Optimized Force Calculation of Molecular Dynamics Simulations for the Intel Xeon Phi

N. Tchipev, A. Wafai, C. Glass, W. Eckhardt, A. Heinecke, H. Bungartz, P. Neumann
Technische Universität München
25.08.2015
Overview

Background and Motivation
Molecular Dynamics
Xeon approach
Early Xeon Phi results

Contributions
Optimized gather-scatter performance
OpenMP Parallelization of the Linked-Cell Method

Results
Hyperthreading experiments
Absolute Single-core Performance
OpenMP Performance

Closing remarks
Final results
Outlook
**ls1-mardyn**: MD for Chemical Engineering

### Method
- cutoff $r_c$, linked-cells
- rigid, multi-centered molecules
- LJ, C, D, Q
- C++, double precision

### Force Calculation

\[
U_{LJ}(r_{ij}) := 4\epsilon\left(\frac{\sigma}{r_{ij}}\right)^{12} - \left(\frac{\sigma}{r_{ij}}\right)^6
\]

\[
F_{ij} = -\nabla U_{LJ}(r_{ij})
\]

- 60-90% of total simulation time
- CoM-cutoff condition
- Newton’s third law optimization (N3): 
  \[
  F_{ij} = -F_{ji}
  \]
- $O(N^2)$ FLOPs (locally), $O(N)$ data 
  \[
  \Rightarrow \text{compute bound}
  \]
- vectorize over *sites*, not *dimension*
Xeon approach

AoS:

```c
struct Molecule {
    double R[3], F[3], V[3];
};
arr<Molecule> cellData(N);
```

SoA:

```c
arr<double> Rx(N), Ry(N), ...;
arr<double> Fx(N), Fy(N), ...;
```

“Sliding Window” win:

```
21 22 23 24 25 26 27 28 29 30
11 12 13 14 15 16 17 18 19 20
1 2 3 4 5 6 7 8 9 10
```

Masking via SSE, AVX:

```
m0s0 m1s0 m1s1 m2s0 m2s1 m3s0 m4s0 m4s1
m5s0 m5s1 m5s2 m6s0 m6s1 m6s2 m6s3
```

MPI: “Full-Shell” Domain Decomposition

Scalability on SuperMUC

![Graph showing scalability on SuperMUC](image-url)
Target platform - SuperMIC

Ivy Bridge Hosts (IVB)
- dual socket Xeon E5-2650v2
- 2.6 GHz
- $2 \times 8 = 16$ cores
- $16 \times 2 = 32$ threads
- AVX: 8 DP FLOPs per cycle (width=4)
- theoretical peak: 0.3 TFLOPS
- 64 GB RAM

Intel MIC Accelerators (MIC)
- Xeon Phi 5110p
- 1.1 GHz
- 60 cores, ring interconnect
- $60 \times 4 = 240$ threads
- KNC: 16 DP FLOPs per cycle (width=8)
- theoretical peak: 1.0 TFLOPS
- 8 GB RAM
- in-order execution
- fused multiply-add
- $\geq 2$ threads per core
- gather-scatter instructions
Early MIC results

- how many threads to run on?
- gather-scatter performance?
- can’t fit 240 MPI ranks for these scenarios...
- ⇒ need OpenMP
- no established method for OpenMP on linked cells

GPU approach:
drop N3 between cells

```
m0s0 m1s0 m1s1 m2s0 m2s1 m3s0 m4s0 m4s1
m5s0 m5s1 m5s2 m5s3 m6s0 m6s1 m6s2 m6s3
m1s0 m1s1 m4s0 m4s1 m6s0 m6s1 m6s2 m6s3
```

Gather positions
Scatter forces
Overview

Background and Motivation
Molecular Dynamics
Xeon approach
Early Xeon Phi results

Contributions
Optimized gather-scatter performance
OpenMP Parallelization of the Linked-Cell Method

Results
Hyperthreading experiments
Absolute Single-core Performance
OpenMP Performance

Closing remarks
Final results
Outlook
Trade gathers and scatters for broadcasts

Indexvector iV; // indices of interaction-partner sites
for m1 in soa1 { iV.clear();
    // evaluate CoM cutoff-condition
    for m2 in soa2 { ... mask_packstore(iV, m2.sites...) }
    // compute LJ-Kernel forces
    if (m1 is single-centered) {for i in iV {...}}
    else {
        for i in iV {
            // get partner positions
            r2 = gather.load(i[0:7], soa2.R, ...)
            for site in m1 {
                r1 = bcast.load(m1.r);
                f = LJ(r2−r1)...
            }
            scatter_store(i[0:7], soa2.F,...)
        }
    }
}
**Novel OpenMP Parallelization Scheme**

- applicable also to IVB
  - “re-shuffling” of cell data
  - need all SoAs simultaneously
Overview

Background and Motivation
Molecular Dynamics
Xeon approach
Early Xeon Phi results

Contributions
Optimized gather-scatter performance
OpenMP Parallelization of the Linked-Cell Method

Results
Hyperthreading experiments
Absolute Single-core Performance
OpenMP Performance

Closing remarks
Final results
Outlook
Hyperthreading

threads/processes pinned on same core

IVB: overlapping *add* and *mul*

MIC: issue-rate, overlapping gather-scatter, in-order
Absolute Single-core Performance

**IVB** 9-14% of theo. peak
- `mul` dominate
- division
- operations are chained

**MIC** 4-8% of theo. peak
- new gather-scatter scalable in num. centers (CoM vs Site)
- FPU not yet saturated (9.3% at 16CLJ)
- only 7 out of 43 FLOPS in LJ kernel can be fused into fused-multiply-add

on same core:
- IVB: 2 processes win
- MIC: 4 threads `c08`
OpenMP Performance - strong scaling

**IVB**
- win hard to beat
- c08 81-89% eff. at 16 cores
- good also at 32 threads

**MIC**
- c08 82-94% eff. at 60 cores
- good also up to 240 threads
- excellent 1.3M mol., decent 0.16M
Overview

Background and Motivation
Molecular Dynamics
Xeon approach
Early Xeon Phi results

Contributions
Optimized gather-scatter performance
OpenMP Parallelization of the Linked-Cell Method

Results
Hyperthreading experiments
Absolute Single-core Performance
OpenMP Performance

Closing remarks
Final results
Outlook
Final results

MIC

- gather-scatter: factor 1-1.4
- recovering N3: factor 1.4-1.5
- total: factor 1.4 - 1.8\(^a\)
- factor 1-1.3 faster than current host

\(^a\)1CLJ rc3: improved SoA management
Outlook

- Charge, Dipole, Quadrupole
- optimize beyond force calculation
- improve MPI performance
- load-balancing between Xeon and Xeon-Phi

Open Source

www.ls1-mardyn.de

Thank you for the attention!
Backup: Breakdown of time in force calculation
Backup: SNB Speed-ups due to Vectorization

Increasing width can decrease hit-rate!

1LJ vector utilisation with masking:
- SSE 70-80%
- AVX 50-60%
- KNC \leq 45%
Backup: Xeon Phi

Speed-up, KNC 5110p 240 threads

SoA & KNC
SoA

#sites potential

m0s0 m1s0 m1s1 m2s0 m2s1 m3s0 m4s0 m4s1
m5s0 m5s1 m5s2 m5s3 m6s0 m6s1 m6s2 m6s3

Gather positions
Scatter forces