DEVELOPING A SMART INTEGRATED SIMULATION MODEL OF MOLECULAR DYNAMICS ON CLOUD COMPUTING AND MACHINE LEARNING SYSTEMS

Janhvi Garg

Department of Information Technology, Bharati Vidyapeeth College of Engineering
Affiliated to Guru Gobind Singh Indraprastha University, New Delhi

ABSTRACT

Supercomputers and other high-speed computing facilities have considerably helped scientific computing applications. However, the requirements, architecture, and computational structure of these applications are undergoing a paradigm shift. Molecular dynamics simulations, in particular, are being supported, accelerated, and improved by using data-driven and machine-learning techniques in scientific computing applications. At the same time, cloud computing platforms are becoming more and more attractive for scientific computing because they offer "infinite" processing power, simpler deployment and programming paradigms, and access to computing accelerators like Tensor Processing Units (TPUs). For academics studying clouds and systems, this intersection of cloud computing with machine learning (ML) offers fascinating prospects. Molecular dynamics simulations with machine learning support represent a novel workload category with distinct computational trends. These simulations provide additional difficulties for high-performance, low-cost implementation. Transient cloud resources, such as inexpensive preemptible cloud virtual machines can successfully handle this new burden, can successfully handle this new burden. Lastly, we discuss potential long-term issues, including low-hanging fruit in cloud resource management and the incorporation of molecular dynamics simulations into machine learning systems (like TensorFlow).

INTRODUCTION

Scientific computing applications are crucial to the study and comprehension of a broad range of artificial and natural systems. These applications are mostly used on high-performance computing (HPC) equipment, such as supercomputers, and are usually built as large-scale parallel programs that use communication frameworks like MPI. Among the most common uses of scientific computing are molecular dynamics (MD) simulations.

Materials scientists, chemical engineers, and physicists have made considerable use of these simulations to study the microscopic causes of the macroscopic behaviour of materials such as polymers, electrolytes, viral capsids, self-assembled nanoparticles, and lubricants [1]–[4]. MD simulations aim to investigate the design space related to the material attributes and establish the connections between the design parameters and the material response—typically represented by the structural and dynamical properties—by investigating the design space.

To achieve this, simulations are used as a group or "bag" of tasks. When used in tandem, a bag of tasks "sweeps" a multidimensional design space and provides the connections between the material reaction (outputs) and the material design parameters (inputs). These resources offer a trustworthy manual for conducting experiments to logically identify areas of the material design space that display intriguing structural and dynamical characteristics. The rapidly evolving field

of applying machine learning (ML) to improve MD simulations and speed up the exploration of the material design space likewise heavily relies on the bags of jobs approach [5]–[14]. The ML models intended to improve the prediction capacity or lower the computational costs of MD simulations are trained and tested using large collections of tasks with independent parameter values. Regression algorithms based on artificial neural networks, for instance, can accurately forecast the connections between the input parameters and the simulation results after being trained on data from MD simulations of soft materials [12], [15]. With a 95% accuracy rate and an inference time 10,000x shorter than the comparable MD simulation runtime, these ML surrogates correctly predicted the ion distributions for a range of confined electrolyte systems [12], [15]. To effectively explore the material design space, it will be necessary to run quick and accurate simulations for larger sets of parameters as the value of MD simulations, and its ML-enhanced variants in the rational design of materials is further shown. To do this, it's critical to use various cutting-edge cyberinfrastructure platforms to run inexpensive simulations.

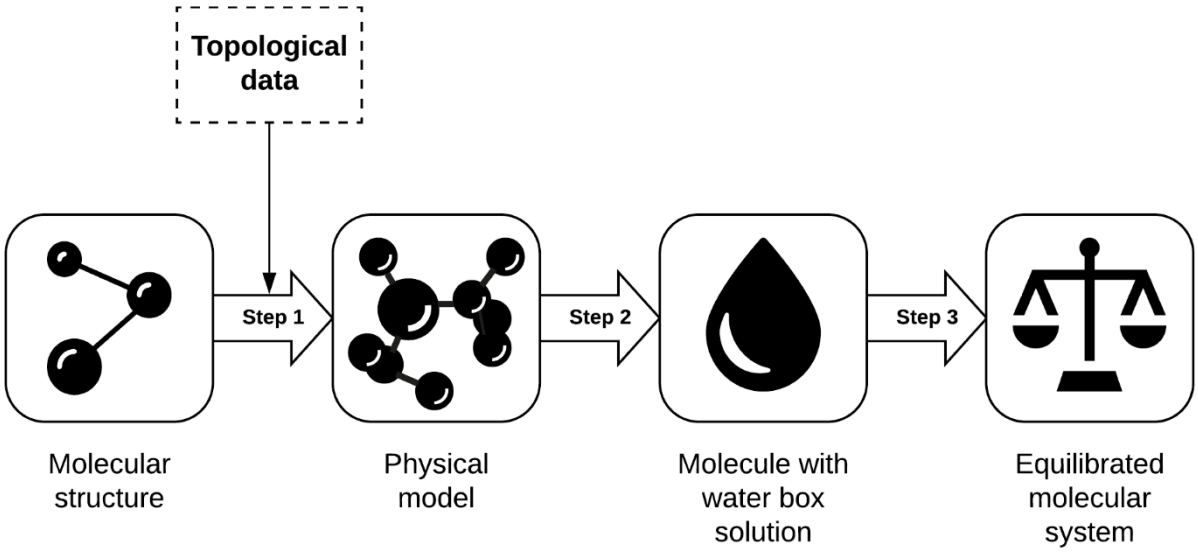

Fig 1: A Web Based Tool for simulating Molecular Dynamics in Cloud Environments

MOLECULAR DYNAMICS ON CLOUD PLATFORMS

To address simulations' high processing and storage demands, cloud computing platforms are increasingly replacing and enhancing traditional HPC infrastructure [16]. On-demand resource allocation, practical pay-as-you-go pricing methods, and simplicity of deployment on an "infinite" resource pool are just a few advantages of public clouds. Optimising cost and performance is a key goal in cloud installations. Transient computing resources, which the cloud provider can unilaterally revoke and pre-empt, can save costs, but because they are preemptible, they frequently cause job failures. Deploying apps on cloud platforms presents several difficulties due to cost, frequent task failures, and server configuration heterogeneity inherent to the system. These difficulties are distinct from those that arise when simulations run in HPC cluster environments.

Low-cost MD simulations can be made possible by systems like SciSpot [17], [18], a framework that optimizes the deployment of scientific computing applications on temporary cloud servers by using a new reliability model for constrained preemptions of Google Preemptible Virtual

Machines (VMs). SciSpot forecasts projected running times and expenses for jobs of various kinds and durations using an empirical and analytical model of transient server availability. When a whole bag of jobs is viewed as an execution unit, straightforward and effective policies for maximizing cost, makespan, and deployment simplicity are made possible. SciSpot's cost-minimizing server selection and job scheduling strategies can save up to five times as much compared to traditional methods.

MOLECULAR DYNAMICS ON ML SYSTEMS

Designing ML-based upgrades (such as surrogates, integrators, and force fields) for MD simulations has mostly limited the usage of ML systems like TensorFlow and PyTorch [6, 12, 13]. The use of ML platforms for "non-ML" activities associated with MD simulations has been investigated in a few recent works [19]–[23]. Nevertheless, there is no research on using ML systems to create and run MD simulations and incorporate ML-based improvements. The benefits of using ML systems as settings for running MD simulations and combining them with data-driven models are described below. The related systems challenges are also covered.

Usually written in C/C++, MD simulations are parallelised with MPI and OpenMP [24]. However, more recent MD software programs like HOOMD-Blue [25] have shown that high-level languages like Python may easily be used to write MD simulations. Rapid MD simulation prototyping is made possible by Python-based ML systems like TensorFlow and the related high-level data-flow abstraction. We point out some significant benefits of using ML systems to run MD simulations:

• High-performance simulations: ML systems facilitate the smooth use of next-generation cloud and HPC hardware, including GPUs and TPUs, by automatically parallelising simulations.

• One-stop platform: All simulation-related operations can be completed on a single platform by utilising the comprehensive support that machine learning systems provide for debugging, data analytics, and post-processing chores.

• Vast ecosystem: The data science and machine learning community is significantly larger for developers to access.

• Better software engineering: Compared to MPI, related tools are more comprehensive and constantly being created and enhanced.

• Reproducibility and ease of sharing: Unlike the laborious process of configuring simulations on many HPC systems, users and developers can effortlessly share all simulation models and methods in a single notebook that can be executed on ML system backends like Google Colab.

• Integration with data-driven approaches: Rather than being a distinct step of the simulation workflow meant to explore the material design space, data operations in machine learning systems are first-class operations.

The smooth integration of simulations and data-driven models on a single platform presents numerous options for a logical and quick exploration of the material design space. We now use ML surrogates as an example to demonstrate a few of these options. Bypassing some or all of the

explicit evolution of the simulated components, an ML surrogate is a model trained on data from MD simulations and is used to approximate the relationships between the input parameters and the simulation results. For example, the ML surrogate in [12] can speed up inference from 30 minutes to 0.2 seconds—a 10,000× speedup! To train this surrogate model, a bag of tasks of size N = 6,000, each representing a distinct set of parameters discretising the high-dimensional input design space, was used.

The traditional, unintegrated method involves a lengthy wait period until a surrogate is trained, and the bag of jobs (of size N) is executed one after the other on HPC computers. N is predetermined (often based on domain knowledge), so there is enough generated data to train an ML model (generally a deep neural network). On the other hand, an integrated method that uses ML systems for simulation makes a seamless transition from simulation to surrogate possible. This allows for the start of surrogate training with fewer simulations and seamless tracking of training progress. Several additional options arise when surrogates are developed using ML systems in an integrated manner:

The development of the surrogate throughout the exploration of the material design space with a bag of n < N jobs is made possible by a unified method for running simulations and training machine learning models to approximate input-output correlations. Machine learning (ML) systems can help automate the transition to employing surrogates to derive outputs when testing and training mistakes are minimal and surrogate accuracy is high.

• A principled method to measure the completion of the material design space exploration is made possible by the surrogates' on-the-fly development.

• With the one-stop platform, which facilitates the smooth accumulation of training data from further simulation iterations, surrogate accuracy and inference times can be easily increased with little overhead.

• Efficient development of new simulation code (e.g., developing new pair interaction potentials) is made possible by the simplicity of surrogate design in conjunction with simulation-driven exploration of the material design space. The trained surrogates can guide code updates, which can serve as benchmarks of current knowledge.

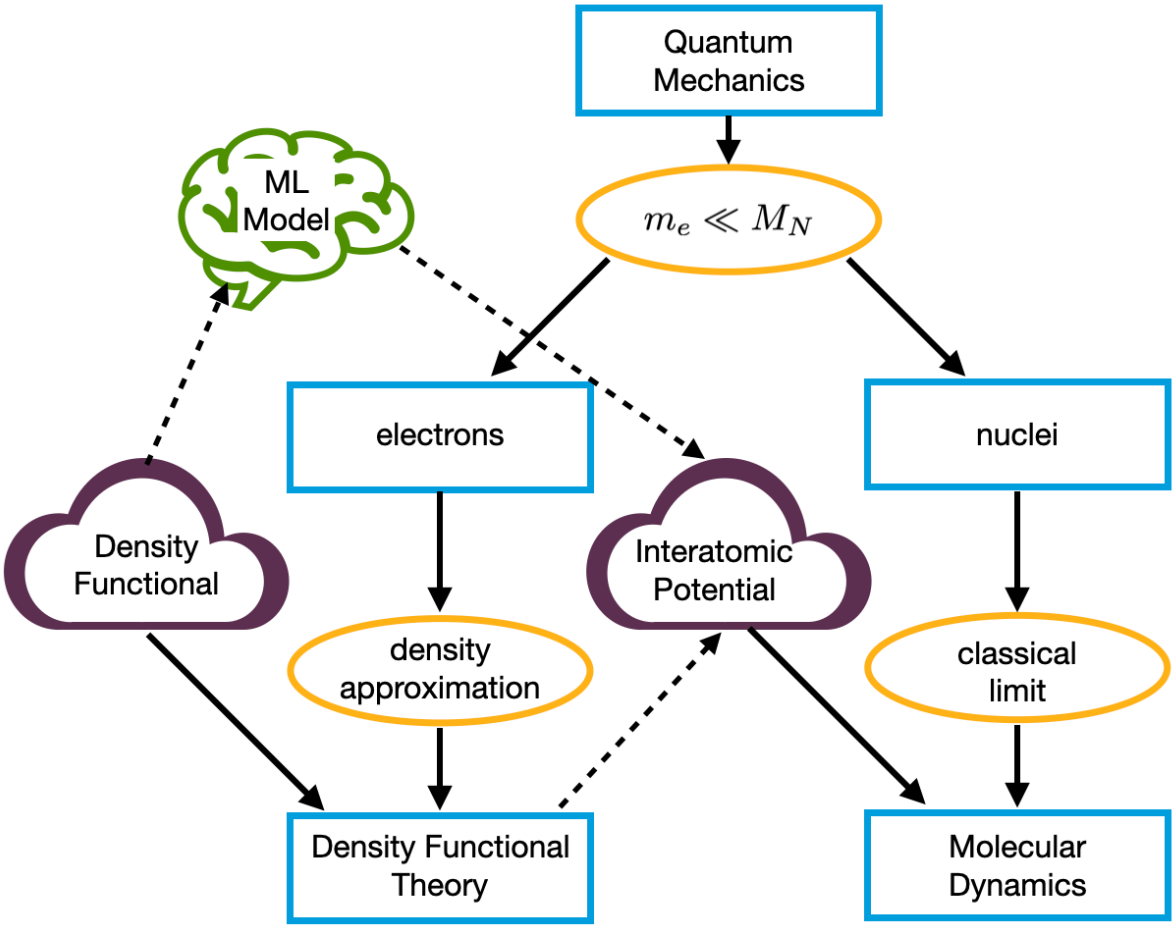

Fig 2: Molecular Simulation using Machine learning

CONCLUSION AND FUTURE DIRECTIONS

Along with offering a special chance to unite these communities, the intersection of machine learning, MD simulations, and cloud computing poses several fascinating challenges to the larger cloud and HPC research communities. The computational demands of ML-assisted MD simulations are distinct and will necessitate advancements in cloud resource allocation, including:

Abstractions: Our bags-of-jobs abstraction is the first step towards a "cloud-native" abstraction for ML-assisted MD workloads. Simpler cloud deployment methods will reduce expenses and facilitate quick iterations for domain scientists.

• Rethinking performance measurements: For ML-assisted workloads that include training and inference, where a trained ML model can supply the solution, traditional metrics like parallel speedups will not be adequate. In our opinion, cost will continue to be a key factor in cloud computing and must be considered when optimising performance and resources.

Additionally, we assert that ML platforms can offer a single, cohesive foundation for upcoming applications that will creatively combine MD and ML simulations. In essence, we suggest that systems be used in ways they were not intended, which results in numerous inherent performance issues.

For example, TensorFlow's dataflow model performs below optimally regarding the fine-grained parallelism needed for MD simulations on GPU and TPU clusters; therefore, additional performance optimisations and abstractions are required.

Lastly, integrating a "classic" HPC workload like MD simulations into a completely new cloud+ML ecosystem would necessitate new collaborations between the cloud, ML, and HPC communities, domain scientists and engineers. The confluence will offer chances to train the upcoming generation of domain scientists in useful cloud and machine learning techniques. Cloud researchers should also be more aware of this new workload category, which differs greatly from the "enterprise" and "big-data" workloads for which cloud platforms have historically been designed.

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/mrscommunications/ article/rational-design-of-nanomaterialsfrom- assembly-and-reconfigurability-of-polymertetherednanoparticles/3399D7AE6B3C6FD0785663FD21CAB085

[2] M. F. Hagan and R. Zandi, "Recent advances in coarse-grained modelling 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 sarscov- 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. H¨ase, I. Fdez. Galv´an, 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, "Modelling 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. P´erez, A. Kr¨amer, 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.