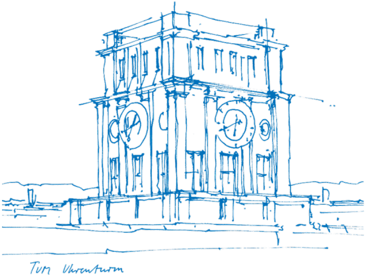
Fully Heterogeneous Load Balancing in ls1 MarDyn

SIAM PP 2018, Tokyo, Japan

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

Scientific Computing in Computer Science
Technical University of Munich, Germany

March 10, 2018

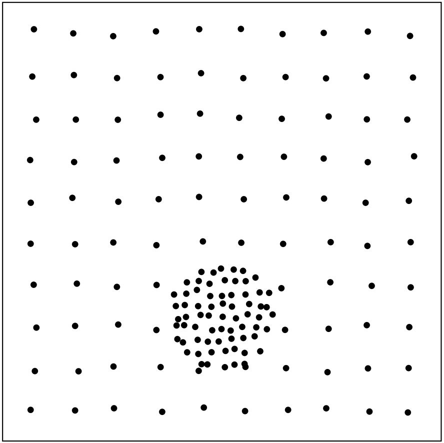


Outline

Challenge 1 – Inhomogeneous Loads

When? Multiple MPI processes

Observations Speedup not as expected, MPI processes idle

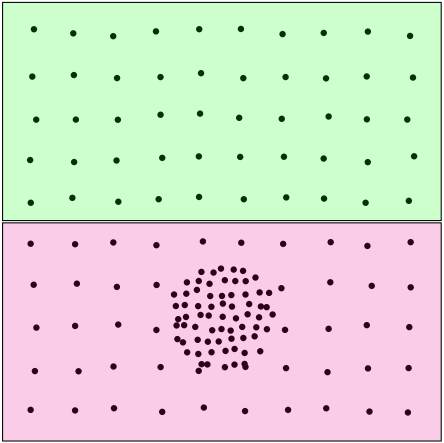


Challenge 1 – Inhomogeneous Loads

When? Multiple MPI processes

Observations Speedup not as expected, MPI processes idle

Reason Load imbalance



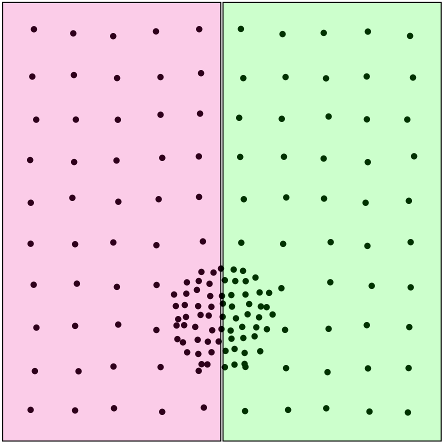
Challenge 1 – Inhomogeneous Loads

When? Multiple MPI processes

Observations Speedup not as expected, MPI processes idle

Reason Load imbalance

Solution Assign equal load to all processes



Challenge 1 – Inhomogeneous Loads

Decomposition Methods

How does a decomposition look like?

- Tree-based methods
  - Orthogonal Recursive Bisection (ORB) / k-d trees
  - Multisection method
  - Octrees
- Space-filling curves
- Diffusive Methods

Here: k-d trees (subdomains are cuboids)

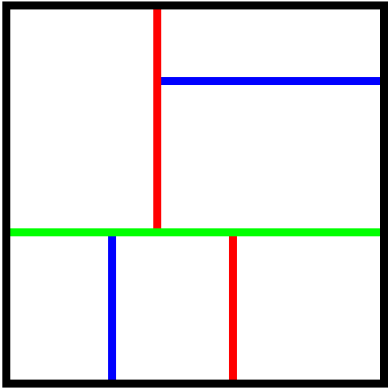


Figure: k-d tree

Challenge 1 – Inhomogeneous Loads

Decomposition Methods

How does a decomposition look like?

- Tree-based methods
  - Orthogonal Recursive Bisection (ORB) / k-d trees
  - Multisection method
  - Octrees
- Space-filling curves
- Diffusive Methods

Here: k-d trees (subdomains are cuboids)

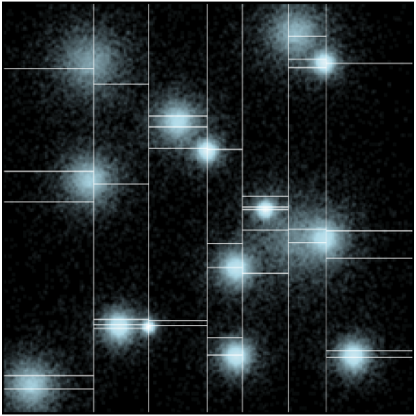


Figure: Multisection Method in FDPS [Iwasawa et. all, 2016]

Challenge 1 – Inhomogeneous Loads

Decomposition Methods

How does a decomposition look like?

- Tree-based methods
  - Orthogonal Recursive Bisection (ORB) / k-d trees
  - Multisection method
  - Octrees
- Space-filling curves
- Diffusive Methods

Here: k-d trees (subdomains are cuboids)

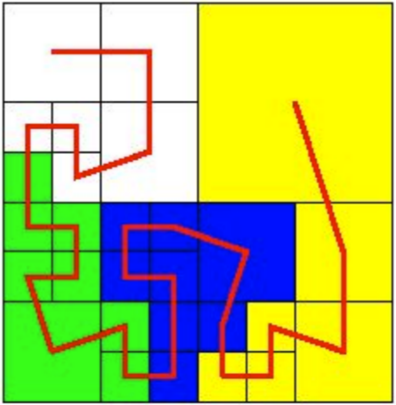


Figure: Partitioning using space-filling curves [Rutgers University]

Challenge 2 – Heterogeneous Hardware

Motivation

General use case	Bundle old workstations
Accelerators	Clusters with coprocessors (e.g. Tianhe-2, SuperMIC, ...) used in native mode
Heterogeneous clusters	Completely heterogeneous clusters (e.g. SuperMUC (as a whole), MAC Cluster) with islands of different configuration

Challenge 2 – Heterogeneous Hardware

When? Heterogeneous hardware

Observation Idling processes, not the desired speedup even though load is balanced

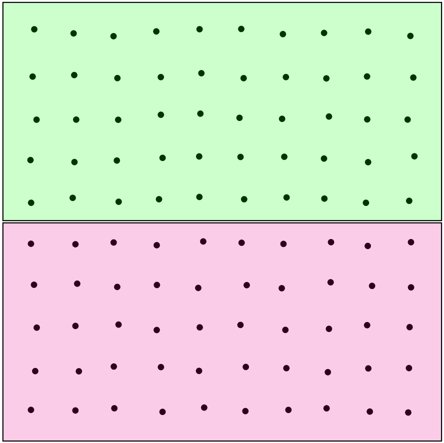


Figure: Heterogeneous splitting

Challenge 2 – Heterogeneous Hardware

When? Heterogeneous hardware

Observation Idling processes, not the desired speedup even though load is balanced

Reason Processes have different performance

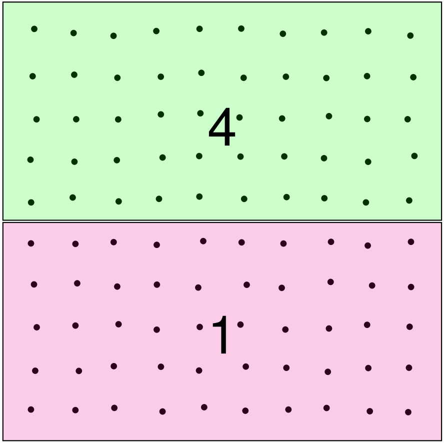


Figure: Bad Heterogeneous splitting with constant performance ratio 4:1

Challenge 2 – Heterogeneous Hardware

When? Heterogeneous hardware

Observation Idling processes, not the desired speedup even though load is balanced

Reason Processes have different performance

Solution Assign load such that every process needs the same amount of time

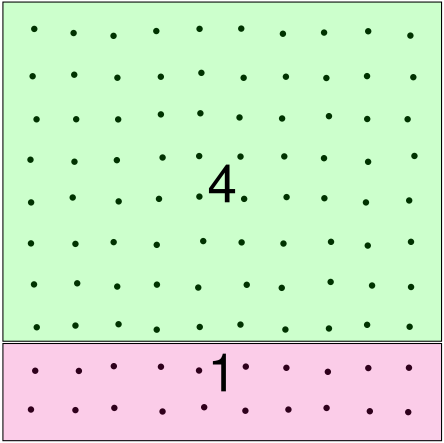
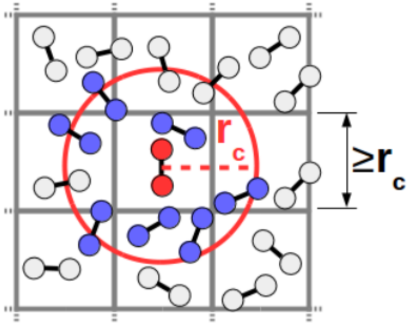


Figure: Heterogeneous splitting with constant performance ratio 4:1

Outline

Short Introduction to Molecular Dynamics

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

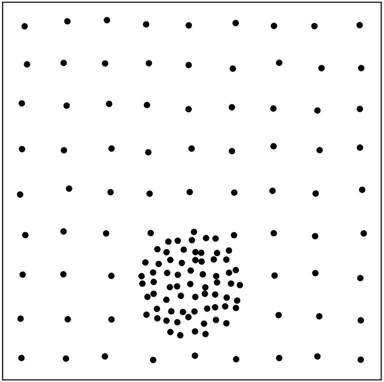


Outline

Challenge 0

What is the load?

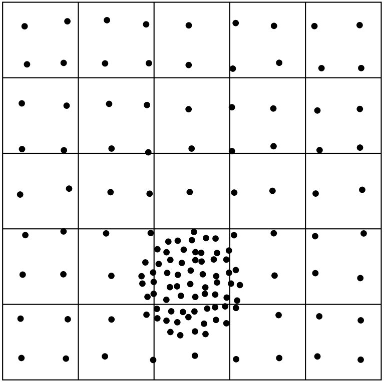
- Often: load/time per particle, sampling (e.g. FDPS)



Challenge 0

What is the load?

- Often: load/time per particle, sampling (e.g. FDPS)
- Here: load/time calculations per cell since
  - Algorithms are cell-based



Challenge 0

What is the load?

- Often: load/time per particle, sampling (e.g. FDPS)
- Here: load/time calculations per cell since
  - Algorithms are cell-based

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

Challenge 0

What is the load?

- Often: load/time per particle, sampling (e.g. FDPS)
- Here: load/time calculations per cell since
  - Algorithms are cell-based

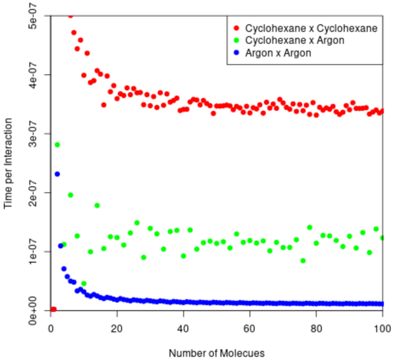
1	1	1	1	1
1	1	1	1	1
1	2	3	2	1
1	3	4	3	1
1	2	3	2	1

Challenge 0

What is the load?

- Often: load/time per particle, sampling (e.g. FDPS)
- Here: load/time calculations per cell since
  - Algorithms are cell-based
  - Performance heavily depends on density (particles per cell) and interaction potential
  - Time for interaction between two cells A and B:

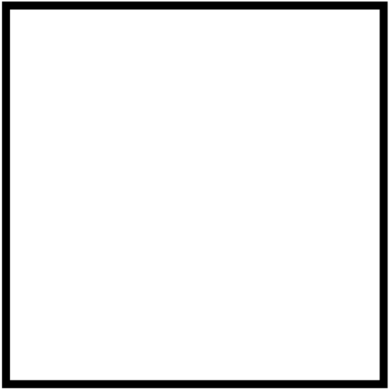
$$t(n_{A,1}, \ldots, n_{A,n_{\text{types}}}, n_{B,1}, \ldots, n_{B,n_{\text{types}}})$$



Challenge 1

Distributing the load

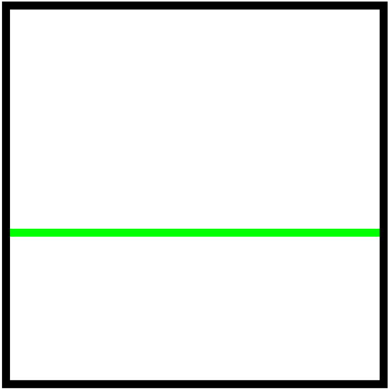
- k-d trees
- Load as in Challenge 0
- Recursive divide and conquer
  - Split domain, assign proper amount of processes
  - Solve sub-problems independently



Challenge 1

Distributing the load

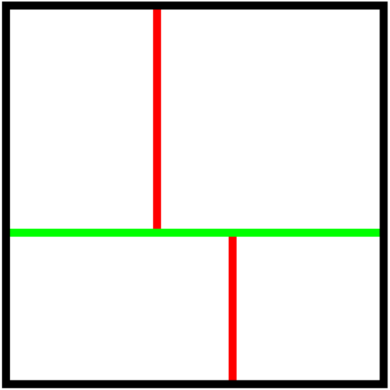
- k-d trees
- Load as in Challenge 0
- Recursive divide and conquer
  - Split domain, assign proper amount of processes
  - Solve sub-problems independently



Challenge 1

Distributing the load

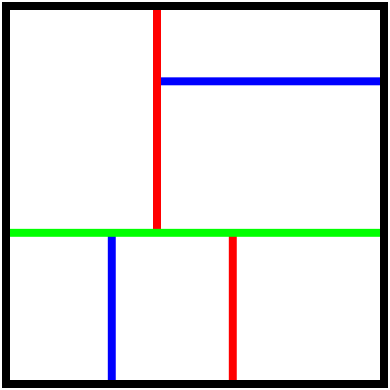
- k-d trees
- Load as in Challenge 0
- Recursive divide and conquer
  - Split domain, assign proper amount of processes
  - Solve sub-problems independently



Challenge 1

Distributing the load

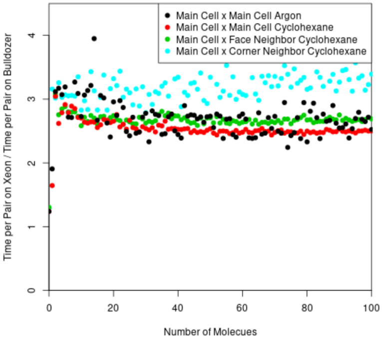
- k-d trees
- Load as in Challenge 0
- Recursive divide and conquer
  - Split domain, assign proper amount of processes
  - Solve sub-problems independently



Challenge 2

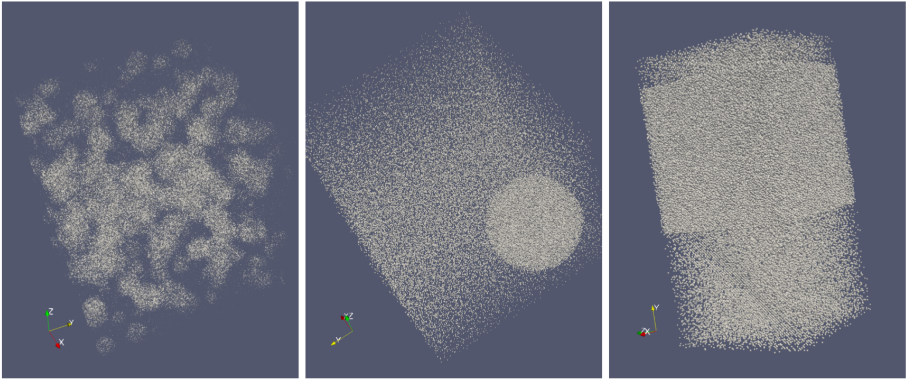
Heterogeneous Architectures

- k-d trees
- Split processes in groups depending on hardware
- Split domain in proper parts for groups
- Load distribution within groups as before
- 2 approaches
  1. Simply split such that all groups need same time – good for const perf. ratios
  2. WIP: choose areas wisely



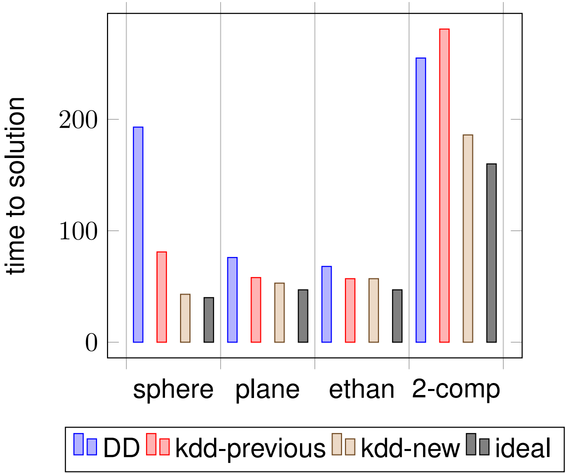
Results

Scenarios



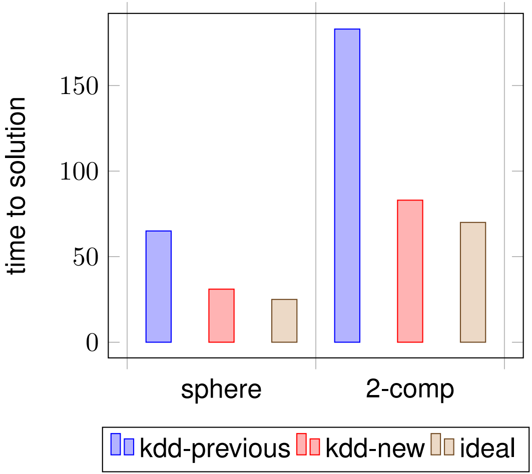
Results

Homogeneous cluster



Results

Heterogeneous cluster



Outline

Challenges of Heterogeneous Load Balancing

Molecular Dynamic Basics

Solving the Challenges

Conclusion & Outlook

Conclusion & Outlook

Conclusion

- Fully heterogeneous
  - Heterogeneous density
  - Multiple components
- Time-measurements
- No a priori knowledge needed

Outlook

- WIP: Wise subdomain choosing; will save energy

low density high frequency / low vectorization width
high density high vectorization width (even though lower frequency)

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)