

Boosting molecular dynamics with advanced hardware and algorithms

Lei Huang

XSEDE ECSS Symposium

Feb 16, 2016



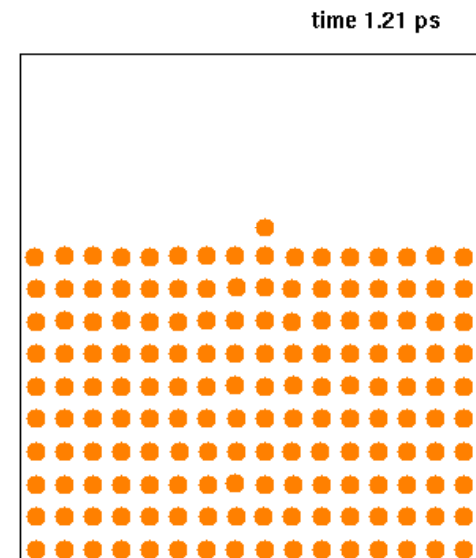
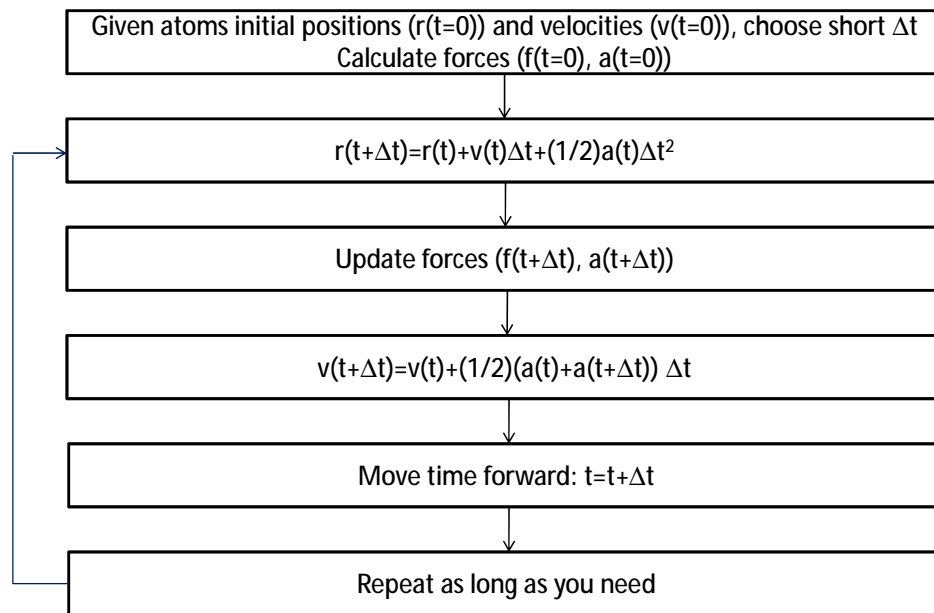
THE UNIVERSITY OF TEXAS AT AUSTIN
TEXAS ADVANCED COMPUTING CENTER

Outline

- Background
- Optimization we have made
 - Buffered I/O
 - Parallelization for bonded energy on host
 - SHAKE algorithm to enable larger integration time step (2fs)
 - Particle mesh Ewald (PME) method for long-range electrostatic interactions
 - Asynchronous MIC offload to utilize host more efficiently
 - An improved criterion to rebuild neighbor list
 - Hybrid code (MPI and OpenMP) to utilize more than one node
- Summary

Background

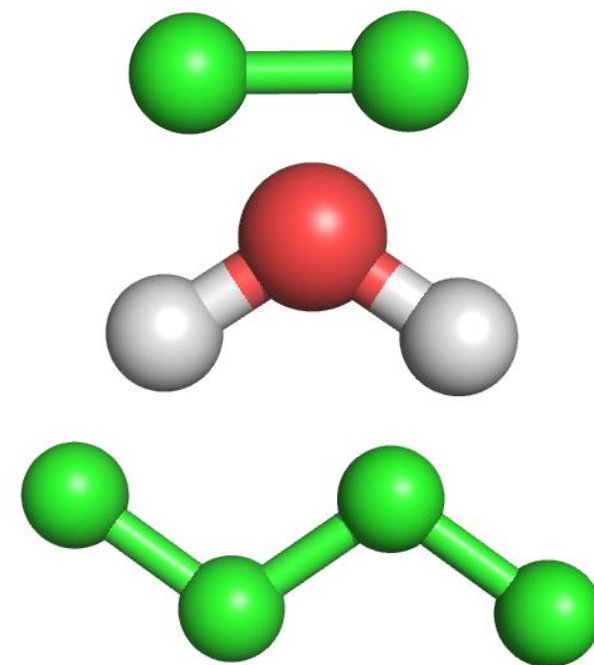
- Work flow of molecular dynamics simulation



https://en.wikipedia.org/wiki/Molecular_dynamics

Hamiltonian

$$\begin{aligned}
 E = & \sum_{\text{bonds}} K_b (b - b_0)^2 + \sum_{\text{angles}} K_q (q - q_0)^2 + \\
 & + \sum_{\text{dihedrals}} K_f (1 + \cos(nf - d)) + \sum_{\text{improper dihedrals}} K_j (1 + \cos(nj - j_0)) \\
 & + \sum_{\text{nonbonded}} \frac{q_i q_j}{4\pi\epsilon_0 r_{ij}} + e_{ij} \left[\left(\frac{R_{\min,ij}}{r_{ij}} \right)^{12} - 2 \left(\frac{R_{\min,ij}}{r_{ij}} \right)^6 \right]
 \end{aligned}$$



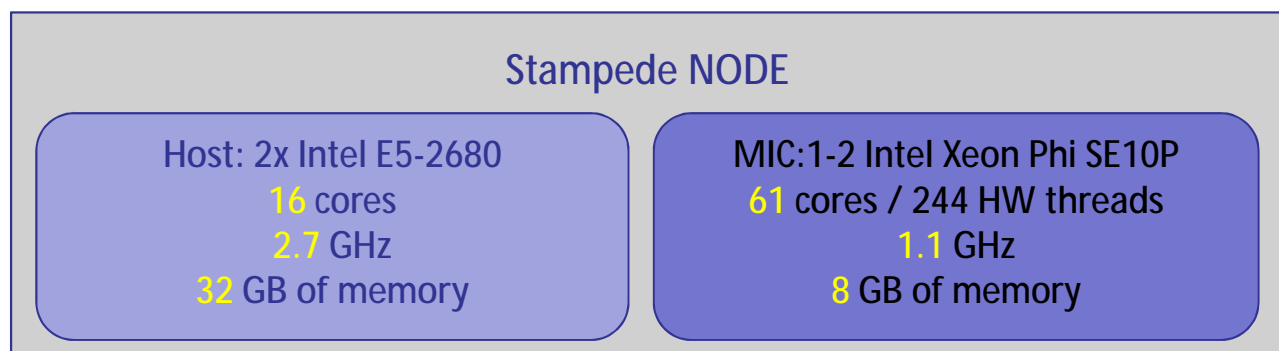
Force and acceleration

$$F(r_i) = -\frac{\partial E}{\partial r_i} \qquad a_i = \frac{F_i}{m_i}$$

A typical node on Stampede

A “node” contains a host component and a MIC component

- Host – refers to the Sandy Bridge component
- MIC – refers to one or two Intel Xeon Phi co-processor cards



0.35 tera-FLOPS DP

~ 1 tera-FLOPS DP

MIC Modules on Stampede

Software package	Module name
NAMD: MD simulation	namd/2.10-mic_mpi
LAMMPS: MD simulation	lammps/10Feb15
NetCDF4: file i/o	netcdf/4.3.3.1
NetCDF4: parallel i/o	parallel-netcdf/4.3.2
HDF5: file i/o	hdf5/1.8.16
HDF5: parallel i/o	phdf5/1.8.16
DDT: debugger	ddt_mic/4.2.1
VTune: profiling	vtune/13.update14
PAPI: profiling	papi/5.3.0
ITAC: profiling	itac/9.0.3
Boost: C++ libraries	boost/1.55.0
GSL: C++ libraries	gsl/1.16
zlib: compression library	zlib/1.2.8

Status

of the MD engine for crystalization from Dr. Doraiswami Ramkrishna's group

- Parallelized with OpenMP
- Synchronized MIC offload
- Too much time spent on I/O
- 1 fs integration time step
- Damped-shifted force was used for long range electrostatic interactions (Needs large cutoff, inferior accuracy)

Optimization we have made

- Buffered I/O
- Task parallelization for bonded energy on host
- SHAKE algorithm to enable using larger integration time step (2fs)
- Particle mesh Ewald (PME) method for long-range electrostatic interactions
- Asynchronous MIC offload to utilize host more efficiently
- An improved criterion to rebuild neighbor list
- Hybrid code (MPI and OpenMP) to utilize more than one node

Buffered I/O with Intel Fortran compiler

- Unbuffered I/O is used by default
- To enable buffered I/O
 - Flag during compiling: `ifort -assume buffered_io ...`
 - Runtime environment variable: `export FORT_BUFFERED=true`

Performance is improved ~40%.

Parallelization of bonded energy on host

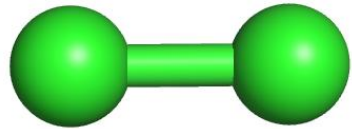
$$E_{bonded} = \sum_{\text{bonds}} K_b (b - b_0)^2 + \sum_{\text{angles}} K_q (q - q_0)^2 + \\ + \sum_{\text{dihedrals}} K_f (1 + \cos(nf - d)) + \sum_{\text{improper dihedrals}} K_j (1 + \cos(nj - j_0))$$

And several correction terms

Task parallelism in OpenMP to tackle individual bonded terms separately

Performance is improved ~10%.

Introducing SHAKE algorithm to enable larger integration time step



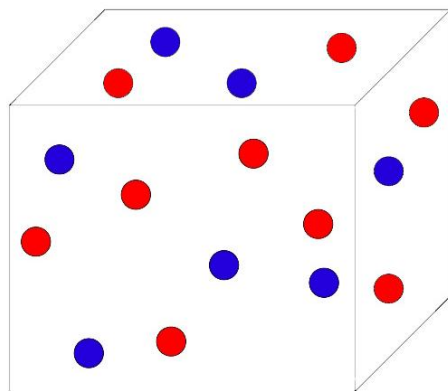
1 fs is needed for flexible bonds with hydrogen atoms involved

With SHAKE algorithm, such bonds are fixed. 2fs is feasible to boost simulations.

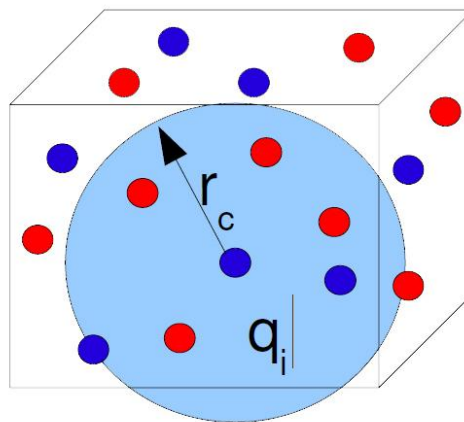
Performance is improved 98%.

Long-range electrostatic interactions

Consider a box of N charged particles with periodic boundary conditions and volume L^3 :



Truncate potential at distance r_c



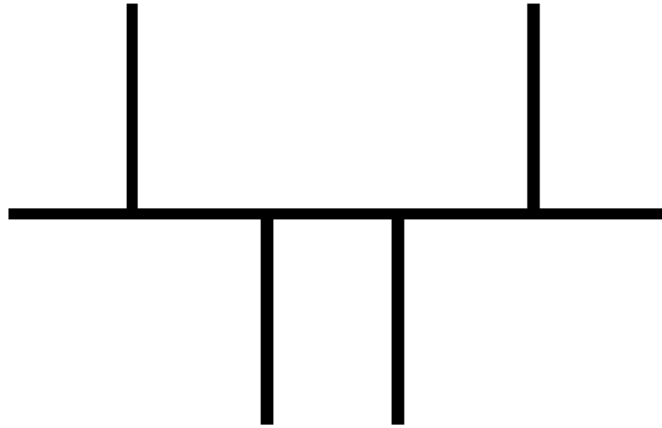
$$U_{tail} = \frac{Nr}{2} \int_{r_c}^{\infty} dr U(r) 4\pi r^2$$

How to calculate the overall coulomb potential?

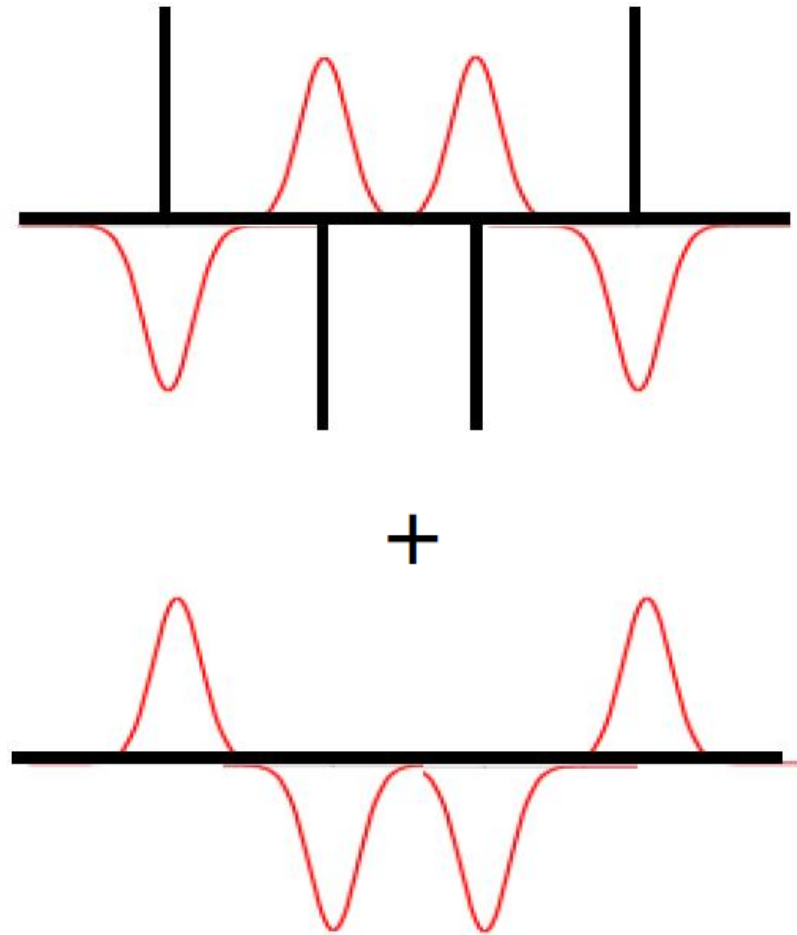
Tail correction diverges, unless $U(r)$ decays faster than r^{-3} . Not suitable for Coulomb potential (r^{-1})

Ewald summation

by Paul Peter Ewald in 1921



=



$$\begin{aligned}
\mathcal{U}_{\text{Coul}} = & \frac{1}{2V} \sum_{\mathbf{k} \neq 0} \frac{4\pi}{k^2} |\rho(\mathbf{k})|^2 \exp(-k^2/4\alpha) \\
& - (\alpha/\pi)^{\frac{1}{2}} \sum_{i=1}^N q_i^2 \\
& + \frac{1}{2} \sum_{i \neq j}^N \frac{q_i q_j \operatorname{erfc}(\sqrt{\alpha} r_{ij})}{r_{ij}}.
\end{aligned}$$

Standard Ewald implementation scales like N^2

The most expensive part: Fourier transformation

Using Fast Fourier Transformation (FFT) leads to speedup

3 popular algorithms, that implement FFT:

1) Particle-particle particle mesh (P3M)

2) Particle mesh Ewald (PME)

3) Smooth particle mesh Ewald (SPME)

scales like $N \log(N)$

Develop a PME library

- Package MDX
(<http://www.ks.uiuc.edu/Development/MDTools/mdx/>) has an serial implementation of PME algorithm
- Parallelization with OpenMP for four functions (“hot spots”)

Performance is improved 92%.

Asynchronous MIC offload

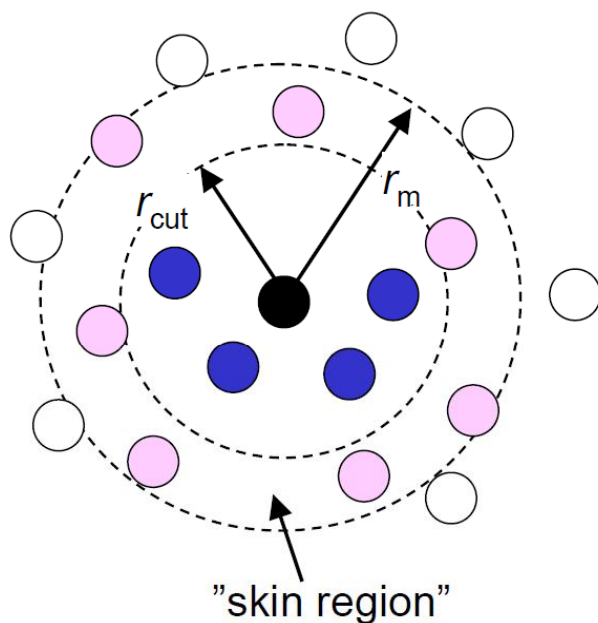
```
integer :: sig=1
```

```
!dir$ offload target(mic:ourmic), signal(sig),
```

```
!dir$ offload_wait target(mic:ourmic) wait(sig)
```

Performance is improved 54%.

Improving building neighbor list



Rebuild neighbor list when any atom has moved a distance larger than $|r_m - r_{\text{cut}}|$ since last neighbor list building

Performance is improved 15%.

A lesson from running MIC offload MD code

Issue: “numactl --cpubind=0” imposes large performance penalty (>30%) compared the case without numactl for socket binding when the task is using all cores of one socket

The issue is gone if we leave at least one core on the socket idle. The available CPU resource is very important for the thread dedicating for the asynchronous MIC offload!

Summary

Issues	Solution	Speedup	Performance (fs/second)
			Original code, 4.59
Unbuffered I/O	Enable buffered I/O export FORT_BUFFERED=true	40%	6.43
Bonded forces were not efficiently parallelized	Task parallelism to tackle different bonded terms separately	10%	7.07
1fs time step	SHAKE algorithm to use 2fs integration time step	98%	14.00
Damped-shifted force algorithm	A PME library was developed and parallelized with OpenMP.	92%	26.84
The CPUs on host were idle due to synchronous offload	Implemented asynchronous MIC offload	54%	41.38
The rule to rebuild neighbor list.	A correct one was introduced	15%	47.48

Performance is improved ~9x.

Acknowledgement

- Dr. Doraiswami Ramkrishna (@ Purdue)
- Conor Park (@ Purdue)
- Dr. Yang Wang (@ PSC)
- Dr. Si Liu (@ TACC)

Thank you
for your attention!

Questions?



THE UNIVERSITY OF TEXAS AT AUSTIN
TEXAS ADVANCED COMPUTING CENTER

Further potential improvement

- Adopting SETTLE (analytical) algorithm instead of SHAKE to keep rigid geometry for water molecules. Expect ~10% speedup.