18.337/6.338 Final Project Proposal

Jeremiah DeGreeff — Spring 2022

InteratomicPotentials.jl is a package being developed as a part of CESMIX. The goal of the package is to define a common interface for potentials between atoms and molecules as well as to implement a handful of basic empirical potentials that implement this interface. Up to this point, the developers of this package have focused on interface design, feature implementation, and correctness without a lot of emphasis on performance and parallelism. This project will involve enhancing the performance of this package through a variety of methods.

The first goal of the project is to performance engineer the existing code. In order to do this, I will establish a collection of benchmark example systems and collect rich profile data on the current state of the package's performance. Depending on the findings at this stage, I might invest some time on serial optimizations or I might jump straight into parallelization. I plan to explore various sources of parallelism and consider multiple options for taking advantage of these, but I particularly anticipate that GPU acceleration will be the most beneficial. This is of particular research interest because one of the main consumers of InteratomicPotentials.jl is Atomistic.jl which is an interface for molecular dynamics simulations. The latter package includes support for Molly.jl as a backend which already has full GPU support, so adding GPU support to InteratomicPotentials.jl could potentially allow full MD time steps to take place on the GPU.

The second part of this project involves investigating the usage of neighbor lists in calculating pairwise potentials between all bodies in a system. Currently InteratomicPotentials.jl contains a simple implementation of this feature that has not been significantly analyzed for performance. I plan to research any literature that discusses this family of algorithms to see what trade offs exist and also how various solutions scale. If there are multiple viable solutions, I might implement them as a user-configurable option. There are a couple of options implemented in Molly.jl that may be of interest. One detail that is of particular interest is whether an iterative approach that reuses an approximate neighbor list from a previous timestep in an MD simulation might be appropriate. Any solutions that I develop for this component will emphasize performance on large-scale datasets as well as ease of use from an interface perspective.

The scope of this project is relatively large but feels reasonable given that I plan to work on it for the better part of a month. If I end up being limited in time, I will prioritize the first half of the project and cover the neighbor lists as time allows (with the plan to continue that work after the term ends if necessary). Though also of great research interest, performance engineering for the related package InteratomicBasisPotentials.jl is outside the scope of this project, though I hope that this work can inform the future improvements to that package as well.