Co-Design for Molecular Dynamics: An Exascale Proxy Application

Jamaludin Mohd-Yusof,
Sriram Swaminarayan, CCS-7;
Timothy C. Germann, T-1

Roadrunner demonstrated the need to understand the interaction between high-performance hardware and software as early in the development process as possible. One mechanism being used to explore this interaction is the notion of proxy applications that provide a means to test both hardware and software modifications while retaining the essential workload of a real application. Co-designed Molecular Dynamics (CoMD) is one of the proxy "apps" being employed for this purpose by the Exascale Co-Design Center for Materials in Extreme Environments (ExMatEx). We present examples of the options available within CoMD and their effects, as well as potential impacts of future hardware modifications.

Molecular dynamics (MD) simulations represent a significant fraction of the DOE workload, as they provide the fidelity required to examine materials' response to extreme conditions. Examples of interest include nuclear reactor lifetime extension and nuclear stockpile aging. Achieving the fidelity required to simulate these problems at scale will continue to require the largest computational resources available.

High-performance computing (HPC) is currently undergoing a transition period where a variety of new architectures are being explored. Roadrunner was the first example of a hybrid supercomputer, and required a huge effort to enable codes to run on it [1]. The options available for future exascale machines include accelerators derived from graphics processing units (GPUs), many-core accelerators such as the Intel Many-Integrated-Core (MIC) architecture, and evolutions from existing CPU architectures from AMD, Intel, and IBM. Each of these hardware choices presents tradeoffs in terms of their relative performance when running scientific computing algorithms. These algorithms, in turn, represent varying workloads to the machine, in terms of both intra-node and inter-node requirements.

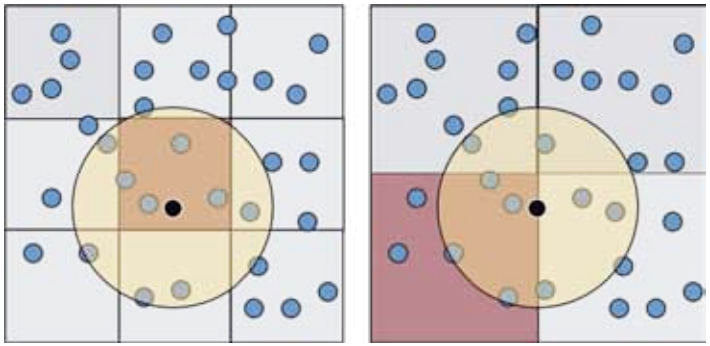

Fig. 1. Sketch showing an example of link-cell data decomposition. The link cells are larger than the cutoff radius for the interatomic potential, so that a search of neighboring cells covers all particles within the cutoff. The tuning parameter box factor is the ratio of the link cell size to the cutoff and can be varied to tune the granularity of the data decomposition. The left sketch shows a box factor of 1, while the right shows the same data decomposed with box factor 1.5.

ExMatEx is one of several efforts to develop a framework in which these hardware-software interactions can be explored. This will enable both the hardware vendors and software developers to co-optimize their products to ensure that future machines are able to provide the performance needed to solve these challenging problems. Part of this strategy uses proxy applications ("proxy apps") that encapsulate the workload of an actual science application. These simplified applications are more amenable to testing and analysis than existing production applications and are available to external collaborators.

The CoMD proxy app represents the typical workload and use cases of MD simulations of material dynamics. We expect that algorithmic improvements and optimizations will ultimately be incorporated into DOE production MD codes such as LAMMPS, ddcMD, and SPaSM. CoMD represents the fundamental workflow in such simulations from problem setup, equilibration, time integration, analysis/visualization, to checkpoint/restart.

For the development of CoMD, the SPaSM code was chosen as a starting point because the code transformations made to port SPaSM to Roadrunner form a good basis for those needed to optimize an exascale code. In particular, the separation of local work and inter-node communication to reduce latency provide a framework that allows us to concentrate on optimizing the local (intra-node) performance.

MD solves Newton's laws of motion for individual atoms–for short-range interatomic potentials typical of metals and other neutral, non-ionic systems this requires the evaluation of an interatomic force on each atom arising from all neighboring atoms within a potential-dependent cutoff radius. For our targeted class of materials (metals) that radius typically corresponds to a few interatomic spacings within a lattice, so only 10-100 atoms are within the cutoff. This informs the choice of domain decomposition (spatial) and the use of a link-cell formulation for ordering particles and iterating particle pairs (see Fig. 1). In typical production runs, approximately 95% of the simulation time (aside from checkpoint I/O and in situ analysis/visualization, which vary widely depending on needs) is spent computing the forces on atoms, giving a clear hot-spot for optimization efforts.

In order to explore the tradeoffs discussed earlier, various options are incorporated into CoMD.

- **Execution models.** The base version of CoMD is a simple serial code and provides the reference for all other variants. OpenCL and OpenMP versions are also provided to allow testing on GPUs and multi/many-core processors.

For more information contact Jamaludin Mohd-Yusof at jamal@lanl.gov.

Funding Acknowledgments
DOE, Office of Science, Advanced Scientific Computing Research Program, ExMatEx Project

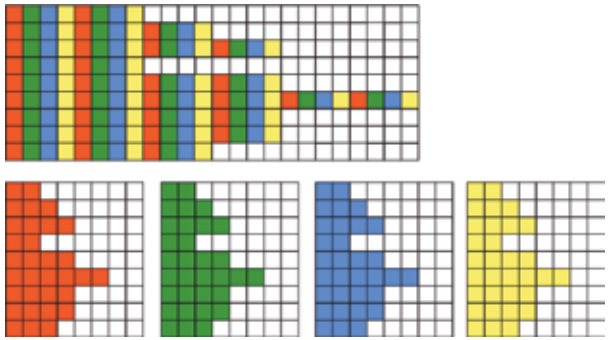

Fig. 2. Sketch of AoS, (top) and SoA, (bottom) data layouts. The different colors represent different data components, for example, the x, y, and z components of particle position. In the CoMD code, typical AoS data would include the three components of particle position and the mass. Each data layout provides different benefits in terms of data locality and vectorizability of operations, among other factors.

• **Data Layout.** The natural way to think of particle data is as a structure–for example, the position, velocity, and force each have three components [x, y, z]. In memory, we can group these three components together to form an Array-of-Structures (AoS, Fig. 2a). Alternatively, we could separate the x-component data (for all the particles) in contiguous arrays, forming a Structure-of-Arrays (SoA, Fig. 2b). The base version of CoMD utilizes an AoS data layout that mimics that in SPaSM. The OpenCL version has both AoS and SoA data layouts to evaluate the relative performance on a variety of architectures. An example of the importance of data layout is shown in Fig. 3.

• **Potential Representation.** For metals, Embedded Atom Method (EAM) tabulated data is traditionally used to represent the interatomic potential. These tables are too large to fit into the shared cache of many newer architectures, so we provide the option to use polynomial approximations which require a small (user-selectable) number of coefficients but increase arithmetic intensity.

• **Data Decomposition.** Depending on the arithmetic intensity of the algorithm, the performance of the code may be determined by the ability to access memory efficiently. We provide a "box factor," the ratio of link cell size to the cutoff radius (see Fig. 1), as a tuning parameter. This makes more efficient use of loads from main memory and also reduces the frequency of re-sorting particles between link cells, which can itself be a bottleneck in some situations.

Although it is helpful to measure the performance differences between different implementations on current hardware, we also need to understand the effects that hardware changes may have on performance. Aspen [2] is a framework for the analytical modeling of exascale applications and architectures that is based on a domain-specific language. The Aspen developers at ORNL constructed a model of CoMD and used it to predict the effect of varying cache size on a GPU, in this case an Nvidia Tesla M 2090, as shown in Fig. 4.

A significant component of the co-design loop is to engage vendors so that changes to future hardware can be incorporated when feasible. CoMD is being used by NVIDIA, AMD, IBM and Intel as a representative workload to study the performance impacts of various design tradeoffs in future processors.

CoMD is one of a suite of proxy apps being developed as part of the ExMatEx project. It provides a variety of options, with more under development, to allow hardware and software developers to understand the tradeoffs needed for future exascale applications. Additional information can be found at https://github.com/exmatex.

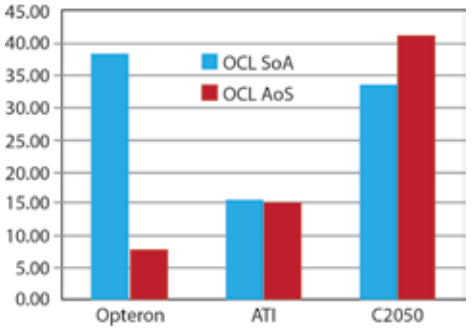

Fig. 3. An example of the effect of data layout choices on application performance. The same OpenCL code is run on an AMD Opteron CPU, ATI Cypress GPU and Nvidia C2050 GPU, using AoS or SoA data layouts. The Opteron performs poorly on the AoS layout. The GPUs show less sensitivity, but differing trends with respect to data layout. The differences may also be attributable to the OpenCL compiler provided by each vendor, showing the importance of the entire hardware/software ecosystem on performance.

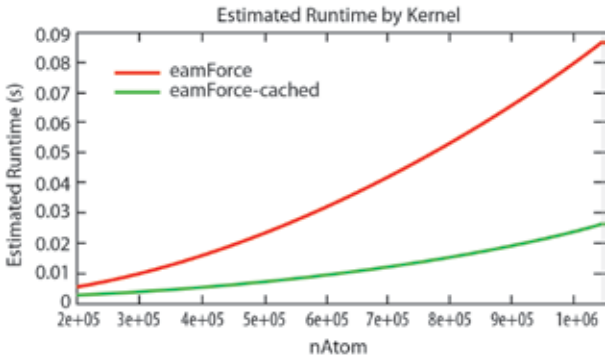

Fig. 4. Predicted performance improvement on an Nvidia M2090 GPU for EAM table lookup, if the cache expanded to hold all the table data. Such predictions can provide important insight for vendors when designing future architectures, as well as for software developers seeking to maximally exploit a given architecture.

[1] *First Science at the Petascale: Results from the Roadrunner Supercomputer*, October 2010. LA-UR-10-06728

[2] Spafford, K. and Vetter, J.S., "Aspen: A domain Specific Language for Performance Modeling," *SC12: ACM/IEEE International Conference for HPC, Networking, Stoage, and Analysis*, 2012.