GPU Acceleration of Coarse-Grained Molecular Dynamics Simulations in GENESIS Using OpenACC

Cheng Tan[1], **Jaewoon Jung**[1,2], **Diego Ugarte La Torre**[1], **Chigusa Kobayashi**[1], **and Yuji Sugita**[1,2,3]

[1]RIKEN Center for Computational Science, [2]RIKEN Pioneering Research Institute, [3]Department of Physics, University of Tokyo

Abstract

To meet the increasing computational demands of large-scale biomolecular simulations, we extended the GENESIS molecular dynamics (MD) software by offloading its core coarse-grained (CG) calculation kernels to GPUs using OpenACC. By targeting the primary bottlenecks in nonbonded force evaluations and neighbor-list construction, we achieved a portable, single-source implementation that maintains full compatibility with existing CPU workflows. Benchmark tests on diverse protein systems demonstrate substantial performance gains over multi-core CPU execution, significantly reducing wall-clock time for CG MD simulations. This work successfully enhances the throughput of GENESIS on modern heterogeneous architectures, providing a robust foundation for future large-scale biophysical research on GPU-accelerated HPC systems.

Introduction

Molecular Dynamics (MD) simulation is a cornerstone of biophysics, providing atomistic insights into the dynamic behavior of complex biomolecular systems. By simplifying groups of atoms into single interaction sites, coarse-grained (CG) models significantly extend the accessible timescales and system sizes, enabling the study of large-scale biological phenomena such as protein droplets formed via liquid-liquid phase separation.

GENESIS [1] provides a versatile platform for biomolecular MD through its ATDYN engine [2], which emphasizes flexibility, and its CGDYN engine [3], which utilizes dynamic domain decomposition for nonuniform CG systems. Despite their high efficiency on conventional CPU clusters, the exponential growth of simulation targets, such as massive protein droplets, has created a performance bottleneck. To overcome these throughput limits, a shift from CPU-centric to GPU-accelerated computing is now essential. To balance high performance with code maintainability, we chose OpenACC over CUDA. This allows us to keep the codebase readable and easily modifiable by scientists while harnessing GPU power.

However, achieving ideal performance requires more than just adding directives. Our development focuses on deep technical optimizations, including asynchronous task streaming, minimized host-device data transfers, and optimized GPU occupancy. By carefully considering the underlying computational flow, we have focused on the HPS model [4] for large-scale protein condensate simulations and transformed the CPU-oriented ATDYN into a high-performance GPU-accelerated engine, bridging the gap between simple code implementation and high computing efficiency.

Main Objectives

1. **Substantial Performance Acceleration**: To achieve significant speedups in CG MD simulations by offloading the most time-consuming kernels to a single GPU.
2. **Seamless User Experience & Compatibility**: To maintain a unified workflow where users can execute simulations on either CPU or GPU platforms using identical input scripts.
3. **Scientific Rigor & Numerical Accuracy**: To ensure that the GPU-accelerated code produces trajectories and energy profiles that are numerically consistent with the original CPU version.

Models and Methods

Langevin Dynamics

To simulate the dynamical evolution in an implicit-solvent environment, we employ the Langevin equation:

$$m_i \frac{d^2\mathbf{r}_i}{dt^2} = \mathbf{F}_i(\{\mathbf{r}\}) - \gamma_i \frac{d\mathbf{r}_i}{dt} + \mathbf{R}_i(t), \tag{1}$$

where $\mathbf{F}_i = -\nabla_i V_{\text{total}}$ is the conservative force , γ_i is the friction coefficient, and $\mathbf{R}_i(t)$ represents the stochastic force satisfying the fluctuation-dissipation theorem:

$$\langle \mathbf{R}_i(t) \cdot \mathbf{R}_j(t') \rangle = 2k_B T \gamma_i \delta_{ij} \delta(t - t'). \tag{2}$$

We utilize the `cuRAND` library to produce pseudo-random numbers with minimal overhead.

Coarse-Grained Potential: the HPS Model

The system's potential energy is defined by the Hydropathy Scale (HPS) model. The total potential V_{total} is the sum of bonded, electrostatic, and short-range non-local interactions.

The bonded potential is a harmonic function of the bond length:

$$V_{\text{bond}}(r) = k_b (r - r_0)^2, \tag{3}$$

where r_0 is the equilibrium distance and k_b is the force constant.

The electrostatic interactions are modeled by the Debye-Hückel theory:

$$V_{\text{ele}}(r) = \frac{q_i q_j e^{-r/\lambda_D}}{4\pi\varepsilon_0 \varepsilon_r r}, \tag{4}$$

where q_i and q_j are charges, ε_0 is the dielectric permittivity of vacuum, ε_r is the relative permittivity of solution, and λ_D is the Debye screening length.

The critical short-range non-bonded interactions (refered as "HPS" potential) is given by:

$$V_{\text{HPS}}(r) = \begin{cases} 4\epsilon \left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \right] + (1-\lambda)\epsilon, & r < 2^{1/6}\sigma \\ 4\lambda\epsilon \left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \right], & r \geq 2^{1/6}\sigma \end{cases} \tag{5}$$

where ϵ and σ are force coefficient and interaction radius, and λ is a value reflecting the hydrophobicity of each residue type.

Dual Pair-list Construction Strategy

We implemented a specialized neighbor-list scheme for CG models:

- **Cell-linked List**: The simulation box is partitioned into spatial grid cells. A parallelized cell-linked list algorithm rapidly filter neighboring particles, significantly reducing the complexity of pair-list construction.
- **Dual-Cutoff Mechanism**: In CG simulations, the Debye-Hückel approximation of electrostatics results in a notably slow decay of forces than that of the HPS forces. To address this disparity, we employ two independent pair-lists with distinct cutoff radii.

Results

Validation of Numerical Accuracy

Property	CPU (DP)	GPU (SP)	Relative Δ
V_{bond}	27717.8260	27717.8613	1.3×10^{-6}
V_{ele}	4314.9350	4314.9321	6.7×10^{-7}
V_{HPS}	203925.0650	203925.2031	6.8×10^{-7}
V_{total}	235957.8260	235958.0000	7.4×10^{-7}
$\lvert F_{\text{it-5}} \rvert^*$	6382.8441	6382.9399	1.5×10^{-5}

Table 1: Numerical deviations in energy (in kcal/mol) and force (in kcal/mol/Å) (*every 5 atoms).

To maximize computational throughput on GPU architectures, single-precision (SP) floating-point operations were primarily utilized in the performance-critical kernels. Given the sensitive nature of biophysical trajectories, we conducted rigorous validation to ensure that this shift does not compromise scientific integrity.

We compared the potential energy terms and force distributions between the original double-precision (DP) CPU version and our SP GPU implementation. The results, summarized in the left table, show that the numerical deviations remain well within the acceptable tolerance for molecular dynamics simulations, confirming that our optimization achieves high efficiency while maintaining robust physical accuracy.

Zero-Effort Transition for End-Users

A core philosophy of our development is to ensure a seamless transition for existing GENESIS users. The OpenACC implementation achieves complete operational transparency, requiring no modifications to the biophysical input scripts.

Integrated Build Workflow The configuration process for OpenACC:

```
$> ./configure --disable-mpi --disable-openmp --enable-single
         --enable-openacc CC=nvcc FC=nvfortran F77=nvfortran
```

Identical Output Files Both CPU and GPU versions generate identical output file structures. This allows users to use their existing parsing scripts and monitoring tools without any adjustments.

High-Throughput Simulation Efficiency

To evaluate the efficiency of the OpenACC-accelerated engine, we benchmarked the simulation performance across various system sizes. We performed our tests using these platforms:

- Intel(R) Xeon(R) Platinum 8480+ (32 OpenMP threads)
- NVIDIA H100 80GB (CUDA ver: 12.8, Driver ver: 570.124.06)
- NVIDIA GH200 480GB (CUDA ver: 13.0, Driver ver: 580.95.05)

To identify computational bottlenecks and validate hardware efficiency, we profiled the wall-clock time for 1,000 simulation steps across the three architectures.

Item	**Xeon** Time (s)		**H100** Time (s)		**H200** Time (s)	
V_{bond}	1.35 ± 1.35	1.65%	0.03 ± 0.03	2.78%	0.01 ± 0.01	1.30%
V_{ele}	1.43 ± 0.03	1.75%	0.06 ± 0.00	5.96%	0.06 ± 0.00	6.53%
V_{HPS}	22.41 ± 0.38	27.40%	0.71 ± 0.00	68.07%	0.71 ± 0.00	71.55%
Pair-list	13.31 ± 0.37	16.27%	0.16 ± 0.00	15.39%	0.13 ± 0.00	13.43%
Integrator	25.28 ± 0.06	30.90%	0.05 ± 0.00	4.71%	0.04 ± 0.00	4.21%
Total	81.81 ± 2.02	100.00%	1.04 ± 0.00	100.00%	0.99 ± 0.00	100.00%

Table 2: Runtime break down of a representative protein system containing 314,160 particles with standard charge distribution.

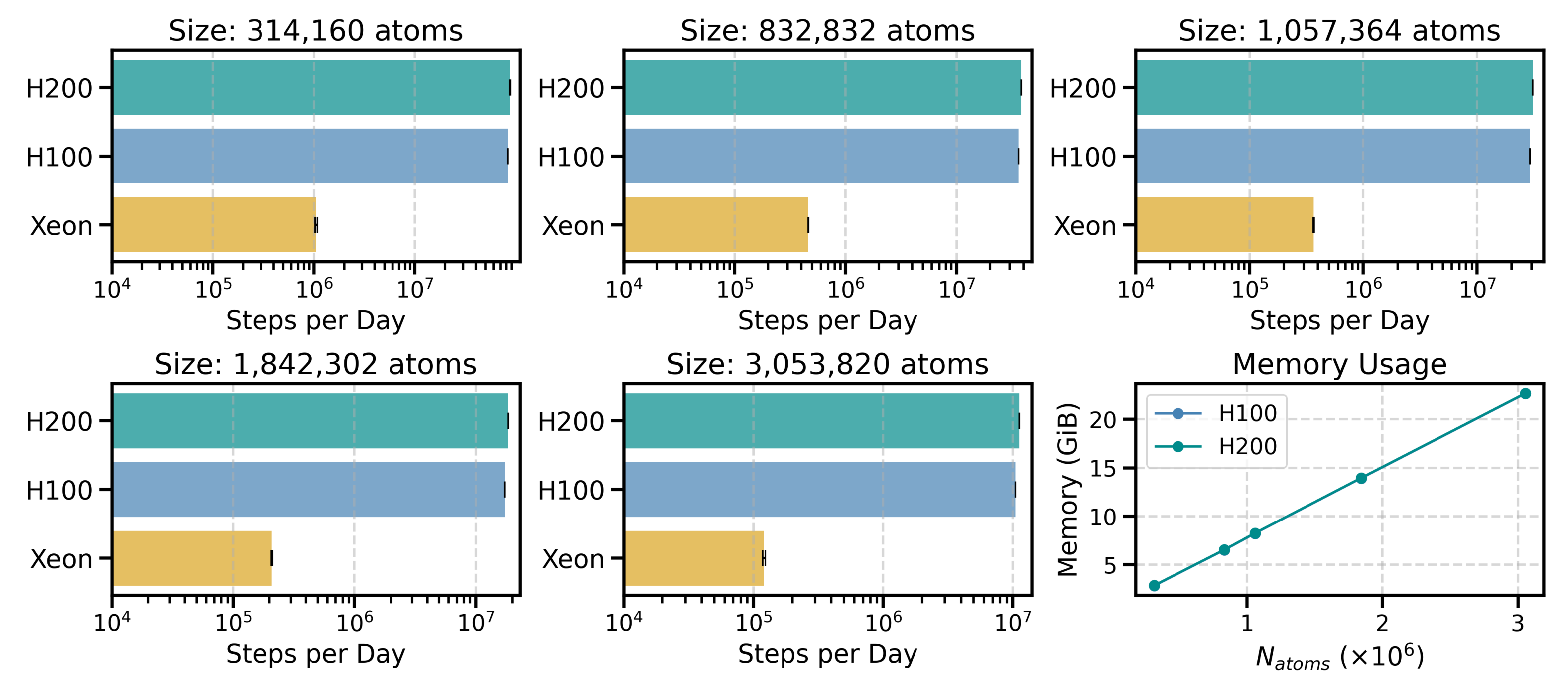

Figure 1: Performance and Memory Consumption Benchmarks. Comparison of simulation throughput (steps/day, log scale) between CPU (Xeon Platinum, 32 cores) and GPU (NVIDIA H100/H200) across varying system sizes ranging from 0.3 million to 3 million atoms.

Based on these results, we highlight three key achievements:

- **Orders of Magnitude Acceleration**: The OpenACC-accelerated engine achieves a substantial speedup ($\sim \times 100$) over the CPU baseline.
- **Robust Scalability**: Performance on NVIDIA H100 and H200 remains stable across all tested system sizes.
- **Memory Efficiency**: The linear memory usage confirms that 80GB GPUs can easily accommodate extremely large systems.

References

[1] J. Jung et al. *The Journal of Physical Chemistry B* **128** (2024):6028–6048.
[2] C. Tan et al. *PLOS Computational Biology* **18** (2022):e1009578.
[3] J. Jung, C. Tan, and Y. Sugita. *Nature Communications* **15** (2024):3370.
[4] G. L. Dignon et al. *PLoS computational biology* **14** (2018):e1005941.