Extending Parallel Scalability of LAMMPS and Multiscale Reactive Molecular Simulations

Yuxing Peng, Chris Knight, Philip Blood, Lonnie Crosby, and Gregory A. Voth
Outline

• Proton Solvation and Transport and Multiscale Reactive Molecular Dynamics
• Reactive MD Challenges
• Choice of New Code
• Parallel Strategies
• Project Challenges
• Results
• Future Work
Proton Solvation and Transport

Unraveling proton transport pathways in chloride transport membrane proteins.

Electron-coupled proton transport in Cytochrome c Oxidase

Wang and Voth, Biophys. J, 97, 121 (2009)

A linear combination of bonding topologies can be used to describe the variable bond topology of a reactive complex.

\[
\mathbf{H} = \begin{pmatrix}
V_{11} & V_{12} & V_{13} & V_{14} \\
V_{12} & V_{22} & 0 & 0 \\
V_{13} & 0 & V_{33} & 0 \\
V_{14} & 0 & 0 & V_{44}
\end{pmatrix}
\]

All interactions are parameterized by fitting to QM or AIMD data.
Algorithmic Challenges and Parallel Scaling

- Efficiently calculate Hamiltonian matrix
- Complex is all possible bond topologies (states) for a given proton.
- As many as 30+ states in a given complex
  - environment — environment
  - complex — environment
  - complex — complex
- Poor parallel scaling (DL_EVB)

$$H = \begin{pmatrix}
V_{11} & V_{12} & V_{13} & V_{14} \\
V_{12} & V_{22} & 0 & 0 \\
V_{13} & 0 & V_{33} & 0 \\
V_{14} & 0 & 0 & V_{44}
\end{pmatrix}$$

![Graph showing parallel scaling with different atom counts](image)
Parallel Scaling Challenges

Ranger (TACC)

- Real-space interactions scale near perfectly (work is local to processor).
- Parallel performance degradation at high processor counts largely because of 3D FFTs (many tens of FFTs per time step).
- Critical to multistate algorithms, which evaluate several 3D FFTs per matrix element.
- Without affecting accuracy, a more efficient parallelization strategy is required.

\[ H = \begin{pmatrix}
V_{11} & V_{12} & V_{13} & V_{14} \\
V_{12} & V_{22} & 0 & 0 \\
V_{13} & 0 & V_{33} & 0 \\
V_{14} & 0 & 0 & V_{44}
\end{pmatrix} \]
Choosing a New Code

• **Old Code**: Reactive MD method implemented in DL_POLY code (DL_EVB)
  – Slow serial and parallel execution
  – Very easy to modify and extend

• **New Code requirements**
  – Reasonably fast
  – Easily modifiable and extensible by graduate students and postdocs
  – Suitable for accurate all-atom biomolecular simulation

• **Benchmarked and compared codes**
  – Gromacs and NAMD very fast, but harder to extend
  – LAMMPS ~2x slower, but simple to modify and extend due to modular design (uses C++ for high-level organization, plain C for low level stuff)
  – Ease of use and maintainability outweighed performance difference
Implementing Reactive MD in LAMMPS

• RAPTOR (Rapid Approach for Proton Transport and Other Reactions)

• Multistate algorithm is written as “fancy” potential and interfaced with LAMMPS through the “fix” mechanism.

• Immediately observed improvements in parallel scaling efficiency, but additional work was required.

Ranger (TACC): 512 Waters + Proton for DL_EVB (black) and RAPTOR (red)

Cytochrome c Oxidase (159K atoms)
Improving Parallel Performance: Hybrid Strategy

• Tested new hybrid OpenMP/MPI LAMMPS developed by Axel Kohlmeyer
• Minimize MPI processes involved in all-to-all communication
• Added capability of using **single precision FFT** in LAMMPS to reduce required communication bandwidth
• Observed good speedup on Kraken
Multiple Program Strategy (R- and K-space)

- Used by Gromacs and NAMD

- Divide processors into (at least) two separate partitions.

  - 1st partition, usually larger, handles real-space forces, equations of motion integration, I/O, etc...

  - 2nd partition handles only 3D FFTs

- Reduced communication and simultaneous evaluation lead to sizeable improvements in performance.

*Implemented as “run_style verlet/split” in LAMMPS*
Multiple Program vs. Hybrid Strategy

- Multiple Program wins out – with proper rank placement
Multiple Program Strategy (R- and K-space)

- At higher processors counts, important to keep MPI ranks within comm. block proximal to each other on physical machine.

- Default assignment of MPI rank order is suitable for small core counts.

- Reordering typically becomes important beyond several nodes.

- Major Breakthrough: integration into main LAMMPS distribution

LAMMPS mechanism for MPI rank reordering: “-reorder nth 4”
Project challenges

• Multi-year ASRT project challenges
  – Sporadic work due to finely divided time commitments
  – Restart overhead
  – Keeping track of project progress, ideas, code versions, benchmark parameters, etc.
  – Eventually used Google docs to standardize data sharing, keep track of results

• Machine-specific issues
  – Compilers (compiler and machine-specific bugs)
  – Performance tools (work with some compilers, not others)
  – Reproducible and self-consistent benchmarks (using same set of nodes for a given set of benchmarks)
Multiple Program Scaling- LAMMPS

- Benchmark: 159K atom CcO system
- Benefits of MP parallelization at 128 nodes and beyond.
- A 3:1 MP ratio was found to be optimal, with larger ratios better at larger processor counts.
- Additional speedups observed when MP is combined with OpenMP threads.
Multiple Program Scaling- RAPTOR

• Benchmark: 159K atom CcO system
• Improvements from MP observed by 32 cores.
• Reactive simulations are only ~2x slower than nonreactive simulations!!
• Scaling analysis identifies real-space partition waiting for k-space partition to finish.
• Total speedup: 3-4x faster than Single Program RAPTOR
Key highlights for final project year

• Improved scaling of the general LAMMPS MD code

• Incorporated improvements into the main LAMMPS code base, benefiting the world-wide LAMMPS community. ([http://lammps.sandia.gov/](http://lammps.sandia.gov/))

• For the reactive LAMMPS code (Raptor), increased the accessible simulation time by **4x (target was 2x)**, while maintaining parallel efficiency over 50%.

• Optimized and benchmarked the code on two XSEDE platforms: Kraken and Ranger.

• Results published in XSEDE12 paper: [http://dl.acm.org/citation.cfm?id=2335833](http://dl.acm.org/citation.cfm?id=2335833)

• Overall improvement over previous reactive code (DL_EVB) is 20-30x
Future Directions…

• Finish updating support for OpenMP in RAPTOR with MP.

• Approximate methods for electrostatics have been developed to greatly reduce the number of 3D-FFTs in multistate algorithms.

• Generalize MP algorithm in RAPTOR to arbitrary number of partitions for single and multiple reactive species. Parallelize over states.

• Address load balancing between partitions (parallelization over independent states may help overlap computation and communication).

• Further optimize electrostatics and FFTs.
Load Imbalance

- Used the fpmpi lightweight profiling tool to obtain information about RAPTOR runs.
- MPI Synchronization time become significant portion of run time.
Source of Synchronization Time

• 64 cores single partition
  – Avg. MPI Sync Time = 56.69 s (206.44 s Wall Clock)
    • Avg. MPI_Wait + MPI_Waitany = 49.56 s
    • Avg. Collective Sync = 3.10 s

• 64 cores two partitions (32:32 cores)
  – Avg. MPI Sync Time = 86.84 s (191.19 s Wall Clock)
    • Avg. MPI_Wait + MPI_Waitany = 23.05 s
    • Avg. Collective Sync = 3.05 s
    • Avg. MPI_Bcast = 58.71 s
      – Tentatively attributed to bcast which transfers data from k-space to r-space partitions.