Molecular Dynamics Simulations on Cloud Computing and Machine Learning Platforms

Prateek Sharma
prateeks@iu.edu
Indiana University Bloomington

Vikram Jadhao
vjadhao@iu.edu
Indiana University Bloomington

***Abstract*—Scientific computing applications have benefited greatly from high performance computing infrastructure such as supercomputers. However, we are seeing a paradigm shift in the computational structure, design, and requirements of these applications. Increasingly, data-driven and machine learning approaches are being used to support, speed-up, and enhance scientific computing applications, especially molecular dynamics simulations. Concurrently, cloud computing platforms are increasingly appealing for scientific computing, providing "infinite" computing powers, easier programming and deployment models, and access to computing accelerators such as TPUs (Tensor Processing Units). This confluence of machine learning (ML) and cloud computing represents exciting opportunities for cloud and systems researchers. ML-assisted molecular dynamics simulations are a new class of workload, and exhibit unique computational patterns. These simulations present new challenges for low-cost and high-performance execution. We argue that transient cloud resources, such as low-cost preemptible cloud VMs, can be a viable platform for this new workload. Finally, we present some low-hanging fruits and long-term challenges in cloud resource management, and the integration of molecular dynamics simulations into ML platforms (such as TensorFlow).**

I. Introduction

Scientific computing applications play an important role in the analysis and understanding of a wide variety of natural and synthetic processes. These applications are typically implemented as large-scale parallel programs that use communication frameworks such as MPI, and are largely deployed on high performance computing (HPC) infrastructure such as supercomputers. Molecular dynamics (MD) simulations are among the most ubiquitous scientific computing applications. These simulations have been used extensively by materials scientists, chemical engineers, and physicists to investigate the microscopic origins of the macroscopic behavior of materials such as self-assembled nanoparticles, viral capsids, electrolytes, lubricants, and polymers [1]–[4].

A key goal for MD simulations is to explore the design space associated with the material attributes, and establish the links between the design parameters and the material response (generally, encoded in the structural and dynamical properties). To this end, simulations are deployed as a collection or "bag" of jobs. Collectively, a bag of jobs "sweeps" a multi-dimensional design space and furnishes the links between the material design parameters (inputs) and the material response (outputs). These links provide a reliable guide to experiments for a rational discovery of regions of the material design space exhibiting interesting structural and dynamical properties.

The bags of jobs approach is also central to the rapidly developing area of using machine learning (ML) to enhance MD simulations and expedite the exploration of the material design space [5]–[14]. Large collections of jobs with independent parameter sets are launched in order to train and test the ML models designed to enhance the predictive power or reduce the computational costs of MD simulations. For example, artificial neural network based regression models, trained on data from MD simulations of soft materials, can successfully predict the relationships between the input parameters and the simulation outcomes [12], [15]. These ML surrogates accurately predicted the distributions of ions for a variety of confined electrolyte systems with 95% accuracy and an inference time $10000\times$ less than the corresponding MD simulation runtime [12], [15]. As the utility of MD simulations and their ML-enhanced versions in the rational design of materials is further demonstrated, it will be necessary for accurate and fast simulations to be performed for larger sets of parameters in order to efficiently explore the material design space. To this end, it is important to leverage diverse advanced cyberinfrastructure platforms to perform low-cost simulations.

II. Molecular Dynamics on Cloud Platforms

Increasingly, cloud computing platforms have begun to supplement and complement conventional HPC infrastructure to meet the large computing and storage requirements of simulations [16]. Public clouds offer many benefits: on-demand resource allocation, convenient pay-as-you-go pricing models, and ease of deployment on an "infinite" resource pool. An important objective in cloud deployments is to optimize for cost in addition to performance. Costs can be reduced through the use of transient computing resources that can be unilaterally revoked and preempted by the cloud provider, but their preemptible nature results in frequent job failures. The considerations of cost, frequent job failures, and server configuration heterogeneity intrinsic to the system present multiple challenges in deploying applications on cloud platforms. These challenges are fundamentally different from those that appear in using HPC clusters as the execution environment for simulations.

Systems such as SciSpot [17], [18], a framework that uses a new reliability model for constrained preemptions of

Google Preemptible Virtual Machines (VMs), can optimize the deployment of scientific computing applications on transient cloud servers and enable low-cost MD simulations. SciSpot uses an empirical and analytical model of transient server availability to predict expected running times and costs associated with jobs of different types and durations. Considering an entire bag of jobs as an execution unit enables simple and powerful policies for optimizing cost, makespan, and ease of deployment. SciSpot's cost-minimizing server selection and job scheduling policies reduce costs by up to $5\times$ compared to conventional cloud deployments.

III. Molecular Dynamics on ML Systems

The use of ML systems such as TensorFlow and PyTorch has been largely limited for designing ML-based enhancements (e.g., surrogates, integrators, force fields) for MD simulations [6], [12], [13]. Some recent studies have explored the utilization of ML platforms for "non-ML" tasks related to MD simulations [19]–[23]. However, the use of ML systems to develop and execute MD simulations and integrate them with ML-based enhancements is unexplored. In the following, we outline advantages of ML systems as environments for executing MD simulations and integrating them with data-driven models. We also discuss the associated systems challenges.

MD simulations are typically coded in C/C++ and parallelized using OpenMP and MPI [24]. However, more modern MD software packages such as HOOMD-Blue [25] have demonstrated that MD simulations can be easily written in high-level languages such as Python. ML platforms such as TensorFlow are based on Python and the associated high-level data-flow abstraction can enable rapid prototyping of MD simulations. We highlight a few key advantages of utilizing ML systems for executing MD simulations:

- High-performance simulations: ML systems offer automatic parallelization of simulations and enable seamless use of next-generation cloud and HPC hardware such as GPUs and TPUs.
- One-stop platform: ML systems offer extensive support for debugging, data analytics, and post-processing tasks that can be leveraged to perform all simulation-related tasks at a single platform.
- Large ecosystem: Developers have access to a much bigger data science and ML community.
- Better software engineering: Associated tools are richer and are actively developed and improved (compared to, for example, MPI).
- Reproducibility and ease of sharing: Users and developers can easily share all simulation models and methods in one notebook that can be run on ML system backends such as Google Colab (compared to cumbersome configuring of simulations on different HPC systems).
- Integration with data-driven approaches: Data operations in ML systems are first-class operations, instead of being a separate stage of the simulation workflow aimed at exploring the material design space.

The seamless integration of simulations and data-driven models on a single platform offers many opportunities for a rational and expedited exploration of the material design space. We now illustrate a set of these opportunities using ML surrogates as an example. An ML surrogate is a model trained on data from MD simulations that is used to approximate the relationships between the input parameters and the simulation outcomes, bypassing part or all of the explicit evolution of the simulated components. For instance, the ML surrogate in [12] can reduce the inference time from 30 minutes to 0.2 seconds—a $10,000\times$ speedup! This surrogate model was trained using a bag of jobs of size $N = 6,000$, each job representing a unique set of parameters discretizing the high-dimensional input design space.

In the conventional, un-integrated approach, the bag of jobs (of size N) is run sequentially on HPC systems, with a long wait time until a surrogate is trained. N is chosen *a priori* (usually informed by domain expertise) such that the generated data is sufficient to train an ML model (typically a deep neural network). In contrast, an integrated approach utilizing ML systems for simulation facilitates a rapid transition from simulation to surrogate. The surrogate training can now begin with a smaller number n of simulations and the training progress can be monitored in a seamless fashion. When surrogates are designed in an integrated manner on ML systems, a number of new opportunities emerge:

- A unified approach to executing simulations and training ML models to approximate input-output relationships enables the development of the surrogate during the exploration of the material design space with a bag of $n < N$ jobs. ML systems can facilitate the automation of the switch to deriving outputs using surrogates when the training and testing errors become small and surrogate accuracy reaches a high value.
- On-the-fly development of the surrogates enables a principled approach to quantify the completion of the material design space exploration.
- Surrogate accuracy and inferences times can be readily improved with minimal overhead associated with the one-stop platform that enables a seamless accumulation of training data resulting from more simulation executions.
- The ease of designing surrogates in parallel with the simulation-driven exploration of the material design space enables efficient new simulation code development (e.g., writing new pair interaction potentials). The trained surrogates can be used as benchmarks of existing understanding that can guide code updates.

IV. Open Questions and Future Directions

The confluence of ML, MD simulations, and cloud computing presents the broader cloud and HPC research communities with several exciting challenges, and also provides a unique opportunity to bring these communities together. ML-assisted MD simulations impose a unique set of computational requirements which will require advances in cloud resource allocation, such as:

- **Abstractions:** Our bags-of-jobs abstraction is the first step towards a "cloud-native" abstraction for ML-assisted MD workloads. Easier ways to deploy such applications on the cloud will lower costs and will make it easier for domain scientists to rapidly iterate.
- **Rethinking performance metrics:** Conventional metrics such as parallel speedups will be insufficient for ML-assisted workloads that use a combination of training and inference, where the answer can be provided by a trained ML-model. We believe that cost will remain a first-level metric in the cloud, and must be a core part of performance and resource optimizations.

We also claim that ML platforms can provide a single, unified framework for future applications that will integrate ML and MD simulations in new ways. Fundamentally, we are proposing to use systems in ways that they are not designed for, which leads to many natural performance challenges. For instance, the dataflow model used by TensorFlow provides suboptimal performance for fine-grained parallelism required for MD simulations on GPU and TPU clusters, and new performance optimizations and abstractions are necessary.

Finally, bringing a "classic" HPC workload such as MD simulations into an entirely new cloud+ML ecosystem will require the HPC, cloud, and ML communities to work together with domain scientists and engineers in new ways. The confluence will provide opportunities for the next generation of domain scientists to be trained in practical ML and cloud skills. At the same time, cloud researchers should be more cognizant of this new class of workload, which is radically different from the "enterprise" and "big-data" workloads that cloud platforms have traditionally been optimized for.

Acknowledgments V.J. was partially supported by NSF through Award 1720625.

References

[1] R. L. Marson, T. D. Nguyen, and S. C. Glotzer, "Rational design of nanomaterials from assembly and reconfigurability of polymer-tethered nanoparticles," MRS Communications, vol. 5, no. 3, pp. 397–406, 2015. [Online]. Available: https://www.cambridge.org/core/journals/mrs-communications/article/rational-design-of-nanomaterials-from-assembly-and-reconfigurability-of-polymertethered-nanoparticles/3399D7AE6B3C6FD0785663FD21CAB085

[2] M. F. Hagan and R. Zandi, "Recent advances in coarse-grained modeling of virus assembly," Current opinion in virology, vol. 18, p. 36, 2016. [Online]. Available: https://doi.org/10.1016/j.coviro.2016.02.012

[3] J. P. Ewen, D. M. Heyes, and D. Dini, "Advances in nonequilibrium molecular dynamics simulations of lubricants and additives," Friction, pp. 1–38, 2018.

[4] N. Anousheh, F. J. Solis, and V. Jadhao, "Ionic structure and decay length in highly concentrated confined electrolytes," AIP Advances, vol. 10, no. 12, p. 125312, 2020. [Online]. Available: https://aip.scitation.org/doi/10.1063/5.0028003

[5] A. L. Ferguson, "Machine learning and data science in soft materials engineering," Journal of Physics: Condensed Matter, vol. 30, no. 4, p. 043002, 2017.

[6] J. Wang, S. Olsson, C. Wehmeyer, A. Perez, N. E. Charron, G. De Fabritiis, F. Noe, and C. Clementi, "Machine learning of coarse-grained molecular dynamics force fields," ACS central science, 2019.

[7] L. Casalino, A. Dommer, Z. Gaieb, E. P. Barros, T. Sztain, S.-H. Ahn, A. Trifan, A. Brace, A. Bogetti, H. Ma et al., "AI-driven multiscale simulations illuminate mechanisms of sars-cov-2 spike dynamics," bioRxiv, 2020. [Online]. Available: https://www.biorxiv.org/content/early/2020/11/20/2020.11.19.390187

[8] A. Moradzadeh and N. R. Aluru, "Molecular dynamics properties without the full trajectory: A denoising autoencoder network for properties of simple liquids," The journal of physical chemistry letters, vol. 10, no. 24, pp. 7568–7576, 2019. [Online]. Available: https://doi.org/10.1021/acs.jpclett.9b02820

[9] F. Häse, I. Fdez. Galván, A. Aspuru-Guzik, R. Lindh, and M. Vacher, "How machine learning can assist the interpretation of ab initio molecular dynamics simulations and conceptual understanding of chemistry," Chem. Sci., vol. 10, pp. 2298–2307, 2019. [Online]. Available: http://dx.doi.org/10.1039/C8SC04516J

[10] Y. Sun, R. F. DeJaco, and J. I. Siepmann, "Deep neural network learning of complex binary sorption equilibria from molecular simulation data," Chemical science, vol. 10, no. 16, pp. 4377–4388, 2019. [Online]. Available: https://pubs.rsc.org/en/content/articlelanding/2019/sc/c8sc05340e!divAbstract

[11] J. Kadupitiya, G. C. Fox, and V. Jadhao, "Machine learning for parameter auto-tuning in molecular dynamics simulations: Efficient dynamics of ions near polarizable nanoparticles," The International Journal of High Performance Computing Applications, vol. 34, no. 3, pp. 357–374, 2020. [Online]. Available: https://journals.sagepub.com/doi/full/10.1177/1094342019899457

[12] J. Kadupitiya, F. Sun, G. Fox, and V. Jadhao, "Machine learning surrogates for molecular dynamics simulations of soft materials," Journal of Computational Science, p. 101107, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S1877750319310609

[13] J. Kadupitiya, G. C. Fox, and V. Jadhao, "Deep learning based integrators for solving newton's equations with large timesteps," arXiv preprint arXiv:2004.06493, 2021, Under review in Physical Review Research. [Online]. Available: https://arxiv.org/abs/2004.06493

[14] J. Kadupitiya and V. Jadhao, "Probing the rheological properties of liquids under conditions of elastohydrodynamic lubrication using simulations and machine learning," Tribology Letters, vol. 69, no. 3, pp. 1–19, 2021. [Online]. Available: https://doi.org/10.1007/s11249-021-01457-3

[15] J. Kadupitiya, G. C. Fox, and V. Jadhao, "Machine learning for performance enhancement of molecular dynamics simulations," in International Conference on Computational Science, 2019, pp. 116–130.

[16] M. A. S. Netto, R. N. Calheiros, E. R. Rodrigues, R. L. F. Cunha, and R. Buyya, "Hpc cloud for scientific and business applications: Taxonomy, vision, and research challenges," ACM Comput. Surv., vol. 51, no. 1, pp. 8:1–8:29, Jan. 2018. [Online]. Available: http://doi.acm.org/10.1145/3150224

[17] J. Kadupitiya, V. Jadhao, and P. Sharma, "Modeling the temporally constrained preemptions of transient cloud vms," in High-Performance Parallel and Distributed Computing, 2020, pp. 41–52. [Online]. Available: https://dl.acm.org/doi/abs/10.1145/3369583.3392671

[18] "Scispot: Scientific computing on transient cloud servers." [Online]. Available: https://github.com/prateek-s/scispot

[19] 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," Chemical science, vol. 9, no. 8, pp. 2261–2269, 2018.

[20] S. Schoenholz and E. D. Cubuk, "Jax md: a framework for differentiable physics," Advances in Neural Information Processing Systems, vol. 33, 2020.

[21] X. Gao, F. Ramezanghorbani, O. Isayev, J. S. Smith, and A. E. Roitberg, "Torchani: A free and open source pytorch-based deep learning implementation of the ani neural network potentials," Journal of chemical information and modeling, vol. 60, no. 7, pp. 3408–3415, 2020.

[22] S. Doerr, M. Majewski, A. Pérez, A. Krämer, C. Clementi, F. Noe, T. Giorgino, and G. De Fabritiis, "Torchmd: A deep learning framework for molecular simulations," Journal of Chemical Theory and Computation, vol. 0, no. 0, p. null, 0. [Online]. Available: https://doi.org/10.1021/acs.jctc.0c01343

[23] R. Barrett, M. Chakraborty, D. B. Amirkulova, H. A. Gandhi, G. P. Wellawatte, and A. D. White, "Hoomd-tf: Gpu-accelerated, online machine learning in the hoomd-blue molecular dynamics engine," Journal of Open Source Software, vol. 5, no. 51, p. 2367, 2020.

[24] S. Plimpton, "Fast parallel algorithms for short-range molecular dynamics," Journal of Computational Physics, vol. 117, no. 1, pp. 1 – 19, 1995.

[25] J. A. Anderson, J. Glaser, and S. C. Glotzer, "Hoomd-blue: A python package for high-performance molecular dynamics and hard particle monte carlo simulations," Computational Materials Science, vol. 173, p. 109363, 2020.