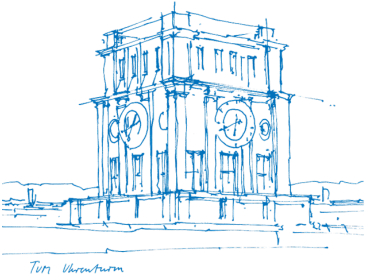
Load Balancing for Molecular Dynamics Simulations on Heterogeneous Architectures

HiPC 2016, Hyderabad, India

S. Seckler, N. Tchipev, H.-J. Bungartz and P. Neumann

Scientific Computing in Computer Science
Technical University of Munich, Germany

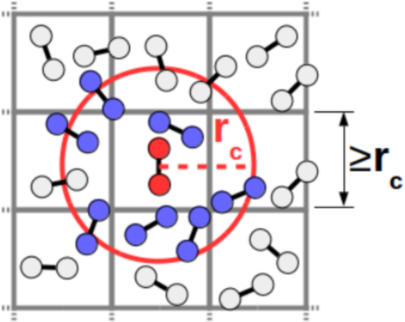
December 20, 2016



Outline

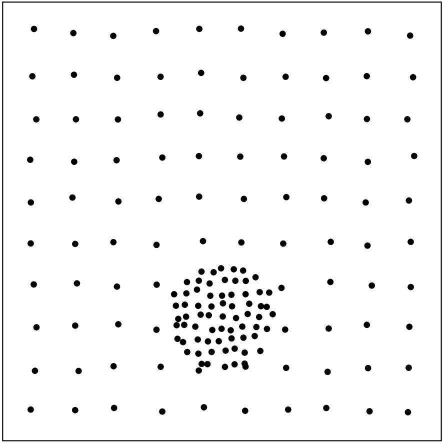
Short Introduction to Molecular Dynamics

- Pairwise particle/molecule interaction
- Short range interaction → cutoff radius → linked cells



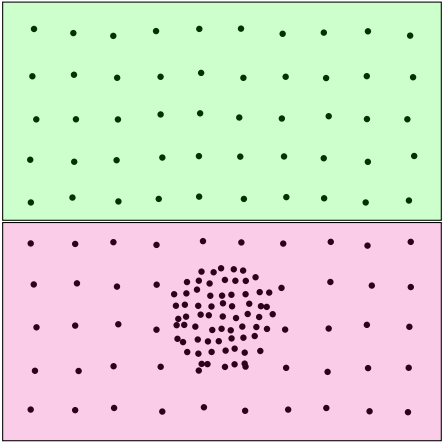
Introduction to Load Balancing for Molecular Dynamics

- Inhomogeneous particle distributions (e.g. droplets)
- Higher particle density in one subdomain → more work in that subdomain (load imbalance)
- Task: find subdomains with equal load



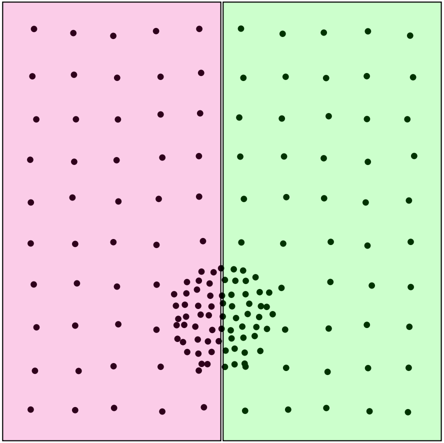
Introduction to Load Balancing for Molecular Dynamics

- Inhomogeneous particle distributions (e.g. droplets)
- Higher particle density in one subdomain $\rightarrow$ more work in that subdomain (load imbalance)
- Task: find subdomains with equal load



Introduction to Load Balancing for Molecular Dynamics

- Inhomogeneous particle distributions (e.g. droplets)
- Higher particle density in one subdomain → more work in that subdomain (load imbalance)
- Task: find subdomains with equal load



Outline

Motivation - Heterogeneous Clusters

1st Category Clusters with coprocessors (e.g. Tianhe-2, SuperMIC, ...)

- Offloading (handle heterogeneity at node level, e.g. GPU's)
  $\rightarrow$ homogeneous cluster
- Native mode (e.g. on Intel Xeon Phi's)
  $\rightarrow$ heterogeneous cluster

2nd Category Completely heterogeneous clusters (e.g. SuperMUC (as a whole), MAC Cluster)

- Islands/Nodes with varying layout, e.g. some with, some without accelerators
  $\rightarrow$ heterogeneous cluster

Motivation - Heterogeneous Clusters

1st Category Clusters with coprocessors (e.g. Tianhe-2, SuperMIC, ...)

- Offloading (handle heterogeneity at node level, e.g. GPU's)
  $\rightarrow$ homogeneous cluster
- **Native mode** (e.g. on Intel Xeon Phi's)
  $\rightarrow$ heterogeneous cluster

2nd Category Completely heterogeneous clusters (e.g. SuperMUC (as a whole), MAC Cluster)

- **Islands/Nodes with varying layout**, e.g. some with, some without accelerators
  $\rightarrow$ heterogeneous cluster

Outline

Introduction

Motivation

Algorithm

Setup

Results

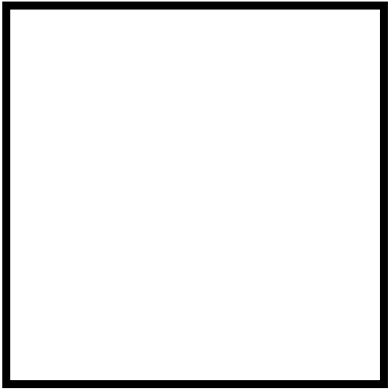
Conclusion & Outlook

Algorithm

k-d Tree-Based Partitioning

Domain partitioning according to k-d tree:

- Binary space partitioning tree

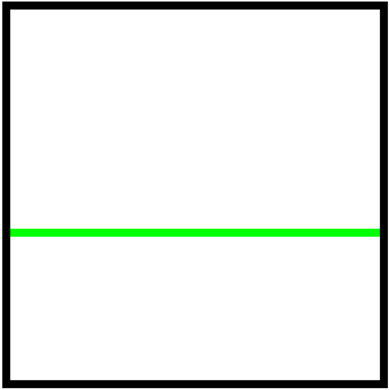


Algorithm

k-d Tree-Based Partitioning

Domain partitioning according to k-d tree:

- Binary space partitioning tree
- Split k-dimensional domain through k-1 -dimensional hyperplanes
- Hyperplanes are orthogonal to coordinate axes

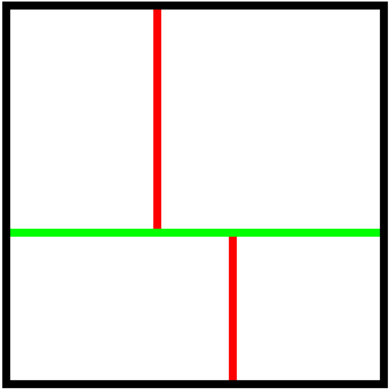


Algorithm

k-d Tree-Based Partitioning

Domain partitioning according to k-d tree:

- Binary space partitioning tree
- Split k-dimensional domain through k-1 -dimensional hyperplanes
- Hyperplanes are orthogonal to coordinate axes
- Apply recursively

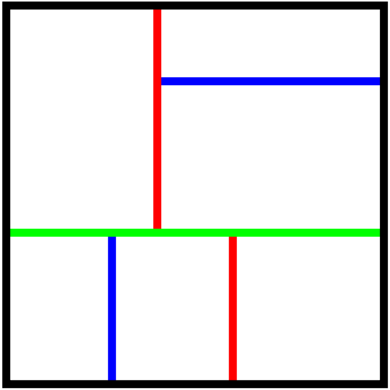


Algorithm

k-d Tree-Based Partitioning

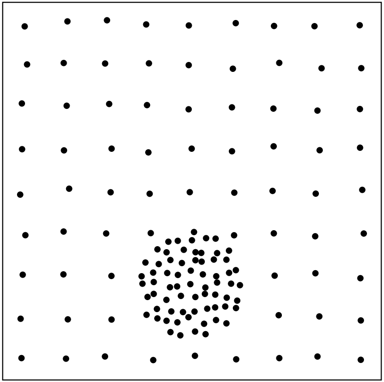
Domain partitioning according to k-d tree:

- Binary space partitioning tree
- Split k-dimensional domain through k-1 -dimensional hyperplanes
- Hyperplanes are orthogonal to coordinate axes
- Apply recursively



Algorithm

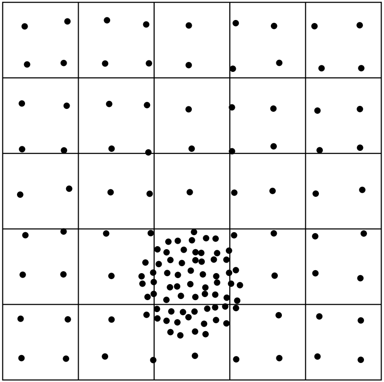
Heterogeneous Particle Distributions



Algorithm

Heterogeneous Particle Distributions

- Cell-based splitting



Algorithm

Heterogeneous Particle Distributions

- Cell-based splitting
- Desired load-balance: for each process: load is the same
- Find best possible splitting plain, s.t.:
  ratio of processes and ratio of loads are almost equal
- Apply recursively

1	1	1	1	1
1	1	1	1	1
1	2	3	2	1
1	3	4	3	1
1	2	3	2	1

Algorithm

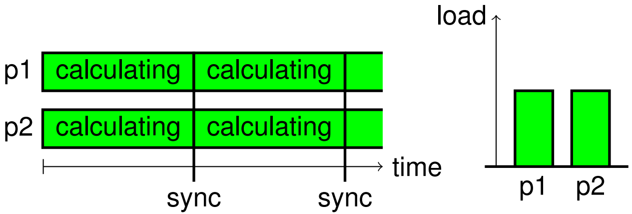
Heterogeneous Particle Distributions

- Cell-based splitting
- Desired load-balance: for each process: load is the same
- Find best possible splitting plain, s.t.:
  ratio of processes and ratio of loads are almost equal
- Apply recursively

1	1	1	1	1
1	1	1	1	1
1	2	3	2	1
1	3	4	3	1
1	2	3	2	1

Algorithm

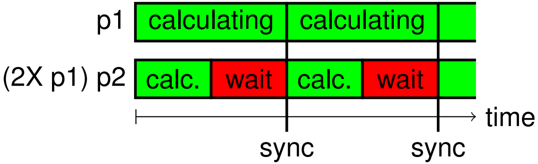
Heterogeneous Architectures

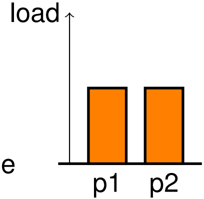


Algorithm

Heterogeneous Architectures

- Inhomogeneous clusters ⇒ nodes provide different performance
- Bad load distribution costs performance, time and energy

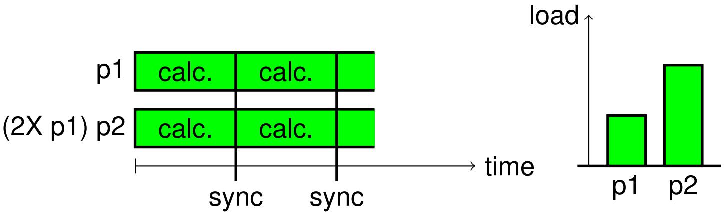




Algorithm

Heterogeneous Architectures

- Inhomogeneous clusters ⇒ nodes provide different performance
- Bad load distribution costs performance, time and energy
- Due to explicit (or implicit) synchronization points: Load has to be balanced properly
- Desired load-balance: for each process: time needed for computation (ratio of load and performance) is constant



Algorithm

Heterogeneous Architectures

Recursive Splitting:

1. Divide processes in two groups
2. Divide subdomain in two parts with load ratio according to performance ratio of the two groups
3. Apply recursively

1	1	1	1	1
1	1	1	1	1
1	2	3	2	1
1	3	4	3	1
1	2	3	2	1

Figure: Performance ratio 3:1

Algorithm

Heterogeneous Architectures

Recursive Splitting:

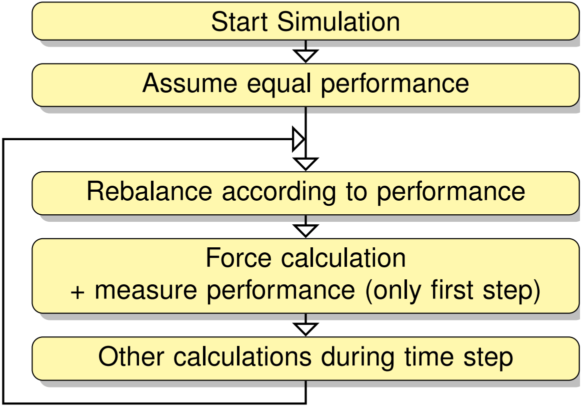
1. Divide processes in two groups
2. Divide subdomain in two parts with load ratio according to performance ratio of the two groups
3. Apply recursively

1	1	1	1	1
1	1	1	1	1
1	2	3	2	1
1	3	4	3	1
1	2	3	2	1

Figure: Performance ratio 3:1

Algorithm

Complete Algorithm



Outline

Setup

MAC Cluster

- Multiple partitions with different architectures (Intel Sandy Bridge (SNB), Intel Westmere (WSM), AMD Bulldozer (BDZ))
- Completely heterogeneous cluster

Scenario

- 512 k molecules à 2 LJ centers (ethane, C_2H_6)
- ≈ 37 molecules (74 sites) per cell
- $25 \times 25 \times 25$ linked cells

Outline

Results

Performance Measurements

Repeated performance measurements dangerous, if performance depends on problem size (smaller load $\Rightarrow$ smaller performance):

1. 2 identical processes, initial load slightly smaller on process 1
2. Perf proc 1 $<$ Perf proc 2
3. Load proc 1 $\downarrow$
4. Perf proc 1 $\downarrow$

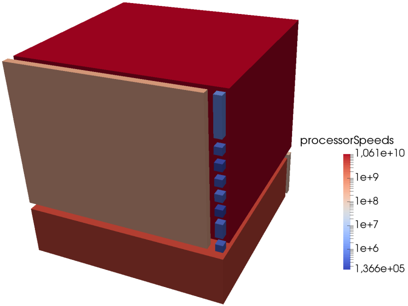


Figure: Small subdomains for dynamic performance measurements

Results

MAC Cluster: BDZ–SNB

- (SNB,BDZ)=(1.9x,1x)
- Performance gain of up to 1.8x

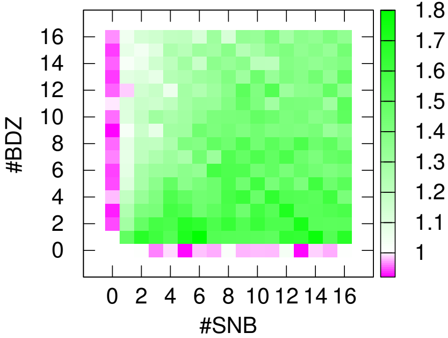
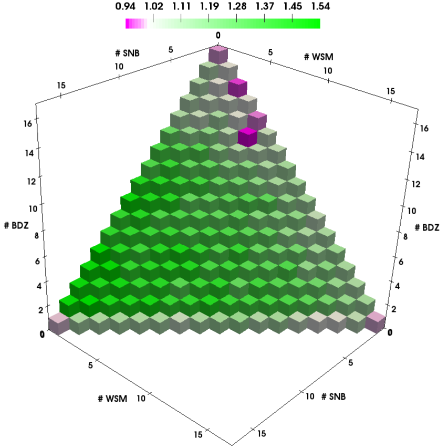


Figure: Speedup of performance-aware version compared to unaware version.

Results

All MAC Cluster Partitions

- BDZ, WSM and SNB partition used
- (SNB,WSM,BDZ)=(1.9,1.3,1)
- Performance gain through performance-aware load balancing of up to 50% for small scale scenario. (picture)



Results

All MAC Cluster Partitions – Production Run

- Production run:
- 344 M molecules à 1 LJ centers
- ≈ 43 molecules per cell
- $200 \times 200 \times 200$ linked cells
- Speedup 1.3x (of a maximum of 1.4x)

Outline

Conclusion & Outlook

Conclusion

- Handling of heterogeneity in clusters important
- Major speedups possible
- Applicable to (almost) any form of heterogeneous cluster
- Similar approaches for other simulation types possible

Outlook

- Validation for heterogeneous particle distributions
- Comparison with Zoltan (current work)
- Time-based performance and rebalancing scheme
  - \+ Circumvents problems with performance evaluation
  - \+ No cost estimation needed
  - \- No direct solution, but rather iterative approach
  - \+ Schemes without global communication possible

Questions?

Appendix

Detailed Cluster Description

MAC Cluster:

BDZ 19 nodes à 4 AMD Bulldozer Opteron 6274 (16 cores, 2.2 GHz), 256 GB RAM, QDR infiniband. AVX + FMA4.

SNB 28 nodes à 2 Intel Sandy Bridge-EP E5-2670 (8 cores, hyperthreading, 2.6 GHz), 128 GB RAM, QDR infiniband. AVX.

WSM 1 node à 4 Intel Westmere-EX Xeon E7-4830 (8 cores, hyperthreading, 2.13 GHz), 512 GB RAM. FDR infiniband. SSE.

SuperMIC:

- 32 nodes à 2 Intel Ivy Bridge-EP E5-2650 v2 (8 cores, hyperthreading, 2.6 GHz). 64 GB RAM. AVX
- Per node: 2 Xeon Phi 5110P (60 cores, 4-way hyperthreading, 1.1 GHz), 8 GB RAM each. FDR14 infiniband. IMCI (512 bit vector length)