

SOFTWARE NOTE OPEN ACCESS

Scaling of the GROMACS Molecular Dynamics Code to 65k CPU Cores on an HPC Cluster

Carsten Kutzner¹ | Vedran Miletic² | Karen Palacio Rodríguez³ | Markus Rampp² | Gerhard Hummer³ | Bert L. de Groot¹ | Helmut Grubmüller¹

¹Theoretical and Computational Biophysics, Max Planck Institute for Multidisciplinary Sciences, Göttingen, Germany | ²Max Planck Computing and Data Facility, Garching, Germany | ³Max Planck Institute of Biophysics, Frankfurt am Main, Germany

Correspondence: Carsten Kutzner (ckutzne@mpinat.mpg.de)

Received: 5 December 2024 | **Revised:** 22 January 2025 | **Accepted:** 26 January 2025

Funding: This work was supported by Bundesministerium für Bildung und Forschung (Project No. 16ME0713) and European Commission (Project No. HORIZON-EUROHPC-JU-2021-COE-01-02).

Keywords: benchmark | GROMACS | high performance computing | molecular dynamics | MPI

ABSTRACT

We benchmarked the performance of the GROMACS 2024 molecular dynamics (MD) code on a modern high-performance computing (HPC) cluster with AMD CPUs on up to 65,536 CPU cores. We used five different MD systems, ranging in size from about 82,000 to 204 million atoms, and evaluated their performance using two different Message Passing Interface (MPI) libraries, Intel-MPI and Open-MPI. The largest system showed near-perfect strong scaling up to 512 nodes or 65,536 cores, maintaining a parallel efficiency above 0.9 even at the highest level of parallelization. Energy efficiency for a given number of nodes was generally equal to or slightly better than parallel efficiency. We achieved peak performances of 687 ns/d for the 82k atom system, 116 ns/d for the 53M atom system, and about 35 ns/d for the largest 204M atom system. These results demonstrate that highly optimized software running on a state-of-the-art HPC cluster provides sufficient computing power to simulate biomolecular systems at the mesoscale of viruses and organelles, and potentially small cells in the near future.

1 | Introduction

Molecular dynamics (MD) simulations have become an essential and powerful tool for understanding, from fundamental physics, the molecular mechanisms of “biological nanomachines”, that is, proteins such as enzymes, ion channels, and ribosomes. While most protein simulations involve between 10,000 and several hundred thousand atoms, recent advances in computer hardware and software enable the study of increasingly larger systems [1, 2], with the current world record exceeding one billion atoms [3]. These developments suggest that it may soon be possible to simulate an entire biological cell, which would comprise 500 million to six billion particles [4]. Even for smaller systems, a significant challenge is adequately sampling the vast configuration space to make statistically robust statements about

the system's average behavior. High-Performance Computing (HPC) accelerates the collection of statistical data by (i) parallelizing individual simulations across multiple compute nodes [5] and (ii) running multiple similar copies of the system concurrently. While the latter is trivial, the former poses significant challenges in terms of algorithms and software engineering. To maximize individual simulation performance and scaling, MD packages such as NAMD [6], Amber [7], LAMMPS [8], GENESIS [3], CHARMM [9], OpenMM [10], or GROMACS [11] employ several levels of parallelization, including message passing (MPI) and shared memory (OpenMP) parallelization.

To distribute an MD system across many compute nodes, GROMACS divides the simulation volume into $N = n_x \times n_y \times n_z$ domains, with interactions between particles in a domain

This is an open access article under the terms of the [Creative Commons Attribution](#) License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited.

© 2025 The Author(s). *Journal of Computational Chemistry* published by Wiley Periodicals LLC.

managed by a dedicated MPI rank. Each MPI rank can further utilize multiple OpenMP threads. In addition, separate MPI ranks [11] are used to compute the long-range electrostatic interactions to mitigate the limited parallelizability of the Particle Mesh Ewald [12] (PME) method [13], which relies on all-to-all communication between the involved ranks. To overcome this limitation, Fast Multipole Methods (FMM) have been recently implemented instead [14, 15].

Splitting the simulation system into domains often leads to load imbalances, as the number or complexity of interactions is usually not the same between domains. A further load imbalance occurs when separate PME ranks are used. Automatic load balancing mechanisms and built-in heuristics, such as estimating the optimal number of separate PME ranks, take much of the performance optimization burden off the scientist. However, when preparing for large-scale simulations on expensive high-end computing resources, it is generally advisable to compare the achieved performance with some kind of reference benchmark. Such reference performance numbers can also help to estimate the achievable simulation throughput and aid in the planning of new simulation campaigns.

Here we aim to provide such reference numbers for GROMACS for a large, modern, CPU-based HPC platform, and at the same time to demonstrate the potential for simulating very large MD systems on state-of-the-art hardware. We use the new AMD-based supercomputer “Viper” [16] of the Max Planck Society, operated at the Max Planck Computing and Data Facility (MPCDF) in Garching.

2 | Methods

In the last few years, AMD's EPYC CPUs have emerged as a competitor to Intel's Xeon family in the server and data center markets, including HPC. The competition between these two companies has driven innovation in the x86 CPU landscape, offering supercomputing centers more choices and driving advancements in performance, efficiency, openness, and cost-effectiveness. The AMD-based Viper HPC cluster has been operated by the MPCDF since July 2024. Each Viper node we used is equipped with two EPYC 9554 “Genoa” CPUs (128 physical cores, 3.1 GHz base frequency, turbo enabled) and 512 GB of RAM. The nodes are interconnected by an NVIDIA/Mellanox NDR200 (200 Gb/s) InfiniBand network with a non-blocking fat-tree topology. Further details on the hardware and software configuration can be found in the Viper user guide [16].

We used five different MD simulation systems for our benchmarks, ranging from 82,000 to 204 million atoms (Table 1). All of these are MD systems that have actually been or are being used in scientific projects and are not synthetic benchmarks. GROMACS .top input files for these systems are available at <https://www.mpinat.mpg.de/grubmueller/bench>. The MEM system has been used to study water permeation through an aquaporin tetramer that is embedded in a lipid bilayer [17]. The ribosome (RIB) system and similar ones are in current use for the study of the function of these protein factories in our cells [18–21]. Systems such as PEP (“peptides”) have been used to study the oligomerization of steric zipper peptides, a

TABLE 1 | Specifications of the benchmark systems.

MD	# atoms	System size (nm)	Time-step (fs)	Cutoff radii (nm)	PME grid spacing (nm)
MEM [17]	82k	10.8 × 10.2 × 9.6	2	1.0	0.12
RIB [18]	2M	31.2 × 31.2 × 31.2	4	1.0	0.135
PEP [19]	12.5M	50.0 × 50.0 × 50.0	2	1.2	0.16
LUM [20]	52.8M	166.3 × 144.0 × 21.8	2	1.1	0.16
SCF [20]	204.4M	166.4 × 144.1 × 84.2	2	1.1	0.16

process important for understanding the early stages of protein aggregation in many neurodegenerative diseases [19]. The largest systems are related to the human nuclear pore complex (hNPC) in the dilated conformation (PDB: 7R5J) [20]. The LUM system corresponds to the luminal ring of hNPC, while the SCF system is the hNPC scaffold consisting of the inner ring, nuclear ring, cytoplasmic ring, and luminal ring. Both systems are simulated in an explicit solvent with a salt concentration of 0.15 M NaCl. The NPC, arguably the largest protein assembly in human cells, consists of about 1000 proteins that form an 8-fold symmetric pore in the nuclear envelope. Its main function is to regulate the exchange of molecules between the nucleus and the cytoplasm. The hNPC adjusts its diameter in response to changes in nuclear envelope tension, which affects transport [22, 23]. We are currently studying the hNPC scaffold and luminal ring dynamics to elucidate the molecular mechanism behind this structural flexibility and to gain key insights into its role in cellular regulation and disease.

For all benchmarks, we compiled GROMACS 2024.2 in mixed precision with AVX_512 SIMD instructions and OpenMP support, using the GCC 14.1.0 compiler. We built two versions, one with Intel-MPI 2021.11 and a second one with Open-MPI 4.1.6. Both MPI installations are based on UCX 1.16, which acts as a middle layer between the MPI library and the InfiniBand network hardware. For the fast Fourier transforms required by PME we use the FFTW library version 3.3.10 [24].

In our performance tests, we relied on the quite well-performing empirical defaults for the parameters GROMACS can control, such as the number of separate PME nodes and whether or not to enable dynamic load balancing. We have generally refrained from extensive performance fine-tuning, as this would not normally be done by practitioners. For systematic, in-depth performance analysis and tuning, tools such as MDBenchmark [25]

can help automate setup and analysis. One of the runtime settings that GROMACS cannot control is the partitioning of available CPU cores into MPI ranks and OpenMP threads, since the MPI processes are started outside of GROMACS. Since this can significantly affect performance, we benchmarked 1, 2, 4, and 8 OpenMP threads per MPI rank, always using all physical cores of a node, that is, in terms of ranks \times threads per node 1×128 , 2×64 , 4×32 , and 8×16 . From these settings, we report the one that performed best. Simultaneous Multithreading (SMT) did not improve RIB performance on a single-node (in fact, it was slightly worse with SMT), therefore we did not use SMT.

Each benchmark was run for 0.2–0.5 h of wall clock time. Performance was measured for the second half of the run because load balancing mechanisms take some time to reach optimal balance and because memory allocations tend to slow down the first few time steps.

For the Intel-MPI benchmarks, we set the environment variable `I_MPI_FABRICS=shm:ofi` to explicitly enable shared memory transport (shm) for enhanced performance. Shared memory transport was temporarily disabled by default in Intel-MPI, but will be enabled again in future releases starting with Intel-MPI 2021.14.

To extract the power consumption of each GROMACS job, we used the “Smart Energy Manager Suite” (SEMS) provided by Eviden.

3 | Results

Figure 1 shows the strong-scaling behavior for MD systems ranging from 82k to 204M particles between 128 and 65,536 cores (bottom axis) or one to 512 nodes (top axis), respectively. The largest 204M system closely follows the diagonal gray line, indicating near-perfect scaling from four to 512 nodes. Even for the smaller systems (except MEM), scaling remains excellent for a smaller number of nodes. As expected, with decreasing system size, deviations from ideal scaling occur at lower node counts due to the comparatively smaller number of atoms per node and, hence also smaller workload per node.

We measured maximum performances of 687 ns/d for the 82k (MEM) system, 256 ns/d for the 2M (RIB) system, and 226 ns/d for the 12M (PEP) system on Viper, which, to our knowledge, are the highest performances ever reported for these specific benchmark systems, including on GPU-accelerated nodes [11, 26–29].

For systems larger than ten million atoms, the GROMACS performances for Intel-MPI and Open-MPI are very similar. For the smaller 82k and 2M systems, Intel-MPI often outperforms Open-MPI, with up to a 37% performance increase for the 82k system on two nodes, and up to a 19% increase for the 2M system on 16 nodes. For the 204M benchmark system, only Intel-MPI results are available for ≥ 64 nodes, as all Open-MPI runs were aborted due to memory limitations. If desired, benchmark results can probably be further improved for both MPI libraries by setting MPI runtime parameters optimized for the application and hardware setup. These runtime parameters would have to be determined beforehand by running MPI benchmarks.

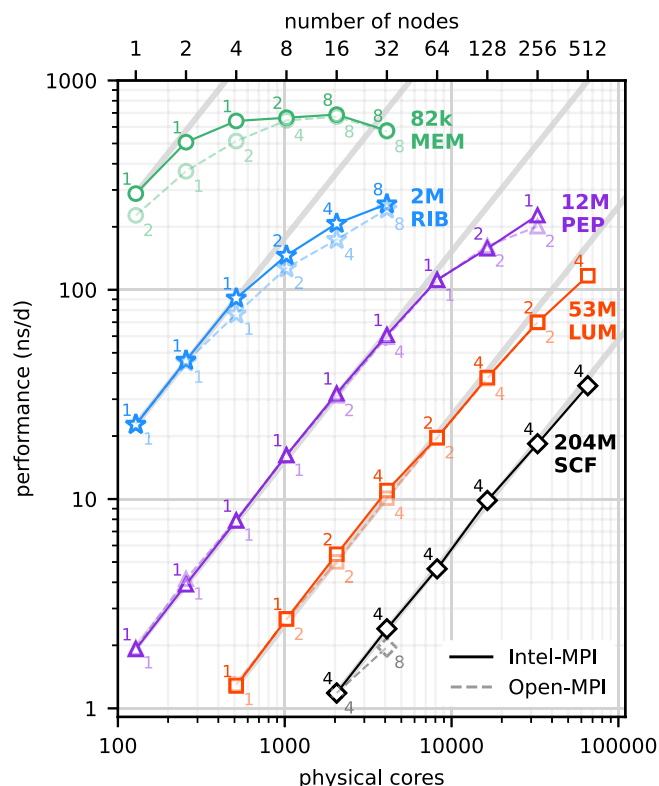


FIGURE 1 | Strong scaling of GROMACS on up to 65,536 CPU cores. Measured performances of GROMACS 2024 on up to 512 Viper nodes with Intel-MPI (solid) and Open-MPI (dashed, semi-transparent) for five MD systems (colors) ranging from 82k to 204M atoms in size. Numbers beside symbols refer to the optimal number of OpenMP threads per MPI rank. Diagonal gray lines indicate perfect scaling.

The numbers next to the symbols in Figure 1 indicate the number of threads per MPI process that deliver the best performance. At low levels of parallelism, single-threaded MPI ranks generally provide the best performance, avoiding any potential OpenMP overhead. However, as the total number of MPI ranks is increased in order to utilize more and more CPU cores, one will eventually reach a fundamental limit set by the cutoff radius. The latter limits the minimum domain size, which translates into a maximum number of domains (and hence MPI ranks) that can be used. When the maximum number of domains is reached, the only way to use more cores is to use more OpenMP threads for each MPI rank.

Figure 2 shows two metrics that indicate whether or not computer time is being used efficiently for a given number N of nodes. First, the parallel efficiency

$$E_{\text{par}}(N) = \frac{P(N)}{N \cdot P(1)} \quad (1)$$

with $P(N)$ denoting the performance on N nodes. Second, the energy efficiency, which is the performance on N nodes divided by the average power consumption $\text{pow}(N)$ of the involved nodes,

$$E_{\text{en}}(N) = \frac{P(N)}{\text{pow}(N)} \quad (2)$$

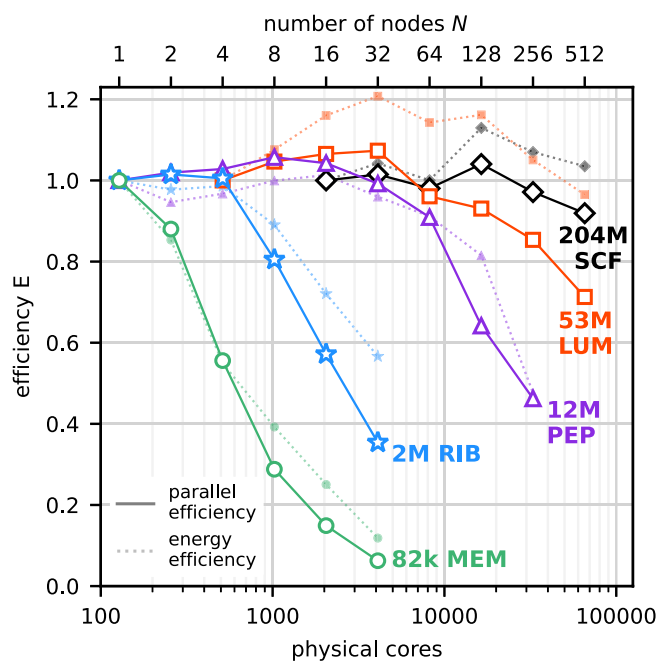


FIGURE 2 | Parallel efficiency of GROMACS on up to 65,536 CPU cores. For the five benchmarked systems (colored), two efficiency metrics are shown for the Intel-MPI benchmarks: (a) the parallel efficiency E_{par} (Equation (1), solid lines), and (b) the normalized energy efficiency E_{en} (Equation (2), dotted lines).

The plot shows the normalized energy efficiency, which is $E_{\text{en}}(N)$ divided by the energy efficiency at the smallest number of nodes that were tested.

For the small 82k system, the parallel efficiency E_{par} drops relatively quickly, but for larger systems, E_{par} remains above 0.8 even at very high levels of parallelization. If a parallel efficiency of at least 0.8 is targeted, the RIB system scales to about 2000 atoms per core, and the PEP system to 1500 atoms per core. For the PEP and LUM systems, we even observe a superlinear speedup ($E_{\text{par}} > 1$) at medium node counts. This can occur especially in strong scaling scenarios, for example, when doubling the number of nodes causes computations to fit better into the CPU cache. At the highest parallelization of 512 nodes or 65,536 cores, the two largest systems still show decent ($E_{\text{par}}^{\text{LUM}} \approx 0.71$) to excellent ($E_{\text{par}}^{\text{SCF}} \approx 0.92$) efficiency, yielding remarkable simulation speeds of ≈ 116 ns/d and ≈ 35 ns/d, respectively.

Normalized energy efficiency generally follows the same trend as parallel efficiency. However, at higher node counts, energy efficiency often becomes higher than parallel efficiency. In the regimes where ideal scaling is observed in Figure 1, the power draw is about 1.2 kW per node¹ which is constant over the runtime of the benchmark and varies only slightly between 1.1 kW per node for the larger and 1.3 kW per node for the smaller MD systems. Thus, in this regime, the total energy to the solution can simply be derived from the runtime and the number of compute nodes used. For example, the energy to solution for the large 204M system amounts to $\approx (256 \cdot 1.1 \text{ kW}) / (18.4 \text{ ns/d}) = 367 \text{ kWh/ns}$.

Where the simulation performance deviates from ideal scaling, the required power per node decreases as nodes become underutilized. For example, running the RIB system on 32

compute nodes consumes only about 1 kW per node. In the regime of ideal scaling on 1–4 nodes, RIB achieves an energy efficiency of 17.0–17.4 ns/d per kW. This was roughly the efficiency of GPU nodes five years ago and represents a threefold improvement over CPU nodes of that age [31].

4 | Conclusions

Our GROMACS benchmarks on the new AMD-CPU-based HPC system of the Max Planck Society, Viper, for realistic and typical biomolecular systems show that the combination of powerful CPUs, a fast interconnect, and well-tuned software makes it possible today to simulate extremely large MD systems with up to several hundred million atoms. For the largest system with 204 million atoms, the scaling behavior was shown to be close to ideal, with parallel efficiencies above 0.9 even at the highest parallelization levels (512 nodes with altogether 65,536 cores), while maintaining high energy efficiency. This demonstrates that modern HPC machines now provide the level of performance needed for simulating MD systems of the size of viruses, organelles, and eventually, small cells.

Besides the interconnect hardware, a high-performance MPI communication library is crucial for enabling the highest GROMACS simulation performance at scale. While Intel-MPI consistently delivered the best performance for our benchmarks, Open-MPI is found to be a very competitive alternative.

Acknowledgments

Many thanks to Harald Braun, John Donners (Eviden), Rafael Lago (Intel), Renate Dohmen, and Mykola Petrov (MPCDF) for their assistance. This work was supported by the German Federal Ministry of Education and Research (BMBF) as part of its initiative “SCALEX – New Methods and Technologies for Exascale Computing” (BMBF project 16ME0713), and by BioExcel CoE (www.bioexcel.eu), a project funded by the European Union contract HORIZON-EUROHPC-JU-2021-COE-01-02. K.P.R. acknowledges support from the “Hessen Horizon Marie Skłodowska-Curie-Stipendium” program. All authors acknowledge support from the Max Planck Society. Open Access funding enabled and organized by Projekt DEAL.

Data Availability Statement

The data that support the findings of this study are available from the corresponding author upon reasonable request.

Endnotes

¹ For comparison, a single-node High-Performance-Linpack run (AMD Zen HPL [30]) draws ca. 1.5 kW per node, which reflects GROMACS being a highly optimized, compute-bound application.

References

1. E. Wilson, J. Vant, J. Layton, et al., “Large-Scale Molecular Dynamics Simulations of Cellular Compartments,” in *Structure and Function of Membrane Proteins*, eds. I. Schmidt-Krey and J. C. Gumbart (Humana, 2021), 335–356.
2. V. Gapsys, W. Kopec, D. Matthes, and B. L. de Groot, “Biomolecular Simulations at the Exascale: From Drug Design to Organelles and Beyond,” *Current Opinion in Structural Biology* 88 (2024): 102887.

3. J. Jung, C. Kobayashi, K. Kasahara, et al., "New Parallel Computing Algorithm of Molecular Dynamics for Extremely Huge Scale Biological Systems," *Journal of Computational Chemistry* 42 (2021): 231–241.
4. J. A. Stevens, F. Grünewald, P. M. van Tilburg, et al., "Molecular Dynamics Simulation of an Entire Cell," *Frontiers in Chemistry* 11 (2023): 1106495.
5. H. Grubmüller, H. Heller, and K. Schulten, "Molecular Dynamics Simulation on a Parallel Computer," *Molecular Simulation* 5 (1990): 133–165.
6. J. C. Phillips, R. Braun, W. Wang, et al., "Scalable Molecular Dynamics With NAMD," *Journal of Computational Chemistry* 26 (2005): 1781–1802.
7. R. Salomon-Ferrer, D. A. Case, and R. C. Walker, "An Overview of the Amber Biomolecular Simulation Package," *Wiley Interdisciplinary Reviews: Computational Molecular Science* 3 (2013): 198–210.
8. A. P. Thompson, H. M. Aktulga, R. Berger, et al., "LAMMPS—A Flexible Simulation Tool for Particle-Based Materials Modeling at the Atomic, Meso, and Continuum Scales," *Computer Physics Communications* 271 (2022): 108171.
9. B. R. Brooks, C. L. Brooks, III, A. D. Mackerell, Jr., et al., "CHARMM: The Biomolecular Simulation Program," *Journal of Computational Chemistry* 30 (2009): 1545–1614.
10. P. Eastman, R. Galvelis, R. P. Peláez, et al., "OpenMM 8: Molecular Dynamics Simulation With Machine Learning Potentials," *Journal of Physical Chemistry. B* 128 (2024): 109–116.
11. S. Páll, M. J. Abraham, C. Kutzner, B. Hess, and E. Lindahl, "Tackling Exascale Software Challenges in Molecular Dynamics Simulations With GROMACS," in *Lecture Notes in Computer Science EASC 2014*, eds. S. Markidis and E. Laure (Springer International Publishing Switzerland, 2015), 1–25.
12. U. Essmann, L. Perera, M. Berkowitz, T. Darden, and H. Lee, "A Smooth Particle Mesh Ewald Method," *Journal of Chemical Physics* 103 (1995): 8577–8593.
13. A. Ayala, S. Tomov, M. Stoyanov, and J. Dongarra, *Scalability Issues in FFT Computation* (Parallel Computing Technologies, 2021), 279–287.
14. Y. Ohno, R. Yokota, H. Koyama, et al., "Petascale Molecular Dynamics Simulation Using the Fast Multipole Method on K Computer," *Computer Physics Communications* 185 (2014): 2575–2585.
15. B. Kohnke, C. Kutzner, and H. Grubmüller, "A GPU-Accelerated Fast Multipole Method for GROMACS: Performance and Accuracy," *Journal of Chemical Theory and Computation* 16 (2020): 6938–6949.
16. Max Planck Computing and Data Facility, "Viper User Guide," <https://docs.mpcdf.mpg.de/doc/computing/viper-user-guide.html>.
17. B. L. de Groot and H. Grubmüller, "Water Permeation Across Biological Membranes: Mechanism and Dynamics of Aquaporin-1 and GlpF," *Science* 294 (2001): 2353–2357.
18. L. Bock, C. Blau, G. Schröder, et al., "Energy Barriers and Driving Forces in tRNA Translocation Through the Ribosome," *Nature Structural & Molecular Biology* 20 (2013): 1390–1396.
19. D. Matthes, V. Gapsys, and B. L. de Groot, "Driving Forces and Structural Determinants of Steric Zipper Peptide Oligomer Formation Elucidated by Atomistic Simulations," *Journal of Molecular Biology* 421 (2012): 390–416.
20. S. Mosalaganti, A. Obarska-Kosinska, M. Siggel, et al., "AI-Based Structure Prediction Empowers Integrative Structural Analysis of Human Nuclear Pores," *Science* 376, no. 6598 (2022): eabm9506.
21. L. V. Bock, S. Gabrielli, M. H. Kolár, and H. Grubmüller, "Simulation of Complex Biomolecular Systems: The Ribosome Challenge," *Annual Review of Biophysics* 52 (2023): 361–390.
22. C. E. Zimmerli, M. Allegretti, V. Rantos, et al., "Nuclear Pores Dilate and Constrict in Cellulo," *Science* 374, no. 6573 (2021): eabd9776.
23. P. C. Hoffmann, H. Kim, A. Obarska-Kosinska, et al., "Nuclear Pores as Conduits for Fluid Flow During Osmotic Stress," *BioRxiv* (2024).
24. M. Frigo and S. G. Johnson, "The Design and Implementation of FFTW3," *Proceedings of the IEEE* 93 (2005): 216–231.
25. M. Gecht, M. Siggel, M. Linke, G. Hummer, and J. Köfinger, "MD-Benchmark: A Toolkit to Optimize the Performance of Molecular Dynamics Simulations," *Journal of Chemical Physics* 153 (2020): 144105.
26. G. G. Alan Gray and S. Páll, "Massively Improved Multi-Node NVIDIA GPU Scalability With GROMACS," 2023, <https://developer.nvidia.com/blog/massively-improved-multi-node-nvidia-gpu-scalability-with-gromacs/>.
27. N. Kondratyuk, G. Smirnov, A. Agarkov, et al., "Performance and Scalability of Materials Science and Machine Learning Codes on the State-Of-Art Hybrid Supercomputer Architecture," in *Russian Supercomputing Days* (Springer, 2019), 597–609.
28. N. A. Simakov, M. D. Jones, T. R. Furlani, E. Siegmann, and R. J. Harrison, "First Impressions of the NVIDIA Grace CPU Superchip and NVIDIA Grace Hopper Superchip for Scientific Workloads," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis in Asia-Pacific Region Workshops* (2024), 36–44.
29. C. Kutzner, C. Knip, A. Cherian, et al., "GROMACS in the Cloud: A Global Supercomputer to Speed up Alchemical Drug Design," *Journal of Chemical Information and Modeling* 62 (2022): 1691–1711.
30. AMD, "Zen HPL Version 2023_07," <https://www.amd.com/de/developer/zen-software-studio/applications/pre-built-applications/zen-hpl.html>.
31. C. Kutzner, S. Páll, M. Fechner, A. Esztermann, B. L. de Groot, and H. Grubmüller, "More Bang for Your Buck: Improved Use of GPU Nodes for GROMACS 2018," *Journal of Computational Chemistry* 40 (2019): 2418–2431.