of simulation time per day on top-tier supercomputers.

Neural network molecular dynamics (NNMD) can effectively balance accuracy and computational efficiency [16] by fitting the high-dimensional nonlinear potential energy surface (PES) using data generated from first-principles calculations [17]. This approach was pioneered by the Behler and Parrinello (BP) schemes [18], such as Simple-NN [19], and TensorMol [20], which calculates the atom-centered symmetry functions (ACSFs) as the hand-crafted descriptor and serves as the inputs of the fitting net to predict the PES. Deep potential molecular dynamics (DeePMD) [21] provides an end-to-end solution by constructing a symmetry-preserving embedding network as a descriptor, which is used to learn the translation, rotation, and permutation invariances of the molecular systems. It can achieve the accuracy of AIMD while reducing computation costs. Currently, the open-source implementation of DeePMD called DeePMD-kit [22] has been applied in many physical problems such as the phase diagram of water [23], water-in-salt electrolytes (WiSE) properties [24], and infrared spectroscopy and Raman spectroscopy [25] etc. Additionally, recent NNMDs like SchNet [26], PhysNet [27], NEquIP [28], PaiNN [29], and Allegro [30] leverage the graph neural networks (GNNs) to perform more accurate MD.

With the help of high performance computing (HPC), NNMD packages can enable large-scale MD simulations that were previously infeasible due to computational limitations. Table I shows the comparative overview of the simulation scales and achieved performance of several representative NNMD implementations on modern supercomputers. Among them, the DeePMD-kit incorporates a series of optimization designs tailored for ARM CPUs and NVIDIA GPUs. It has achieved an unprecedented 3.9 billion atoms simulation on Summit and 1.5 billion atoms simulation on Fugaku with nanoseconds per day [31], outperforming other NNMDs. Notably, the claim of simulating 10 billion atoms on the Fugaku supercomputer in [31] is based on projected scaling rather than actual test data. Thus, our analysis and comparisons rely solely on the tested results reported in the paper. Meanwhile, the new Sunway supercomputer provides a brand-new opportunity to further extend the temporal and spatial scales of the DeePMD-kit. Recent works have redesigned and accelerated molecular dynamics software by leveraging the advanced capabilities of the new Sunway processor, including the optimization of the AIREBO [32] and Tersoff [33] potentials, as well as the FHI-AIMS package [34].

Although DeePMD-kit has been successfully adapted to platforms such as Summit and Fugaku [31], [35], it is currently unable to fully utilize the computational power of the new Sunway. This is primarily due to the unique architectural features of the new Sunway processor, the SW26010-pro, which features a heterogeneous many-core design with a total of 390 cores. We face several challenges in optimizing the DeePMD-kit for the new Sunway: First, the existing parallelization strategies for CPU and GPU platforms are inefficient for the new Sunway architecture, resulting in under-utilization of the CPE cores, which account for 99% of the system's theoretical computing power. Second, many of the DeePMD-kit operators are memory bandwidth-bound, but the memory bandwidth of the SW26010-pro processor is limited to 300GB/s, significantly lower than that of contemporary GPUs, thus hindering the efficient use of computational units. Third, the SW26010-pro provides substantial low precision floating-point computational capabilities, which could be exploited to reduce memory access overhead and accelerate computations. However, this necessitates an advanced compression scheme that maintains accuracy while accelerating performance.

In this paper, we redesign the DeePMD-kit to address the aforementioned challenges. We tailor the parallelization, operator optimization, and mixed precision scheme specifically for the new Sunway supercomputer. These enhancements enable us to perform molecular dynamics simulations with ab initio accuracy for systems comprising up to 29 billion atoms. To the best of our knowledge, this represents the largest MD simulation with ab initio accuracy achieved to date. Our source code is publicly available here[1]. Specifically, the main contributions are as follows:

1) We propose a multi-level parallelization scheme for the DeePMD-kit on the new Sunway, which covers MPI, SACA, and SIMD.
2) We perform specialized optimizations for the time-consuming operators including the customized operators and official TensorFlow operators.

[1]https://bitbucket.org/xiangyumengupc/deepmd-kit-sunway

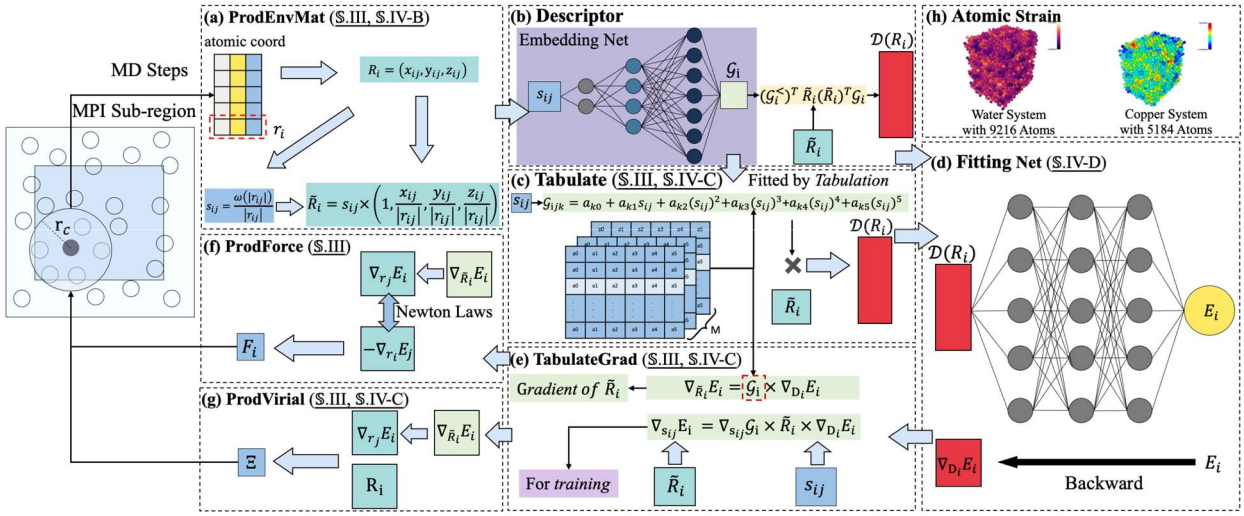


Fig. 1. MD simulation in DeePMD-kit. LAMMPS completes a one-step MD simulation based on these statistics and updates the atomic positions. In one step, the DP model takes atomic coordinates as the input and then predicts the PES, atomic force, and virial. (a) The `ProdEnvMat` operator calculate the environment matrix $\tilde{R}_i$ according to the atomic coordinates. (b) The descriptor network maps s_{ij} and $\tilde{R}_i$ to $\mathcal{D}(R_i)$. It preserves the physical symmetry during the model training. (c) The `Tabulate` operator compresses the descriptor network using Weierstrass approximation. (d) The fitting net that predicts atomic energy E_i. (e) The `TabulateGrad` operator computes the gradient of the descriptor network. (f) The `ProdForce` operator calculates atomic force F_i according to the gradient of r_j (i.e. $\nabla_{r_j} E_i$). (g) The `ProdVirial` operator calculates the virial coefficient Ξ. (h) Perform *ab initio* molecular dynamics and analyze the physical properties of simulation regions.

3) We develop a novel mixed precision scheme for the DeePMD-kit customized operators to fully harness the low precision computing capabilities of the new Sunway while reducing the memory access volume.
4) The re-designed DeePMD-kit achieves a remarkable speedup, reaching 67.6× speedup for the water system with 9216 atoms and 56.5× speedup for the copper system with 5184 atoms on a single new Sunway node. Meanwhile, it has good strong scalability and can scale to 35 million cores for 29 billion atoms with a peak performance of 57.1 PFLOPS, which is 7.9× larger and 1.2× faster than current state-of-the-art tested data.

The rest of this paper is organized as follows. Section II provides a background of the new Sunway supercomputers and the DeepMD-kit package. Section III describes the large-scale parallelization scheme. Section IV illustrates the optimizations for DeePMD-kit operators. Section V presents our innovative mixed precision scheme. Section VI presents the performance evaluation. Section VIII concludes our work.

II. Background

A. DeePMD-kit Package

DeePMD-kit is an open-source NNMD package based on the deep potential (DP) method. It contains several customized TensorFlow operators, including `Tabulate`, `TabulateGrad`, `ProdEnvMat`, `ProdForce`, and `ProdVirial`. DeePMD-kit supports training the DP model and performs inference to derive the energy contribution E_i, atomic force F_i, and virial stress Ξ. The main procedure of the DeePMD-kit, illustrated in Fig. 1, consists of two main components: a descriptor $\mathcal{D}$ and a fitting net $\mathcal{N}$. $\mathcal{D}$ is responsible for acquiring symmetry-preserving characteristics from local environment R_i. For molecular system with atomic coordinates $(r_1, ..., r_N) \in \mathbb{R}^{N\times 3}$, R_i is denoted as $\{r_{ij} : j \in L(i)\}$, where $r_{ij} = r_j - r_i$. $L(i)$ is the neighbor list of the atom i denoted as $\{j : |r_{ij}| \le r_c\}$, where r_c is the predefined cutoff radius. $\mathcal{N}$ learns the high dimensional functional relationship between local environment features obtained from $\mathcal{D}$ and atomic energy contribution E_i. As a result, the PES of the entire system is expressed as the sum of the atomic energy contribution: $E = \sum_i E_i$. Currently, DeePMD-kit is integrated as a force field plugin for LAMMPS [38] to perform large-scale MD simulations.

One key feature of the DeePMD-kit is its physical-invariance descriptor $\mathcal{D}$, as shown in Fig. 1(b). $\mathcal{D}$ is denoted as:

$$\mathcal{D}(R_i) = (\mathcal{G}_i^<)^T \tilde{R}_i (\tilde{R}_i)^T \mathcal{G}_i, \tag{1}$$

where $\tilde{R}_i \in \mathbb{R}^{N_m \times 4}$ denotes the environment matrix calculated by the `ProdEnvMat` operator in Fig. 1(a), and N_m is the predefined max neighbor numbers of the molecular systems. Each row vector of the $\tilde{R}_i$ is formulated as

$$s_{ij} \times \left(1, \frac{x_{ij}}{|r_{ij}|}, \frac{y_{ij}}{|r_{ij}|}, \frac{z_{ij}}{|r_{ij}|}\right), \tag{2}$$

where $s_{ij} = \omega(|r_{ij}|)/|r_{ij}|$ and $\omega(|r_{ij}|)$ is a gating function that decays smoothly from 1 to 0 while $|r_{ij}| < r_c$. The (x_{ij}, y_{ij}, z_{ij}) are the Cartesian coordinates of r_{ij} in R_i. The $\mathcal{G}_i \in \mathbb{R}^{N_m \times M}$ stands for the embedding matrix. The sub-matrix $\mathcal{G}_i^< \in \mathbb{R}^{N_m \times M^<}$, with $M^< < M$, is formed by taking the first $M^<$ columns of $\mathcal{G}_i$. Each row vector $\mathcal{G}_{ij}$ of $\mathcal{G}_i$ is obtained by an embedding network with the input of s_{ij}. The embedding net with m hidden layers is denoted as

$$\mathcal{G}_{ij} = \mathcal{L}_m^e \circ \mathcal{L}_{m-1}^e \circ \cdots \circ \mathcal{L}_0^e(s_{ij}), \tag{3}$$

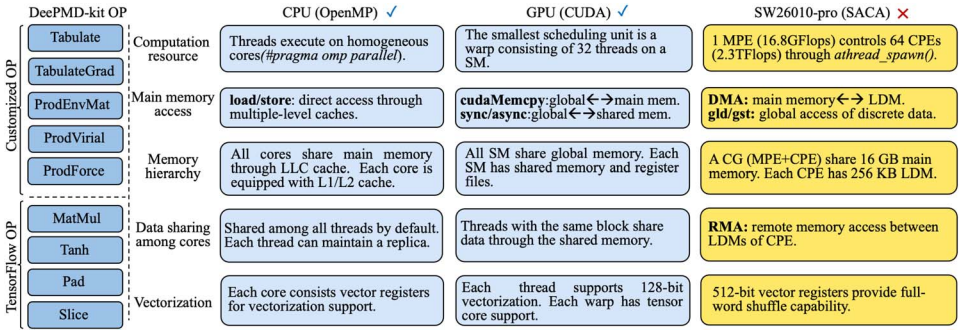


Fig. 2. Architectural difference across CPU, GPU and SW26010-pro (processor of new Sunway supercomputer).

where $\circ$ denotes the function composition, and $\mathcal{L}_0^e$ to $\mathcal{L}_m^e$ represent the fully connected layers. For each layer, the output size is two times of the input size, culminating in a final output size of M.

The fitting network $\mathcal{N}$, as shown in Fig. 1(d), has three fully connected layers that take $\mathcal{D}(R_i)$ as the input and predict atomic energy E_i. Note that `Tanh` is the activation function in both embedding and fitting networks. Combining $\mathcal{D}$ and $\mathcal{N}$, DeePMD-kit can reach *ab initio* accuracy by training with the data generated from scientific computing software such as VASP, Quantum Espresso, and PWmat [11], [12]. The force of atom i is denoted as $F_i = -\nabla_{r_i} E = -\Sigma_j \nabla_{r_i} E_j$, where $\nabla_{r_i} E_j = -\nabla_{r_j} E_i$ in LAMMPS and the $\nabla_{r_j} E_i$ is the gradient of E_i with respect to r_j during the back propagation. This calculation is performed by the `ProdForce` operator (Fig. 1(f)). The virial of the system, denoted as $\Xi = \sum_i R_i F_i$, is calculated through `ProdVirial` operator as shown in Fig. 1(g).

Previously, Guo et al. [31] found that the floating-point operations of the embedding network take more than 95% of the overall FLOPs, and $\mathcal{G}_i$ consume more than 95% of the total memory footprint. They have proposed a *tabulation* method based on fifth-order polynomials to reduce both the FLOPs and memory footprint. Specifically, the *tabulation* calculates the fifth-order polynomial to fit each embedding elements $\mathcal{G}_{ijk}$ in $\mathcal{G}_i$, which is denoted as

$$\mathcal{G}_{ijk} = a_{k0} + a_{k1} s_{ij} + a_{k2} s_{ij}^2 + a_{k3} s_{ij}^3 + a_{k4} s_{ij}^4 + a_{k5} s_{ij}^5, \tag{4}$$

where the a_{k0} to a_{k5} represent the fitted polynomial coefficients. Then, $\mathcal{D}(R_i)$ is

$$\mathcal{D}(R_i) = (\mathcal{G}_i)^T \tilde{R}_i. \tag{5}$$

The *tabulation* is constructed by the `Tabulate` and `TabulateGrad` operators [31] as shown in Fig. 1(c) and 1(e). Note that the *tabulation* maintains high accuracy, for example, the average root mean square error (RMSE) error of the water system with 100 water molecular is $2.0\times\ 10^{-3}$ eV/atom for energy and $3.6\times\ 10^{-2}$ eV/$\mathring{A}$ for atomic force [31].

B. The New Sunway Supercomputer

The new Sunway supercomputer is equipped with new-generation heterogeneous many-core processors (i.e., SW26010-pro). Each computing nodes are interconnected through a fat-tree topology, with every 256 nodes in a rack forming a supernode [39]. Each node is equipped with an SW26010-pro processor, which contains 6 core groups (CGs) offering a theoretical peak performance of around 14 TFLOPS in double precision. Each CG contains a management pressing element (MPE) and an 8×8 computing processing element (CPE) grid connected to 16 GB DDR4 main memory with 51.2 GB/s bandwidth. Each CPE has a 256KB manually controlled scratchpad memory called LDM. MPE and CPE have 256-bit and 512-bit SIMD units, respectively. The SW26010-pro processor supports 307GB/s direct memory access (DMA) to transfer contiguous data between LDM and main memory. In addition, the data transfer among CPEs within the same CG has around 400GB/s bandwidth through remote memory access (RMA). The programming interface of the new Sunway is *SACA*, which provides a basic compilation environment, basic libraries, and runtime.

C. Challenges and Motivations

Although previous works have successfully optimized the DeePMD-kit on both ARM and NVIDIA GPU platforms, migrating DeePMD-kit to the new Sunway supercomputer still faces several challenges.

SW26010-pro processor has a unique architecture compared with mainstream CPUs and GPUs. Unlike typical multi-core CPUs that execute instructions on homogeneous cores simultaneously and GPUs that use thread blocks to organize parallel threads, SW26010-pro employs an MPE-CPE collaborative scheme, where one MPE controls 64 CPEs. Therefore, the task partition and assignment should be carefully determined by programmers. Instead of utilizing multi-level caches like CPUs to optimize the main memory access, SW26010-pro features a programmable fast LDM for each CPE, which is similar to a manually controlled L1 cache. Therefore, it enables a more flexible caching strategy and configurable memory access. Moreover, unlike GPUs that use shared memory for thread data sharing and CPUs that rely on caches for inter-thread communication, SW26010-pro does not provide physically shared on-chip memory for CPEs. Instead, data exchange between different CPEs is managed through the RMA mechanism. The uniqueness, as illustrated in Fig. 2, enables a unique

OP	Time(s)	FLOPS	BW(GB/s)
MatMul	5.36 (67.0%)	2.07G	0.26
Pad	0.99 (12.4%)	–	0.54
Tabulate	0.52 (6.5%)	6.42G	4.00
TabulateGrad	0.48 (6.0%)	5.28G	1.05
ProdEnvMat	0.24 (3.0%)	2.46G	2.67
Tanh/Grad	0.09 (1.1%)	1.82G	6.60
ProdVirial	0.06 (0.7%)	2.41G	6.68
ProdForce	0.05 (0.6%)	2.45G	8.13
Add	0.04 (0.4%)	0.54G	6.78
Slice	0.03 (0.4%)	–	7.4

Fig. 3. Time-consuming operators of DeePMD-kit on a SW26010-pro.

programming framework. We should leverage these features of SW26010-pro architecture for superior performance.

Currently the resource utilization of the DeePMD-kit operator is quite low. Unlike the implementations using OpenMP and CUDA, the design of MPE and CPE codes requires careful consideration. The MPE launches the *athread_spawn()* function to invoke 64 CPEs for computation, with each CPE executing computations and memory accesses simultaneously. However, previous work did not optimize DeePMD-kit operators for SW26010-pro architecture, resulting in all operators being executed only on the MPEs. Furthermore, some new features in CPEs have not been effectively utilized, including memory access through DMA, on-chip data sharing through RMA, and 512-bit vectorization. The performance analysis of time-consuming DeePMD-kit operators, shown in Fig. 3, indicates that the achieved peak performance and memory bandwidth of each operator are significantly lower than the theoretical peak performance (14 TFLOPS) and peak memory bandwidth (300 GB/s). Therefore, it is crucial to optimize each DeePMD-kit operator for the new Sunway architecture to ensure high resource utilization and computing efficiency.

III. Large-Scale Parallelization Scheme on New Sunway

We adopt the MPI+SACA+SIMD three-level parallelization scheme to exploit the parallel computing power of the new Sunway in Fig. 4. We use MPI to implement data-parallel inference. For each DeePMD-kit operator, we design a heterogeneous SACA kernel, leveraging the MPE to schedule 64 CPEs for simultaneous computation. Finally, we use the SIMD intrinsic of CPE to achieve 512-bit vectorization.

A. MPI Spatial Partition

Fig. 4(a) shows the MPI partition schemes on the new Sunway. Large molecular systems are evenly divided into multiple sub-regions based on the number of active MPI ranks. Each MPI rank independently invokes a DeePMD-kit graph to perform inference on its assigned sub-region. The SW26010-pro processor contains 6 NUMA CGs. Therefore, we launch 6 MPI ranks on a single SW26010-pro processor, with each CG holding an MPI sub-region. The reasons for this configuration are: 1) The SACA model does not support launching more than one MPI rank per CG. 2) Even though it supports launching multiple MPI ranks per CG, some vendor-optimized kernel libraries designed for full CPEs can hardly be utilized.

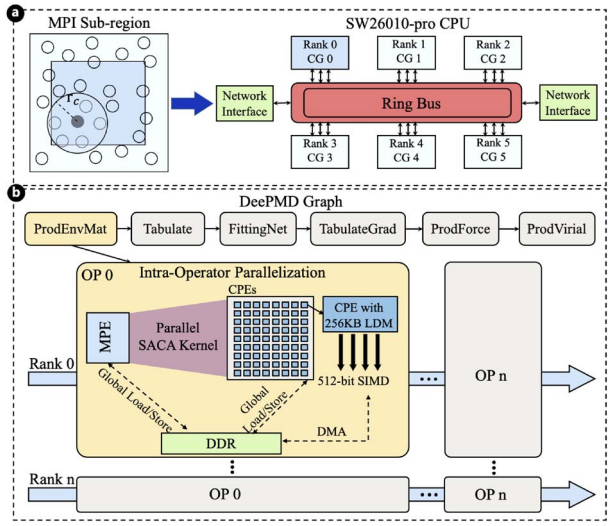


Fig. 4. Parallelization schemes of DeePMD-kit. (a) Spatial partition of the MD system associated with the CGs. (b) MPI+SACA+SIMD parallelization on single SW26010-pro node of new Sunway.

B. MPI+SACA+SIMD Parallelization

As shown in Fig. 4(b), we combine the architectural features of SW26010-pro to establish the MPI+SACA+SIMD parallelization scheme. The first level is the process parallel scheme based on MPI. We launch 6 MPI ranks on a SW26010-pro node, with each CG holding an MPI rank that computes a sub-region. The second level is the intra-operator parallel scheme based on SACA. We leverage SACA to accelerate the time-consuming operators, including `Tabulate`, `TabulateGrad`, `ProdEnvMat`, `ProdForce`, `ProdVirial`, `Slice`, and `Pad`. Specifically, their computation-intensive parts are assigned to 64 CPEs according to the atom *id*. Each CPE computes a tiled block that has been uniformly divided according to the operator input. After the calculation, MPE executes the athread_join() instruction to synchronize the 64 CPEs and store the result in the main memory. Finally, the DMA strategy is utilized to achieve efficient continuous data transfer between LDM and the main memory. Each DeepMD-kit operator has a demand for large LDM space, so we cancel the *cache configuration* to save LDM space. Except for program stack, temporary variables, and program pointers, the remaining LDM space is all used as DMA buffers. The third level is SIMD. For each CPE, we leverage 512-bit vectorization instructions to further improve the parallel efficiency of the DeePMD-kit operator.

IV. Optimizing Critical Operators

DeePMD-kit relies on several computation-intensive TensorFlow customized operators and official operators. The customized operators include `Tabulate`, `TabulateGrad`, `ProdEnvMat`, `ProdForce`, and `ProdVirial`. The official

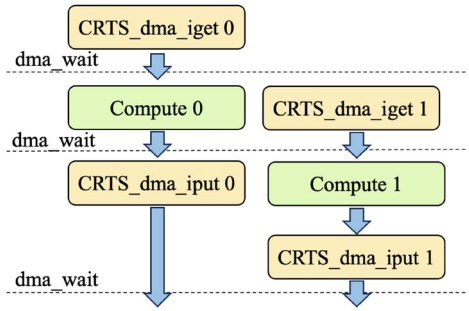


Fig. 5. Overlapping between computation and memory access on the new Sunway.

operators include `MatMul`, `Tanh`, `TanhGrad`, `Slice`, and `Pad`. In this section, we provide detailed performance optimization for these operators to maximize the resource utilization of the new Sunway.

A. Accelerate the Tabulation

The *tabulation* method [31] compresses the embedding net $\mathcal{G}_i$ using the fifth-order polynomial approximation, saving 82% of the total FLOPs. This method is implemented by `Tabulate` and `TabulateGrad` operators in DeePMD-kit. However, the current implementation of the *tabulation* on the new Sunway is time-consuming. Profiling the `Tabulate` and `TabulateGrad` operators reveals ineffective memory access (i.e., memory bandwidth less than 1%) when loading polynomial coefficients, which accounts for 68.6% and 54.6% of the execution time, respectively. Therefore, we further accelerate these operators to maximize the computing efficiency and bandwidth utilization with the following optimizations.

1) Computation and Memory Access Overlapping: We adopt an overlapping strategy of computation and memory access for the `Tabulate` and the `TabulateGrad` operators, whose core computation is a three-level nested loop. The new Sunway supports high-bandwidth asynchronous DMA, enabling efficient access to contiguous data blocks in main memory. Each CPE can invoke an asynchronous memory access request using CRTS_dma_iget() or CRTS_dma_iput() instructions. After initiating the DMA request, the CPE executes the synchronizes to wait for the memory access and checks the DMA status by executing the dma_wait() instruction. Computation is not blocked before executing the dma_wait() instruction, allowing unrelated computational tasks before dma_wait() to overlap computation and DMA.

As shown in Fig. 5, the asynchronous DMA transfer between LDM and main memory is manually controlled to maximize the overlapping using dma_wait(). Each CPE is responsible for executing several description vectors ($\mathcal{D}(\tilde{R}_i)$) to fully utilize the LDM space (e.g., 32 in the water system and 8 in the copper system, using 223KB LDM). We divide the input s_{ij} and the corresponding table coefficients into two buffers. Each buffer independently processes a part of $\mathcal{D}(\tilde{R}_i)$ in CPEs including calculation and the DMA access between LDM and the main memory. When the first buffer completes its DMA transfer from the main memory to the local memory and begins computation, the second buffer can launch the DMA request to load s_{ij} and table coefficients from the main memory. Once the second buffer completes the DMA request and starts computation, the first buffer requests to write $\mathcal{D}(\tilde{R}_i)$ back to the main memory. Finally, we use dma_wait() to synchronize the computations of both buffers. Through this overlapping strategy, these operators can maximize the utilization of the 300GB/s DMA bandwidth and increase the throughput.

2) Kernel Fusion: As shown in Fig. 1(c) and 1(e), both `Tabulate` and `TabulateGrad` operators calculate the matrix $\mathcal{G}_i$ described in Equation 5. However, calculating $\mathcal{G}_i$ involves frequent memory access for inputs s_{ij} and polynomial coefficients. We design the kernel fusion scheme to remove the redundant calculation. In this approach, the `Tabulate` saves the intermediate $\mathcal{G}_i$, and then `TabulateGrad` can directly load $\mathcal{G}_i$ for subsequent calculations. Compared to the original `TabulateGrad` design, the kernel fusion can eliminate the redundant calculation and the memory access for s_{ij} and polynomial coefficients by directly loading $\mathcal{G}_i$.

B. Data Layout Conversion and Branch Elimination

DeePMD-kit is implemented as a force field plugin for LAMMPS. Some variables in `ProdEnvMat` are stored as the array-of-structures (AoS) layout to ensure compatibility with LAMMPS and the backpropagation process in the MD simulation, which hinders the vectorization. Furthermore, the `ProdEnvMat` includes many conditional branches, causing branch divergence and reducing the computational efficiency of SIMD. Therefore, we perform data layout conversion and branch elimination for `ProdEnvMat` operator to exploit the 512-bit vectorization.

1) Data Layout Conversion: The DeePMD-kit variables rij_a, $descrpt\_a$, and $descrpt\_a\_deriv$ are stored as the AoS layout in memory, which stores each member in non-contiguous memory. For example, the float64 atomic coordinate rij_a is stored as $\{(x_0, y_0, z_0), (x_1, y_1, z_1), ...\}$, where each structure contains three coordinate members. However, the memory spacing of the same type of member (e.g., all $\boldsymbol{x}$ values) from adjacent structures is 3, which hinders optimal utilization of vectorized load instructions. In contrast, the structure-of-array (SoA) layout separates each member into contiguous arrays to exploit the vectorization. For example, the SoA format of rij_a is $\{(x_0, x_1, x_2, x_3, \ldots), (y_0, y_1, y_2, y_3, \ldots), (z_0, z_1, z_2, z_3, \ldots)\}$. Using three 512-bit SIMD load instructions can efficiently load each contiguous float64 member into three vector registers $x = (x_0, x_1, x_2, x_3)$, $y = (y_0, y_1, y_2, y_3)$, $z = (z_0, z_1, z_2, z_3)$, simultaneously. Then, these vectors can be processed in parallel for relevant calculation leveraging vectorization.

The new Sunway supports the full-word vector shuffle instruction `VSHFW`, which enables efficient value shuffling. In `VSHFW`, each 512-bit vector is divided into 16 32-bit **words**. The shuffle mask consists of 16 5-bit components, specifying the word location in the source vectors to be placed in the destination vector. By performing `VSHFW`, the 16 words of the

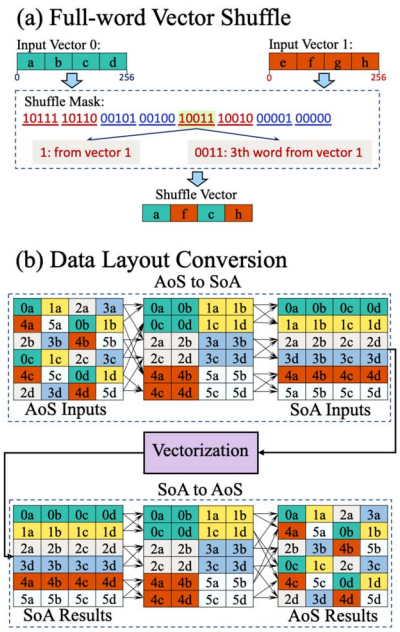


Fig. 6. Conversion between AoS and SoA. This figure illustrates the 256-bit VSHFW instruction and 6×4 array conversion using 256-bit VSHFW.

destination vector are assigned according to the mask values. Fig. 6(a) shows an example of 256-bit VSHFW instruction, which shuffles two source 256-bit vectors based on the defined mask. The mask is divided into eight 5-bit components, where the fifth bit denotes which source vector to select from, and the remaining four bits denote which word in the chosen source vector to select. As shown in Fig. 6(a), the fourth component of the mask is 0x13, indicating that the corresponding word of the destination vector is fetched from the third word of source vector 1. Therefore, we design the efficient conversion between AoS and the SoA using the VSHFW SIMD instruction. This approach first transforms the AoS variables to SoA using VSHFW for 512-bit vectorization. Then, it reverts the calculated SoA results to SoA format to store them in the memory after the vectorization. Fig. 6(b) illustrates the 256-bit conversion examples for AoS variables with a memory interval of 6. Using the above conversion in Fig. 6(b), only 24 additional VSHFW instructions (i.e., around 24 cycles)can increase the overall 4 × throughput.

2) Branch Elimination: ProdEnvMat operator uses conditional branch to remove the redundant calculation [31]. For each atom, ProdEnvMat retrieves the global atom index of the neighbors from the neighbor list $L(i)$. If the global atom index is less than 0 (i.e. all neighbors of the current atom are retrieved), ProdEnvMat skips the current loop and retrieves the neighbors from the next atom in $L(i)$. Each loop executes the branch instruction to check the global atom index and causes branch divergence. Then, we create the branch elimination strategy to remove the branch divergence. Specifically, we calculate the condition mask for the SIMD vector, and the branch output is derived by multiplying the SIMD vector with the mask.

C. Write Conflict Avoidance

Write conflicts arise in ProdVirial operator when using the intra-operator parallel scheme on the new Sunway to calculate the global virial stress, which is typically mitigated using atomic operations. The atomic operations on the new Sunway are implemented through the fetch-and-add mechanism, with each atomic operation requiring a global synchronization across all CPEs of a CG. However, calculating the global virial stress with nine elements introduces $9 \times N_{des} \times N_{loc} \times N_{nnei}$ synchronization using atomic operations on new Sunway, where N_{des} denotes the descriptor number, N_{nloc} denotes the atom number, and the N_{nnei} denotes the neighbor atom number. The frequent synchronizations deteriorate the parallel efficiency, resulting in inferior performance compared to the serial implementation on the MPE. To address this issue, we introduce a replication-and-reduction strategy. Fig. 7(a) illustrates the replication strategy for the mutex variable $virial$.

We create a replica of $virial$ called $virial\_rep$ for each CPE and set their elements to 0 as shown in line 1 of the algorithm. The global virial only contains nine elements, so the $virial\_rep$ can be stored in the LDM, enabling efficient memory access and data reuse. Then each CPE calculates its increment on virial called tmp_v and adds to the $virial\_rep$ (line 2-8). After each CPE calculates $virial\_rep$, the CRTS_ssync_array() instruction is executed to synchronize the 64 CPEs, ensuring the completeness of the code before this instruction. Meanwhile, the efficient RMA reduction in Fig. 7(b) is implemented to reduce the replication on each CPE. According to the architecture of the CPE array, each *cluster* contains 2 × 2 CPEs and a slave cluster management (SCM) unit (consisting of an RMA engine and a DMA engine). The 16 SCMs in a CPE array are interconnected through the on-chip network. Therefore, RMA operations can be quickly performed within a cluster, while communication latency increases as the distance between CPEs increases. As shown in Fig. 7(b), we propose an RMA-based reduction that contains 3 steps. The first step is the reduction within the cluster. Through two rounds of RMA operations, each chief CPE (i.e., the first CPE) within a cluster aggregates the partial values. The second step is the row reduction of each cluster. By executing RMA reductions on clusters sharing the same row number within the CPE array, the chief CPEs in the first column of the CPE array derive the partial reduction in a row. Finally, the third step is column reduction. The chief CPEs in the first column perform the partial reduction through RMA and the chief CPE of the first cluster (i.e., CPE 0) derives the final reduction result that then is written back to the main memory through the DMA. Compared with the lock mechanism, the replication-and-reduction method avoids redundant global synchronizations by utilizing the RMA communication of the CPE array.

D. Tailoring Deep Learning Framework for DeePMD-kit

TensorFlow lacks optimization for the new Sunway architecture, and some official TensorFlow operators are not efficient enough in DeePMD-kit. Therefore, we further optimize these

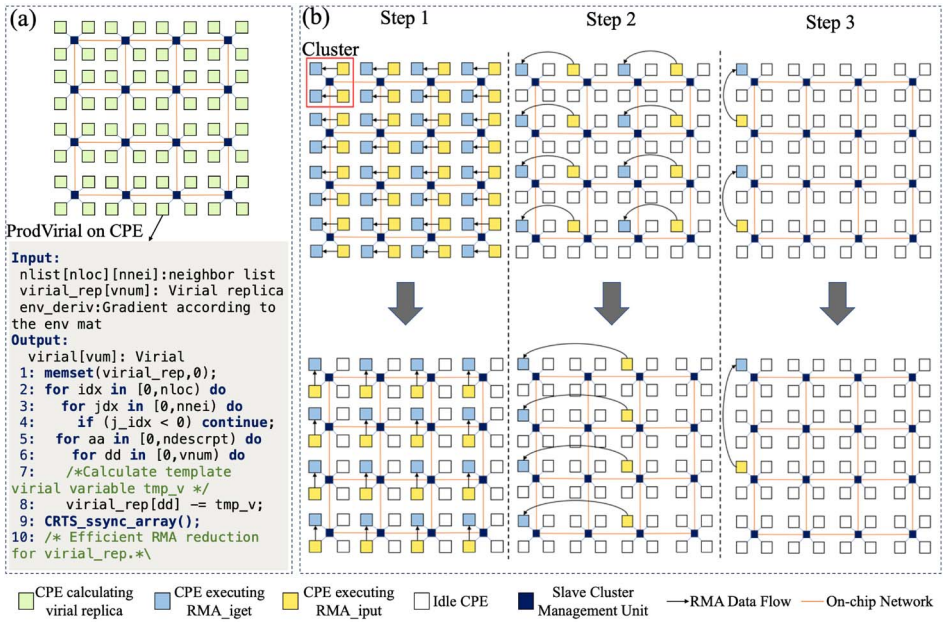


Fig. 7. Replication-and-reduction strategy on `ProdVirial` calculation. (a) Replication strategy for the mutex variable `virial`. (b) Efficient reduction among CPE arrays using RMA, which contains three steps: 1) cluster reduction, 2) row reduction, and 3) column reduction.

operators on the new Sunway to fully exploit the computing power of this architecture.

1) Fitting Net Optimization: The fitting net consists of multiple fully connected layers and `Tanh` activation functions. Additionally, it needs to calculate the corresponding gradients used for the force calculation during backpropagation. However, the current TensorFlow implementation is not efficient and cannot exploit the computational power of the new Sunway. The original fully connected layer of DeePMD-kit, denoted as $\mathcal{L}_k^f(x) = x \cdot W_k^f + b_k^f$, is implemented using two operators, including a `MatMul` operator and an `Add` operator, introducing redundant data movement of intermediate tensors and extra kernel launches. Therefore, we fuse `MatMul` and `Add` in Eigen into the GEMM (calculating $A \times B + C$ in one kernel) of the vendor-optimized swBLAS on the new Sunway. Meanwhile, we accelerate the `Tanh` and the `TanhGrad` operators using SACA to exploit the computing power of the CPEs in the SW26010-pro processor.

2) Parallelize the Tensor Manipulation Operators: Some specialized tensor manipulation operators, such as `Pad` and `Slice`, lack support for parallel execution on CPEs and do not utilize the 300 GB/s DMA bandwidth effectively. As a result, the computational performance of the DeePMD-kit is significantly impacted. To tackle these challenges, we parallelize these operators use SACA, and exploit DMA instructions to maximize memory bandwidth utilization.

V. Mixed Precision Scheme

As shown in Fig. 1(c), for each smooth gate value s_{ij}, the `Tabulate` operator searches for the corresponding table weight $w_{ij} \in \mathbb{R}^{M \times 6}$ to calculate the embedding vector $\mathcal{G}_{ij} \in \mathbb{R}^M$, where M denotes the feature dimensions of the embedding matrix. However, each s_{ij} requires to access $M \times 6 \times 8$ bytes

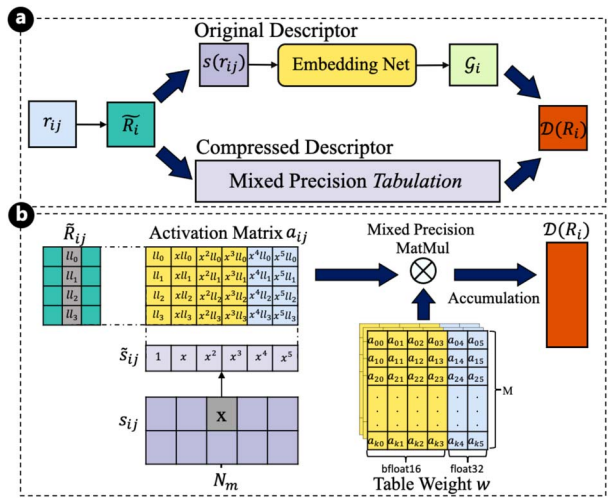


Fig. 8. Mixed precision *tabulation* scheme. (a) Framework of the mixed precision *tabulation* method. This method directly uses the table weight mixed by bfloat16 and float32 to compress the descriptor model of the DeePMD-kit. (b) The optimized computation workflow of the `Tabulate` operator.

w_{ij}, which consumes massive memory access and results in low computational efficiency during large-scale MD simulations. To alleviate the bandwidth bottleneck of the `Tabulate` operator, we propose a mixed precision *tabulation* method. Fig. 8(a) shows the specific mixed precision method. Note that we have removed the corresponding operation through the kernel fusion method in `TabulateGrad` operator, so we only need to optimize the `Tabulate` operator.

Unlike the low precision truncation method, the proposed mixed precision method uses low precision table weight to directly fit the DeePMD-kit descriptor from scratch, which can

minimize the compression error. Through statistical analysis of each item in the tabulation weights, we find that the higher-order coefficients (i.e. a_4 and a_5) exhibit a substantial value range during the training stage, which shows that these items are more sensitive to precision change. For example, the absolute maximum values of a_4 and a_5 reach 8.5×10^{10} and 1.3×10^{11}, respectively on water system. Since the values of higher-order coefficients are significantly greater than the representative range of float16, directly using float16 to compress the weight would cause the overflow issue. Even using bfloat16 to compress the high order weights including a_4 and a_5 would cause significant errors. Therefore, we use bfloat16 to create the low-order weights (i.e. a_0, a_1, a_2, and a_3) and use float32 to create the high-order weights (i.e. a_4 and a_5). Then we fit the low precision weights using the proposed mixed precision *tabulation* method. The reason for using bfloat16 to create the low-order weights instead of float16 is that the value range of bfloat16 is consistent with that of float32, which can prevent overflow issues during the training process. The new Sunway supercomputer supports low precision floating point calculation and 512-bit SIMD support including float16, bfloat16, and float32. Therefore, we can design the mixed precision `Tabulate` kernel using SACA and vectorization intrinsic to exploit the efficient low precision computing power of CPEs. As a result, `Tabulate` can save approximately 3 $\times$ the memory footprint and increase throughput by around 23 $\times$.

Meanwhile, we optimize the calculation workflow of the original `Tabulate` operator. As shown in Fig. 1(c), the original `Tabulate` requires M iterations to load the polynomial coefficients in w_{ij} and compute the embedding values $\mathcal{G}_{ijk}$ for each $x = s_{ij}$. However, each iteration involves redundant calculations of x^2, x^3, x^4, and x^5 to obtain the polynomial. We then devise an optimized `Tabulate` workflow. As shown in Fig. 8(b), we first calculate the $\tilde{s}_{ij} = (1, x, x^2, x^3, x^4, x^5)$ before M polynomials calculation and perform a vector-vector multiplication with row vector of $\tilde{R}_i$ to obtain the activation matrix $a_{ij} \in \mathbb{R}^{4\times 6}$. Then, the a_{ij} performs mixed precision matrices multiplication with w_{ij} and accumulates along the N_m dim to derive the $\mathcal{D}(R_i)$. Compared with the original workflow, the proposed method calculates a_{ij} before the time-consuming loop for calculating $\mathcal{D}(R_i)$, reducing $M \times 4$ redundant calculations.

VI. Evaluation

In this section, we choose two typical physical systems, water and copper, as benchmarks for performance evaluation. Water is a challenging system even for AIMD methods, which must balance between weak non-covalent intermolecular interactions, thermal (entropic) effects, as well as nuclear quantum effects [40]. Copper is a representative metal where properties such as surface formation energy and stacking fault energies are challenging to accurately predict using empirical force fields. The training data of the water and copper are described in [41]. To simulate and analyze the performance of our work, we use the Tabulate-based DP model [31] as the baseline,

TABLE II
Errors of Energy and Force Comparison for the Purposed Mixed Precision Scheme and the Baseline Method

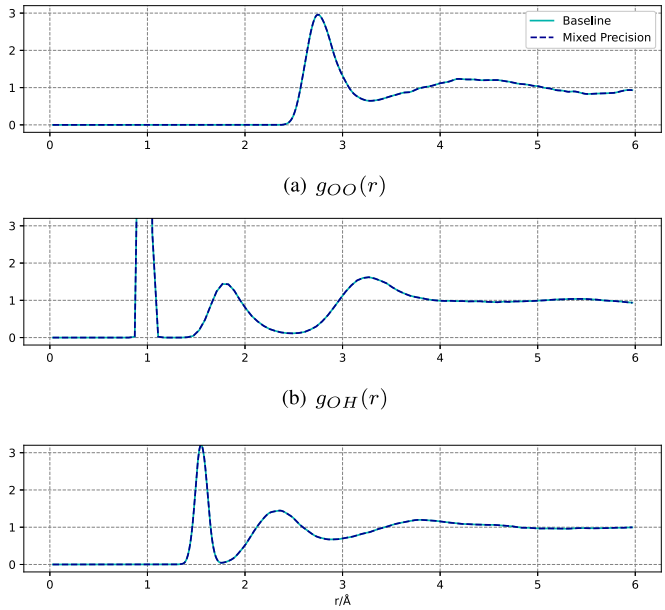
Schemes	Energy Error [eV/Molecule]	Force Error [eV/Å]
Baseline [31]	2.0×10^{-03}	3.6×10^{-02}
Mix-bf16	4.4×10^{-03}	3.9×10^{-02}

where the descriptor is a compressed `Tabulate` module with the output dimension of 128. The fitting network dimension is $224 \times 224 \times 224$. The cutoff radius settings are 6 $\mathring{A}$ and 8 $\mathring{A}$, respectively, with the maximal neighbor numbers of 138 and 512. The MD equations are numerically integrated by the Velocity-Verlet scheme for 100 steps at time-steps of 0.5 fs and 1.0 fs, respectively. The velocities of the atoms are randomly initialized according to the Boltzmann distribution at 330 K. The neighbor list with a 2 $\mathring{A}$ buffer region is updated every 50 time steps. The thermodynamic data including the kinetic energy, potential energy, temperature, and pressure are collected and recorded in every 20 time steps.

A. Accuracy of Mixed Precision Scheme

1) Energy and Force Error: The accuracy of the DeePMDkit is investigated by comparing the force and energy error with the AIMD method. Specifically, we use the test dataset calculated by the VASP as an example to evaluate the precision, which is composed of 100 water molecules. First, we use the mixed precision model to predict the energy and forces of the 100 water molecules, and then calculate the root mean square error (RMSE) of the predicted energy and forces against the AIMD results. Table II shows the RMSE errors of our mixed precision scheme compared to the baseline tabulation scheme in [31]. Compared with the baseline, the proposed mixed precision approach achieves an energy error of 2.43×10^{-3} eV/molecule and a force error of 2.9×10^{-3} eV/$\mathring{A}$. Compared to the AIMD method, the energy error predicted by the mixed precision scheme is much smaller than 4×10^{-2} eV/molecule, and the force error is no less than 4×10^{-2} eV/$\mathring{A}$ [35], denoting the proposed mixed precision method demonstrates very low error during the MD simulation.

2) Radial Distribution Function: To further evaluate the accuracy of mixed precision, we calculate the radial distribution function (RDF) of water, which is the normalized probability of searching a neighboring atom at the spherically averaged distance r. The RDFs of oxygen-oxygen ($g_{OO}(r)$), oxygen-hydrogen ($g_{OH}(r)$), and hydrogen-hydrogen ($g_{HH}(r)$) are commonly employed to characterize the structural properties of water. Specifically, we choose 192 water atoms to perform 20,000 MD steps using the proposed mixed precision DeePMD-kit and the baseline in [35]. We then calculate the RDFs of the water system based on the simulation results and plot the RDF curves in Fig. 9. According to Fig. 9, the RDFs of the proposed mixed precision *tabulation* scheme and the baseline method are perfectly overlapped. The baseline method has been demonstrated to achieve accuracy comparable to AIMD methods in prior works [35]. Therefore, we demonstrate that



(a) $g_{OO}(r)$

(b) $g_{OH}(r)$

(c) $g_{HH}(r)$

Fig. 9. Radial distribution functions $g_{OO}(r)$, $g_{OH}(r)$, and $g_{HH}(r)$ of liquid water at ambient conditions, calculated by two DeePMD-kit implementations: baseline with the precision of float64, mixed precision using bfloat16 and float32.

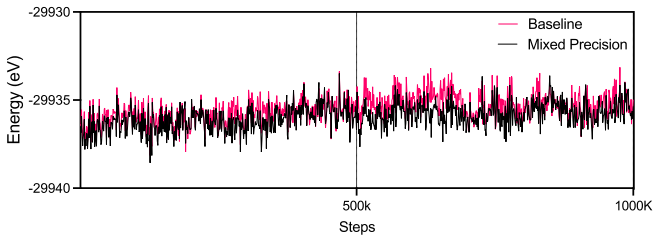


Fig. 10. System energy curves along 1,000,000 steps of the water simulation. The mixed precision scheme can perform a stable simulation during long timesteps with a low error rate.

the proposed mixed precision method achieves AIMD-level accuracy while effectively simulating and predicting physical observables.

3) Stability Evaluation: To evaluate the MD stability of the mixed precision scheme. We choose the water system with 192 atoms to perform a long MD simulation for the proposed mixed precision scheme. We use the same configurations to perform one million steps long NVE simulations for the proposed mixed precision DeePMD-kit and the baseline method proposed in [35]. Then we observe the energy fluctuations of the system during the MD simulation and plot the energy curves calculated by these two DeePMD-kit schemes in Fig. 10. The baseline is the result of a standard DeePMD-kit with the double precision *tabulation*. The mixed precision result is the MD simulation result using the DeePMD-kit after the mixed precision *tabulation*. According to the results in Fig. 10, the mixed precision implementation follows the same trend but has certain deviations compared to the baseline. This deviation is primarily attributed to lossy compression and the computation order modification, which lead to the error accumulation. However, the simulation results are stable enough for long time simulation.

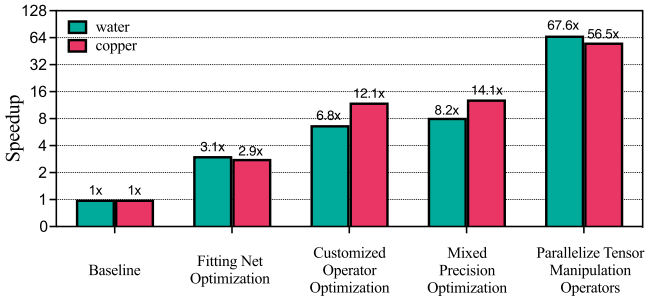


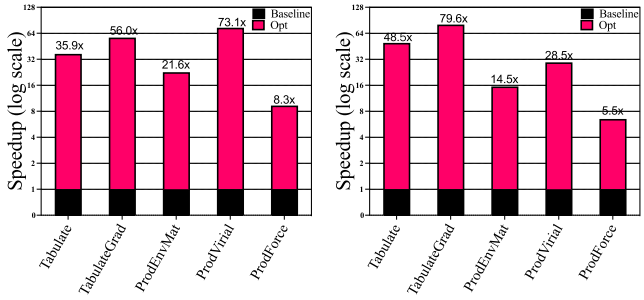
Fig. 11. Breakdown performance improvement on the single SW26010-pro node for 9,126 atoms water and 5,184 atoms copper systems. The performance speedup is normalized to the baseline.

B. Single Node Breakdown Analysis

As illustrated in Section III, IV and V, our work on the new Sunway supercomputer includes optimizations on customized operators (i.e. `Tabulate`, `TabulateGrad`, `ProdEnvMat`, `ProdVirial`, and `ProdForce`), fitting net, mixed precision scheme, and tensor manipulation operators. Through detailed performance profiling, the top-down order of time-consuming parts is *1)* fitting net (occupying 68.1% of the total time), *2)* customized operators (occupying 16.8%), and *3)* tensor manipulation operators (occupying 13.9%). We perform the breakdown analysis following this order and report the corresponding performance improvements on a single new Sunway node (one SW26010-pro processor). We perform 100 steps of MD simulation and measure the corresponding **loop time** (reported by LAMMPS) for the water system configured with 9,126 atoms and the copper system configured with 5,184 atoms. As for the baseline, we use the flat MPI version of the DeePMD-kit with *tabulation* proposed in [31]. We perform MD simulations with the same configurations and calculate the corresponding speedup for each optimization. Fig. 11 shows the breakdown analysis results of the DeePMD-kit on a single new Sunway node.

1) Effectiveness of the Fitting Net Optimization: Compared to the baseline, the optimized DeePMD-kit with fitting net optimization achieves 3.1× speedup for the water system and 2.9× speedup for the copper system. Each fully connected layer of the fitting net consists of `MatMul`, `Add`, `Tanh`, and `TanhGrad` operators. These results indicate that the proposed kernel fusion and the `Tanh` optimization methods can effectively enhance the computational performance of DeePMD-kit.

2) Effectiveness of the Customized Operator Optimization: By carrying out the customized operator optimization, the optimized DeePMD-kit achieves 6.8× speedup on the water system and 12.1× on the copper system, which denotes the proposed optimization including tabulation optimization, data layout optimization, branch optimization, and write conflict avoidance can significantly improve the performance of DeePMD-kit.



(a) Water System (b) Copper System

Fig. 12. Performance evaluation for customized operators in DeePMD-kit. We evaluate the performance improvements for customized operators on 9,216 atoms water system and the 5,184 copper system, respectively.

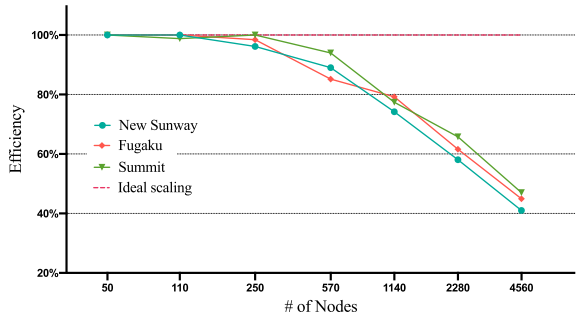


Fig. 13. Strong scalability evaluations of 100 MD steps for water systems. We evaluate the efficiency of 14,910,336 atoms simulation on new Sunway, and compare the scalability of 8,294,400 atoms on Fugaku and 41,472,000 atoms on Summit [31].

3) Effectiveness of the Mixed Precision: After the mixed precision optimization on *tabulation*, our work achieves 8.2× on water system and 14.1× on copper system, which denotes that using the proposed mixed precision scheme can further enhance the computing efficiency of CPEs by maximizing the bandwidth utilization of the DMA.

4) Effectiveness of the Manipulation Operators Parallelization: Finally, after optimizing the tensor manipulation operators, our work can achieve a significant 67.6× speedup for the water system and 56.5× for the copper system. This interesting phenomenon denotes that the pre-installed TensorFlow library lacks sufficient optimization for some tensor manipulation operators and severely affects the performance of DeePMD-kit. After the parallelization, these operators can significantly enhance the computing efficiency thus accelerate the overall MD simulation.

C. Performance Evaluation of Customized Operators

In *customized operator optimization*, we have implemented many contributions including *acceleration the tabulation* for `Tabulate` and the `TabulateGrad`, *data layout conversion and branch elimination* for `ProdEnvMat`, and the *write conflict avoidance* for `ProdVirial`. Some operators have low computing occupancy, demonstrating limited overall performance improvement. We further evaluate the performance improvements of each customized operator on water and copper systems to provide a more detailed breakdown analysis. Specifically, we select the DeePMD-kit model used in the *customized operator evaluation* in Fig. 11 to conduct a single step of MD simulation. The elapsed time of the customized operators is measured for a water system with 9,216 atoms and a copper system with 5,184 atoms. For comparison, we apply the same configurations to measure the elapsed time of the baseline model and calculate the corresponding speedup. Fig. 12 presents the speedup of each customized operator relative to the baseline.

Compared to the baseline, the `Tabulate` and the `TabulateGrad` operators have achieved 35.9/48.5 × and 56.0/79.6 × speedup on water/copper system, denoting the *accelerate the tabulation* design can effectively enhance the performance of Tabulate and TabulateGrad operators. The `ProdEnvMat` operator has achieved 21.6/14.5 times speedup on water/copper system, denoting the *data layout conversion and branch elimination design* can effectively enhance the performance `ProdEnvMat` operator. The `ProdVirial` operator has achieved 73.1/28.5 × speedup on water/copper system, denoting the proposed *write conflicts avoidance* design can significantly enhance the performance of `ProdVirial` operator. Finally, the `ProdForce` operator has achieved 8.3/5.5 × speedup on water/copper system, denoting the `ProdForce` operator also achieves good performance after the optimization.

Among these operators, the `ProdEnvMat` and `ProdForce` operators exhibit relatively lower performance compared to the other customized operators, which is primarily due to the non-contiguous memory accesses within these operators so that we have to resort to using *gld* and *gst* instructions to perform global memory access with a theoretical bandwidth of 51.2 GB/s. Through further bandwidth evaluation, we find the achieved memory bandwidth of `ProdEnvMat` and `ProdForce` operators are only 39.7 / 41.8 GB/s and 49.2 / 45.6 GB/s on water/copper systems. However, these two operators are not the bottleneck, consuming less than 10% of the total execution time, which has a negligible impact on the overall performance of the DeePMD-kit.

D. Scalability

1) Strong Scalability: We evaluate the strong scalability of the optimized DeePMD-kit on the new Sunway. To compare with existing DeePMD-kit implementations on Summit [31], [42] and Fugaku [31], we adopt the same configuration of the node numbers, and the experiments begin with 20 nodes and gradually expand to 4,560 nodes on new Sunway. Specifically, we select a water system with 14,910,336 atoms and a copper system with 2,540,160 atoms as the benchmarks on the new Sunway to saturate the memory capacity (96 GB) of each node. Figs. 13 and 14 illustrate the parallel efficiency of the water system and the copper system, which is normalized to the efficiency with 50 nodes and 20 nodes respectively. For the water/copper system, the parallel efficiency reaches 41.2% / 23.1% on 4,560 nodes of the new Sunway, respectively. We find that all machines demonstrate good scalability with up to 570 nodes, achieving a parallel efficiency larger than 60%. Furthermore, all machines show the potential to scale to 4,560 nodes.

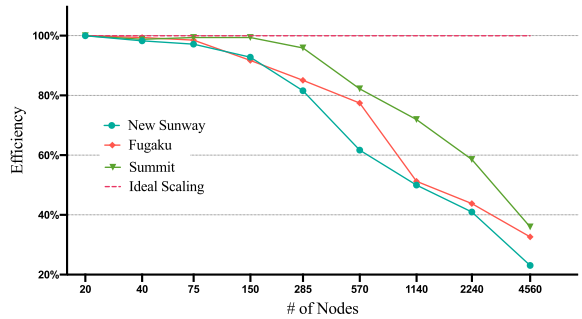


Fig. 14. Strong scalability evaluations of 100 MD steps for copper systems. We evaluate the efficiency of 2,540,160 atoms simulation on new Sunway, and compare the scalability of 2,177,280 atoms on Fugaku and 13,500,000 atoms on Summit [31].

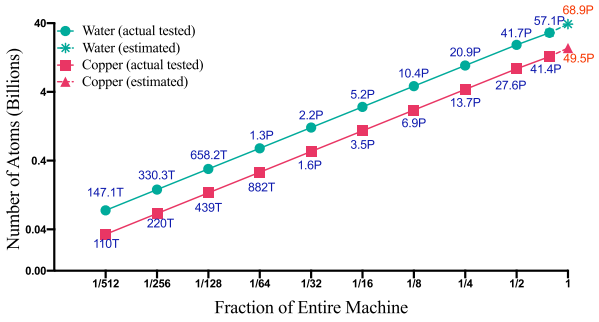


Fig. 15. Weak scalability of water and copper on new Sunway. The re-designed DeePMD-kit can achieve 35 billion atoms simulation with a peak performance of 68.9 PFLOPS for water system and achieve 16 billion atoms simulation with a peak performance of 49.5 PFLOPS for copper system on the whole new Sunway machine.

However, scaling to 4,560 new Sunway nodes to simulate the copper system shows about 5% to 10% lower parallel efficiency compared to the Summit and Fugaku. Our analysis suggests that the primary reason is the limited LDM memory per CPE on the new Sunway processor. Each LDM only stores eight copper elements of the environment matrix, leading to poorer overlap between computation and communication and lower parallel efficiency in strong scaling.

2) Weak Scalability: We evaluate the weak scalability of the optimized DeePMD-kit and report the maximum simulation size (i.e., number of atoms) and the attainable FLOPS on water systems and copper systems with different node numbers, as shown in Fig. 15. According to the weak scaling results, the optimized DeePMD-kit can achieve 41.4 PFLOPS on copper systems with 13 billion atoms, and can further scale to 57.1 PFLOPS on water systems with 29 billion atoms. Through further estimation for the whole machine (i.e. 107,520 nodes reported in [39]), the re-designed DeePMD-kit implementation can perform large-scale simulations of 35 billion atoms with 68.9 PFLOPS (∼5% theoretical peak performance) on the new Sunway. Note that the performance utilization is primarily caused by the bandwidth limitation on the customized operators (including `Tabulate`, `TabulateGrad`, `ProdEnvMat`, `ProdForce`, and `ProdVirial`) in DeePMD-kit. Through further performance analysis on a single new Sunway node, the average achievable bandwidth of the operators in DeePMD-kit operators is 264.2 GB/s (∼87% DMA bandwidth) for the water system and 258.1 GB/s (∼84% DMA bandwidth) for the copper system. We remark that the SW206010-pro processor is only equipped with 307GB/s DDR4 memory, around 1/4 of the current HBM memory bandwidth. If equipped with HBM, our work will reach 263.6 PFLOPS (∼20% of the peak) without modifying a single line of code, which would be equivalent to our previous works on Fugaku (124.8 PFLOPS of 442 PFLOPS, 28% of the peak). Meanwhile, the SW26010-pro processor lacks a multi-level cache. If data reuse of the cache is considered, peak performance could be further enhanced.

VII. Related Works

1) Neural Network Molecular Dynamics (NNMD): NNMD presents an efficient way to fit the high-dimensional nonlinear PES [17]. It can reach the accuracy of AIMD while reducing computational costs by several orders of magnitude [16]. Traditional NNMD relies on manually crafted descriptors and uses separate networks to predict the PES of each descriptor element. Behler-Parrinello network [18] and Simple-NN [19] compute ACSFs as the descriptor based on pairwise distances and angles among triplets of atoms, which serve as inputs of NN for predicting the PES. TensorMol [20] incorporates a charge transfer model based on ACSFs, enhancing simulation accuracy for polar or charged molecules. In contrast, the DeePMD method [21] offers an end-to-end way of constructing a symmetry-preserving embedding network as the descriptor and using the fitting net to predict the accurate and robust PES and atomic force for various systems. Meanwhile, many NNMD methods employ GNNs to increase the prediction accuracy of the PES and atomic interactions. SchNet [26] and PhysNet [27] devise a massage-passing network to enhance MD performance. NEquIP [28] and PaiNN [29] introduce the equivariance GNN to accurately capture the interaction characteristics between molecules, enabling more complex MD simulations but introducing more computation costs.

2) Large Scale Molecular Dynamics: With the help of HPC, many works have developed large-scale NNMD packages to extend the spatial and temporal scales, enabling the simulation of more realistic scenarios. The Simple-NN [19] package employs TensorFlow for high expandability and efficient training, interfaced with LAMMPS for large-scale molecular dynamics simulations. The N2P2 package [36] provides neural network potentials with OpenMP support for efficient neural network potential energy and force evaluations interfaced with LAMMPS. The SNAP ML-IAP package [37] provides several designs including reduction of computational complexity, extraction of parallelism, data layout optimizations, and increasing arithmetic intensity to achieve excellent strong scaling on Summit. Allegro [30] enhances the accuracy and sample efficiency of NNMD simulations through innovative model architecture, massive parallelization, and models and implementations optimized for efficient GPU utilization. DeePMD-kit [31] further extends the spatial and temporal scales through

the optimizations including model tabulation, kernel fusion, and redundancy removal.

3) Molecular Dynamics on Emerging Architecture: Recently, the emerging many-core architectures such as Sunway and FPGA have provided novel platforms to further enhance the computational performance of MD. Gao et al. [33] have designed a series of optimizations for Tersoff potential in LAMMPS on new Sunway Supercomputer to achieve 2^{32} atoms simulation with nanoseconds. Gao et al. [32] have redesigned the AIREBO Bond-Order Potential on the new Sunway supercomputer and achieved 2 billion atoms simulation on 798,720 new Sunway cores. Wu et al. [43] have devised an optimized FPGA-based MD range-limited accelerator with several design variations processing 50K particles without off-chip memory. Rodríguez-Borbón et al. [44] present the FPGA-based accelerator for fast and energy-efficient quantum dynamics simulations of large chemical/material systems. These prior FPGA-based approaches also used mixed precision methods to enable the acceleration of atomistic simulations. Shaw et al. [45] have designed a massively parallel machine called Anton, which can execute millisecond-scale classical MD simulations of such biomolecular systems.

VIII. Conclusion

In this paper, we re-design the DeePMD-kit and enable large-scale *ab initio* molecular dynamics on the new Sunway supercomputer. We carry out large-scale parallelization, optimizations for time-consuming operators, and the mixed precision scheme to exploit the computation power of the heterogeneous many-core processors of the new Sunway. The re-designed DeePMD-kit can efficiently perform unprecedented large-scale simulations of 29 billion atoms with a peak performance of 57.1 PFLOPS (around 5% peak performance of 90,000 nodes, ∼84% of the entire supercomputer). Our work brings opportunities for simulating realistic scenarios like mechanical properties of metals, semiconductor devices, and other materials and physical systems requiring up to billions of atoms over nanosecond timescales.

Acknowledgment

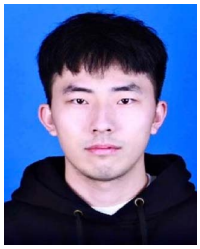
This work used computational resources of the supercomputer Fugaku provided by RIKEN through the HPCI System Research Project (Project ID: hp220310).

References

[1] J. M. Saveant, "Electron transfer, bond breaking, and bond formation," *Accounts Chem. Res.*, vol. 26, no. 9, pp. 455–461, 1993.
[2] M. Chen et al., "Hydroxide diffuses slower than hydronium in water because its solvated structure inhibits correlated proton transfer," *Nature Chem.*, vol. 10, no. 4, pp. 413–419, 2018.
[3] M. A. Sk, A. Ananthanarayanan, L. Huang, K. H. Lim, and P. Chen, "Revealing the tunable photoluminescence properties of graphene quantum dots," *J. Mater. Chem. C*, vol. 2, no. 34, pp. 6954–6960, 2014.
[4] A. C. Van Duin, S. Dasgupta, F. Lorant, and W. A. Goddard, "ReaxFF: A reactive force field for hydrocarbons," *J. Phys. Chem. A*, vol. 105, no. 41, pp. 9396–9409, 2001.
[5] J. Wang, R. M. Wolf, J. W. Caldwell, P. A. Kollman, and D. A. Case, "Development and testing of a general amber force field," *J. Comput. Chem.*, vol. 25, no. 9, pp. 1157–1174, 2004.
[6] B. Jelinek et al., "Modified embedded atom method potential for Al, Si, Mg, Cu, and Fe alloys," *Phys. Rev. B*, vol. 85, no. 24, 2012, Art. no. 245102.
[7] J. Tersoff, "New empirical approach for the structure and energy of covalent systems," *Phys. Rev. B*, vol. 37, no. 12, p. 6991, 1988.
[8] P. Hohenberg and W. Kohn, "Inhomogeneous electron gas," *Phys. Rev.*, vol. 136, no. 3B, p. B864, 1964.
[9] W. Kohn and L. J. Sham, "Self-consistent equations including exchange and correlation effects," *Phys. Rev.*, vol. 140, no. 4A, p. A1133, 1965.
[10] W. Jia et al., "The analysis of a plane wave pseudopotential density functional theory code on a GPU machine," *Comput. Phys. Commun.*, vol. 184, no. 1, pp. 9–18, 2013.
[11] W. Jia et al., "Fast plane wave density functional theory molecular dynamics calculations on multi-GPU machines," *J. Comput. Phys.*, vol. 251, pp. 102–115, 2013.
[12] L. Wang, Y. Wu, W. Jia, W. Gao, X. Chi, and L.-W. Wang, "Large scale plane wave pseudopotential density functional theory calculations on GPU clusters," in *Proc. Int. Conf. High Perform. Comput., Netw., Storage Anal. (SC)*, 2011, pp. 1–10.
[13] L. Lin, J. Lu, L. Ying, R. Car, and W. E, "Fast algorithm for extracting the diagonal of the inverse matrix with application to the electronic structure analysis of metallic systems," *Commun. Math. Sci.*, vol. 7, p. 755, 2009.
[14] W. Hu et al., "2.5 million-atom ab initio electronic-structure simulation of complex metallic heterostructures with DGDFT," in *Proc. Int. Conf. High Perform. Comput., Netw., Storage Anal. (SC)*, Los Alamitos, CA, USA: IEEE Comput. Soc., 2022, pp. 48–60.
[15] Y.-J. Yan et al., "10-million atoms simulation of first-principle package LS3DF on Sugon supercomputer," *J. Comput. Sci. Technol*., vol. 39, no. 1, pp. 45–62, Jan.
[16] O. T. Unke et al., "Machine learning force fields," *Chem. Rev.*, vol. 121, no. 16, pp. 10142–10186, 2021.
[17] J. Behler, "Four generations of high-dimensional neural network potentials," *Chem. Rev.*, vol. 121, no. 16, pp. 10037–10072, 2021.
[18] J. Behler and M. Parrinello, "Generalized neural-network representation of high-dimensional potential-energy surfaces," *Phys. Rev. Lett.*, vol. 98, no. 14, 2007, Art. no. 146401.
[19] K. Lee, D. Yoo, W. Jeong, and S. Han, "SIMPLE-NN: An efficient package for training and executing neural-network interatomic potentials," *Comput. Phys. Commun.*, vol. 242, pp. 95–103, 2019.
[20] K. Yao, J. E. Herr, D. W. Toth, R. Mckintyre, and J. Parkhill, "The tensormol-0.1 model chemistry: A neural network augmented with long-range physics," *Chem. Sci.*, vol. 9, no. 8, pp. 2261–2269, 2018.
[21] L. Zhang, J. Han, H. Wang, R. Car, and E. Weinan, "Deep potential molecular dynamics: A scalable model with the accuracy of quantum mechanics," *Phys. Rev. Lett.*, vol. 120, no. 14, 2018, Art. no. 143001.
[22] H. Wang, L. Zhang, J. Han, and E. Weinan, "DeePMD-kit: A deep learning package for many-body potential energy representation and molecular dynamics," *Comput. Phys. Commun.*, vol. 228, pp. 178–184, 2018.
[23] L. Zhang, H. Wang, R. Car, and E. Weinan, "Phase diagram of a deep potential water model," *Phys. Rev. Lett.*, vol. 126, no. 23, 2021, Art. no. 236001.
[24] F. Wang, Y. Sun, and J. Cheng, "Switching of redox levels leads to high reductive stability in water-in-salt electrolytes," *J. Amer. Chem. Soc.*, vol. 145, no. 7, pp. 4056–4064, 2023.
[25] L. Zhang, M. Chen, X. Wu, H. Wang, E. Weinan, and R. Car, "Deep neural network for the dielectric response of insulators," *Phys. Rev. B*, vol. 102, no. 4, 2020, Art. no. 041121.
[26] K. T. Schütt, P.-J. Kindermans, H. E. Sauceda Felix, S. Chmiela, A. Tkatchenko, and K.-R. Müller, "SchNet: A continuous-filter convolutional neural network for modeling quantum interactions," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, Red Hook, NY, USA: Curran Associates Inc., 2017, pp. 992–1002.
[27] O. T. Unke and M. Meuwly, "PhysNet: A neural network for predicting energies, forces, dipole moments, and partial charges," *J. Chem. Theory Comput.*, vol. 15, no. 6, pp. 3678–3693, 2019.
[28] S. Batzner et al., "E (3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials," *Nature Commun.*, vol. 13, no. 1, p. 2453, 2022.
[29] K. Schütt, O. Unke, and M. Gastegger, "Equivariant message passing for the prediction of tensorial properties and molecular spectra," in *Proc. Int. Conf. Mach. Learn.*, PMLR, 2021, pp. 9377–9388.
[30] B. Kozinsky, A. Musaelian, A. Johansson, and S. Batzner, "Scaling the leading accuracy of deep equivariant models to biomolecular simulations

of realistic size," in *Proc. Int. Conf. High Perform. Comput., Netw., Storage Anal.*, 2023, pp. 1–12.
[31] Z. Guo et al., "Extending the limit of molecular dynamics with ab initio accuracy to 10 billion atoms," in *Proc. 27th ACM SIGPLAN Symp. Princ. Pract. Parallel Program.*, 2022, pp. 205–218.
[32] P. Gao et al., "Redesign and accelerate the AIREBO bond-order potential on the new sunway supercomputer," *IEEE Trans. Parallel Distrib. Syst.*, vol. 34, no. 12, pp. 3117–3132, Dec. 2023.
[33] P. Gao et al., "LMFF: Efficient and scalable layered materials force field on heterogeneous many-core processors," in *Proc. Int. Conf. High Perform. Comput., Netw., Storage Anal.*, 2021, pp. 1–14.
[34] X. Chen et al., "Increasing the efficiency of massively parallel sparse matrix-matrix multiplication in first-principles calculation on the new-generation sunway supercomputer," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 12, pp. 4752–4766, Dec. 2022.
[35] W. Jia et al., "Pushing the limit of molecular dynamics with ab initio accuracy to 100 million atoms with machine learning," in *Proc. Int. Conf. High Perform. Comput., Netw., Storage Anal. (SC)*, Piscataway, NJ, USA: IEEE, 2020, pp. 1–14.
[36] A. Singraber, J. Behler, and C. Dellago, "Library-based LAMMPS implementation of high-dimensional neural network potentials," *J. Chem. Theory Comput.*, vol. 15, no. 3, pp. 1827–1840, 2019.
[37] K. Nguyen-Cong et al., "Billion atom molecular dynamics simulations of carbon at extreme conditions and experimental time and length scales," in *Proc. Int. Conf. High Perform. Comput., Netw., Storage Anal.*, 2021, pp. 1–12.
[38] A. P. Thompson et al., "LAMMPS: A flexible simulation tool for particle-based materials modeling at the atomic, meso, and continuum scales," *Comput. Phys. Commun.*, vol. 271, 2022, Art. no. 108171.
[39] Y. Liu et al., "Closing the 'quantum supremacy' gap: Achieving real-time simulation of a random quantum circuit using a new sunway supercomputer," in *Proc. Int. Conf. High Perform. Comput., Netw., Storage Anal.*, 2021, pp. 1–12.
[40] M. Chen et al., "Ab initio theory and modeling of water," *Proc. Nat. Acad. Sci.*, vol. 114, no. 41, pp. 10846–10851, 2017.
[41] Y. Zhang et al., "DP-GEN: A concurrent learning platform for the generation of reliable deep learning based potential energy models," *Comput. Phys. Commun.*, vol. 253, 2020, Art. no. 107206.
[42] D. Lu et al., "86 PFLOPS deep potential molecular dynamics simulation of 100 million atoms with ab initio accuracy," *Comput. Phys. Commun.*, vol. 259, 2021, Art. no. 107624.
[43] C. Wu et al., "FPGA-accelerated range-limited molecular dynamics," *IEEE Trans. Comput.*, vol. 73, no. 6, pp. 1544–1558, Jun. 2024.
[44] J. Rodríguez-Borbón, A. Kalantar, S. S. R. K. C. Yamijala, M. B. Oviedo, W. Najjar, and B. M. Wong, "Field programmable gate arrays for enhancing the speed and energy efficiency of quantum dynamics simulations," *J. Chem. Theory Comput.*, vol. 16, no. 4, pp. 2085–2098, Apr. 2020.
[45] D. E. Shaw et al., "Anton 3: Twenty microseconds of molecular dynamics simulation before lunch," in *Proc. Int. Conf. High Perform. Comput., Netw., Storage Anal.*, 2021, pp. 1–11.

Xun Wang received the Ph.D. degree in social systems and management from Tsukuba University. Currently, she is a Professor with the School of Computer Science and Technology, China University of Petroleum, Qingdao, China. Her research interests include bioinformatics, parallel computing, and computational materials science.



Xiangyu Meng is working toward the Ph.D. degree with the College of Computer Science and Technology, China University of Petroleum, Qingdao, China. His research interests include bioinformatics, parallel computing, and computational materials science.

Zhuoqiang Guo is working toward the Ph.D. degree with the Institute of Computing Technology, Chinese Academy of Sciences, and the University of Chinese Academy of Sciences, Beijing, China. His research interests include parallel computing, large-scale scientific computing, and high-performance computing.

Mingzhen Li received the Ph.D. degree from the School of Computer Science and Engineering, Beihang University, in 2023. Currently, he is an Assistant Professor with the Institute of Computing Technology, Chinese Academy of Sciences. He is currently working on identifying performance opportunities for scientific applications and machine learning systems. His research interests include high-performance computing, performance optimization, and cluster scheduling.

Lijun Liu is an Assistant Professor with the Department of Mechanical Engineering, Osaka University, Japan. Her research is directed toward the development and analysis of metal and nanostructured materials. Her research interests include parallel-in-time molecular dynamics, first-principles calculation, and high-performance computing.

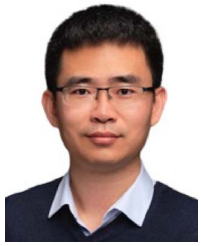
Mingfan Li received the B.S. degree in computer science from the University of Electronic Science and Technology of China, in 2017, and the Ph.D. degree from the University of Science and Technology of China, in 2023. His research interests include dataflow systems, parallel and distributed computing in heterogeneous environments, and artificial intelligence in the high-performance computing supercomputing ecosystem.

Qian Xiao is working toward the Ph.D. degree in computer science with the University of Science and Technology of China. His research interests include high-performance computing, compiler optimization, data flow computing, and artificial intelligence framework.

Tong Zhao received the Ph.D. degree in operation research and control theory from Fudan University, Shanghai, in 2021. Currently, he is a Postdoctoral Researcher with the Institute of Computing Technology, Chinese Academy of Science, Beijing. His research interests include the fundamental theory of artificial intelligence, high-performance computing, and game theory.

Ninghui Sun received the Ph.D. degree from the Institute of Computing Technology, Chinese Academy of Science, Beijing, in 1999. Currently, he is the Academic Director of the Institute of Computing, Chinese Academy of Sciences, Beijing. His research interests include parallel processing architecture, distributed operating systems, performance evaluation, and file systems.

Guangming Tan (Senior Member, IEEE) is currently a Professor with the Institute of Computing Technology, Chinese Academy of Sciences. His research interests include parallel programming and algorithms, domain-specific architecture, and bioinformatics. He has served as an Associate Editor of IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS and a PC member for SC, PPoPP, and ICS. He has published papers including conference/ journals such as SC, PLDI, PPoPP, and IEEE/ACM TRANSACTIONS.



Weile Jia is currently a Professor with the Institute of Computing Technology, Chinese Academy of Sciences. His research interests focus on high-performance computing, artificial intelligence for science, and first-principle calculation.