# AutoPas: Optimising Multi-site Molecular Dynamics Simulations with Auto-tuning and Kokkos

Samuel J. Newcome

Chair for Scientific Computing in Computer Science
School of Computation, Information and Technology
Technical University of Munich

16th June 2022

# Molecular Dynamics Simulations

- Consider a simulation of N rigid body molecules.
- Each molecule consists of a small number of sites.
- Every timestep, we require the force and torque acting on every molecule.
- Typically, this requires force calculations between every inter-molecular pair of sites every timestep.
- $\Rightarrow$ $O(N^2)$ calculations per timestep.

For large N, this is a big computational cost, and can easily take $> 99\%$ the time of a simulation.

# Molecular Dynamics Simulations

- Consider a simulation of N rigid body molecules.
- Each molecule consists of a small number of sites.
- Every timestep, we require the force and torque acting on every molecule.
- Typically, this requires force calculations between every inter-molecular pair of sites every timestep.
- $\Rightarrow$ $O(N^2)$ calculations per timestep.

For large N, this is a big computational cost, and can easily take $> 99\%$ the time of a simulation.

# Molecular Dynamics Simulations

- Consider a simulation of N rigid body molecules.
- Each molecule consists of a small number of sites.
- Every timestep, we require the force and torque acting on every molecule.
- Typically, this requires force calculations between every inter-molecular pair of sites every timestep.
- $\Rightarrow O(N^2)$ calculations per timestep.

For large N, this is a big computational cost, and can easily take $> 99\%$ the time of a simulation.

# Molecular Dynamics Simulations

- Consider a simulation of N rigid body molecules.
- Each molecule consists of a small number of sites.
- Every timestep, we require the force and torque acting on every molecule.
- Typically, this requires force calculations between every inter-molecular pair of sites every timestep.
- $\Rightarrow O(N^2)$ calculations per timestep.

For large N, this is a big computational cost, and can easily take $> 99\%$ the time of a simulation.

# Molecular Dynamics Simulations

- Consider a simulation of N rigid body molecules.
- Each molecule consists of a small number of sites.
- Every timestep, we require the force and torque acting on every molecule.
- Typically, this requires force calculations between every inter-molecular pair of sites every timestep.
- $\Rightarrow$ $O(N^2)$ calculations per timestep.

For large N, this is a big computational cost, and can easily take $> 99\%$ the time of a simulation.

# Molecular Dynamics Simulations

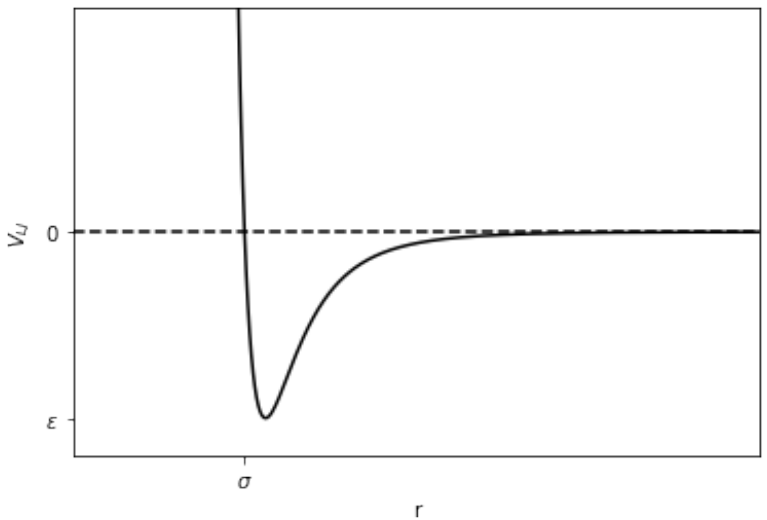
- Consider a simulation of N rigid body molecules.
- Each molecule consists of a small number of sites.
- Every timestep, we require the force and torque acting on every molecule.
- Typically, this requires force calculations between every inter-molecular pair of sites every timestep.
- $\Rightarrow$ $O(N^2)$ calculations per timestep.

For large N, this is a big computational cost, and can easily take $> 99\%$ the time of a simulation.

# Short range potentials

Consider the Lennard-Jones potential:

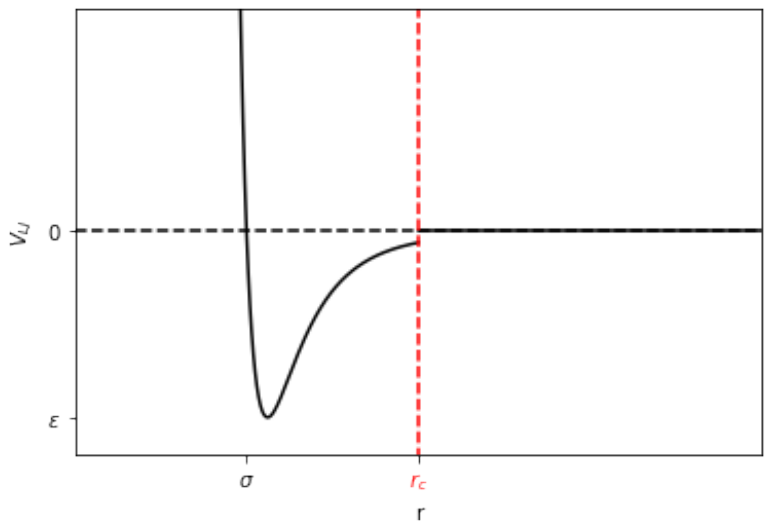
$$V_{LJ}(r) = 4\epsilon\left(\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right)$$



# Short range potentials

Consider the Lennard-Jones potential:

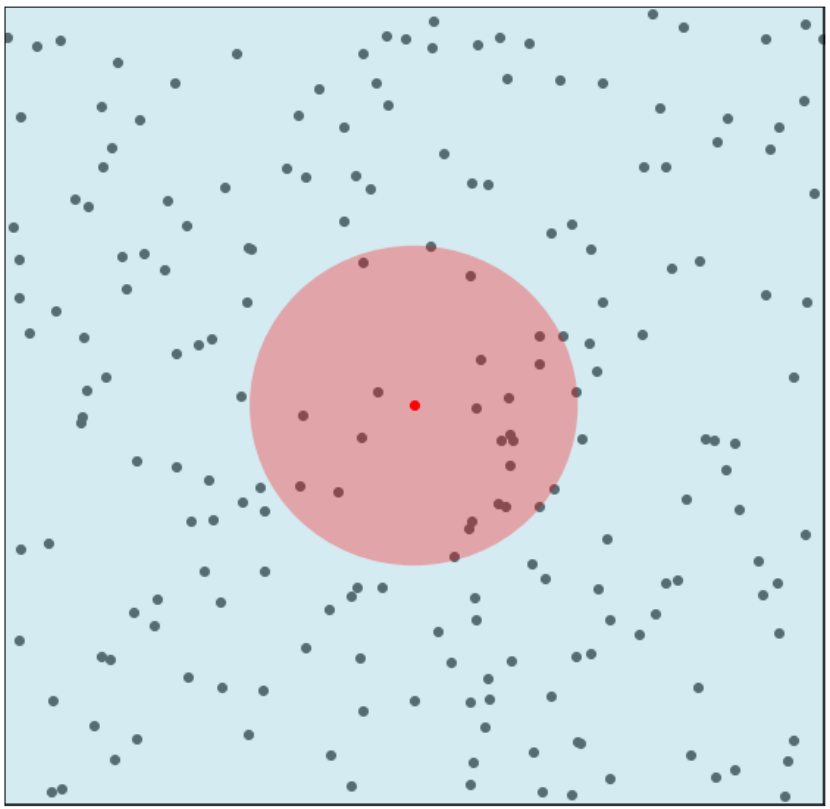
$$V_{LJ}(r) = 4\epsilon\left(\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right)$$




Now introduce a cutoff for $r > r_c$

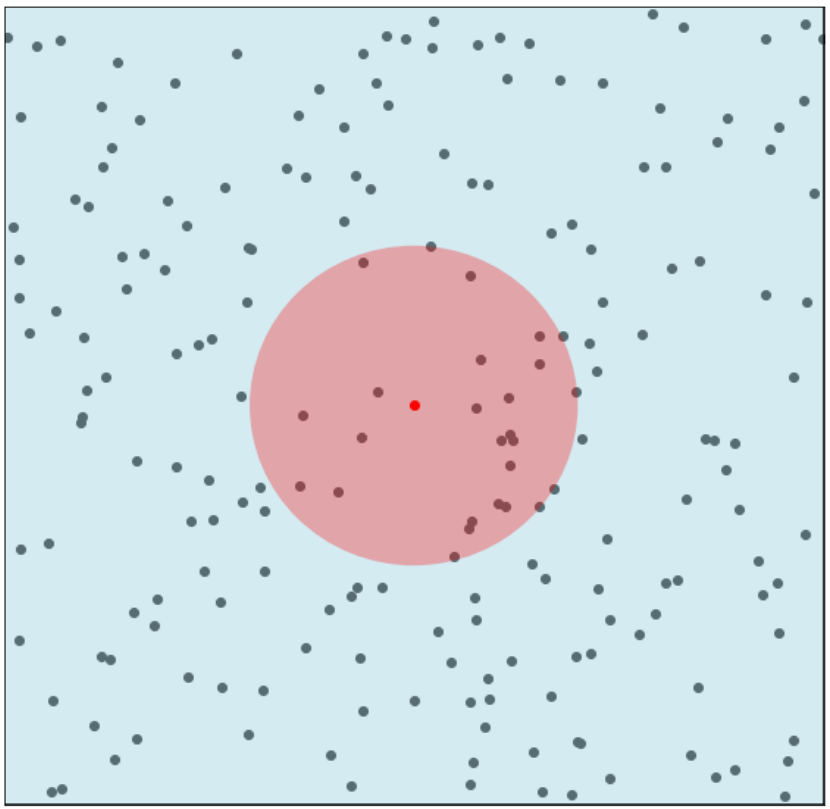
# Short range particle containers

- We still have $O(N^2)$ distance calculations to determine the cutoff.
- This is still too costly.
- Particle containers have been developed to alleviate this.



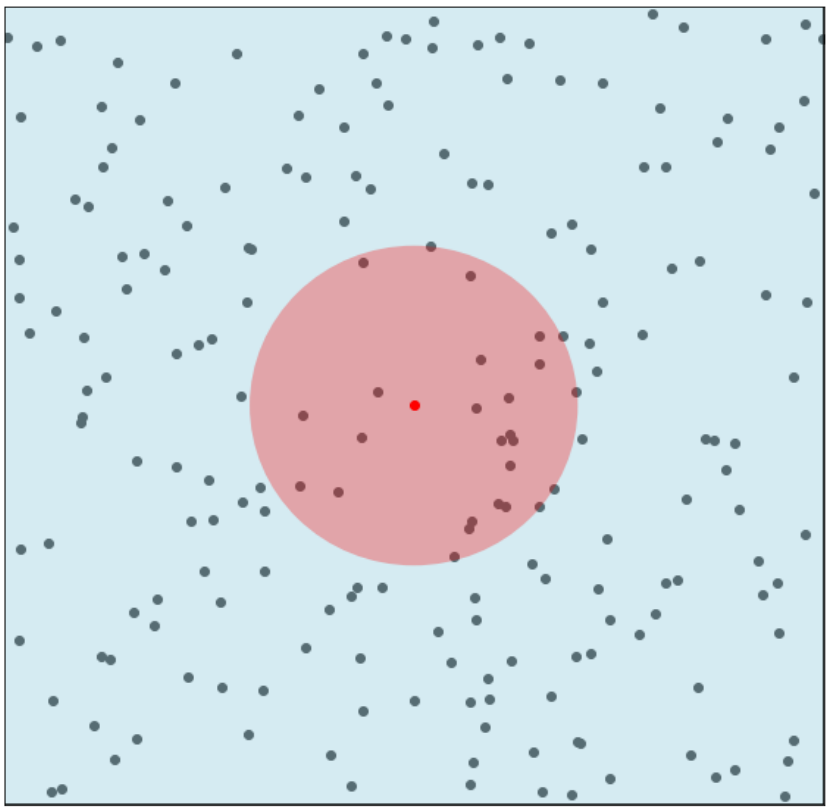
# Short range particle containers

- We still have $O(N^2)$ distance calculations to determine the cutoff.
- This is still too costly.
- Particle containers have been developed to alleviate this.



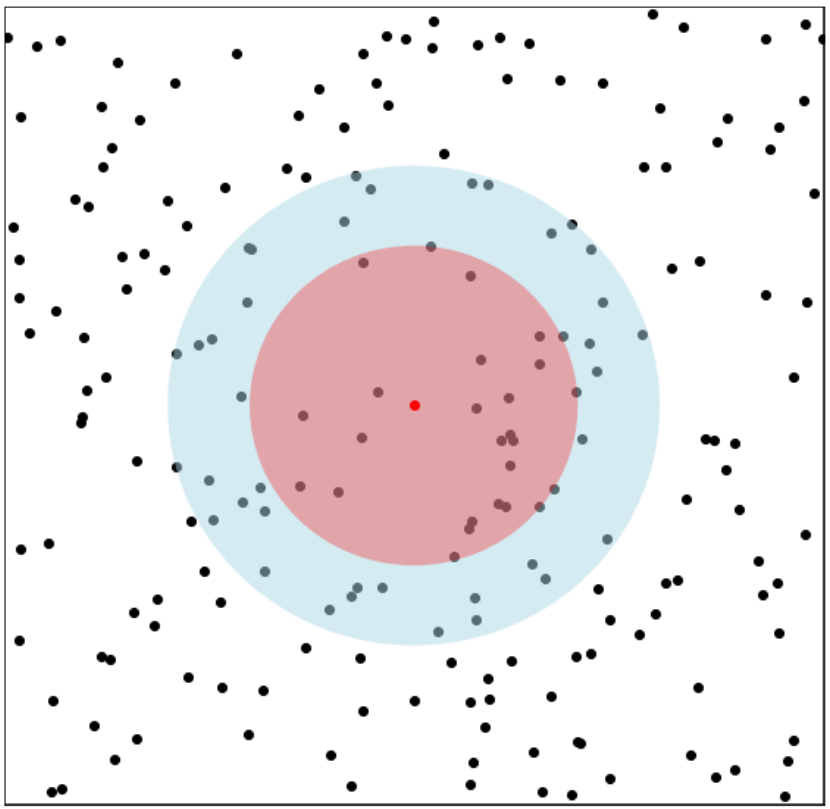
# Short range particle containers

- We still have $O(N^2)$ distance calculations to determine the cutoff.
- This is still too costly.
- Particle containers have been developed to alleviate this.



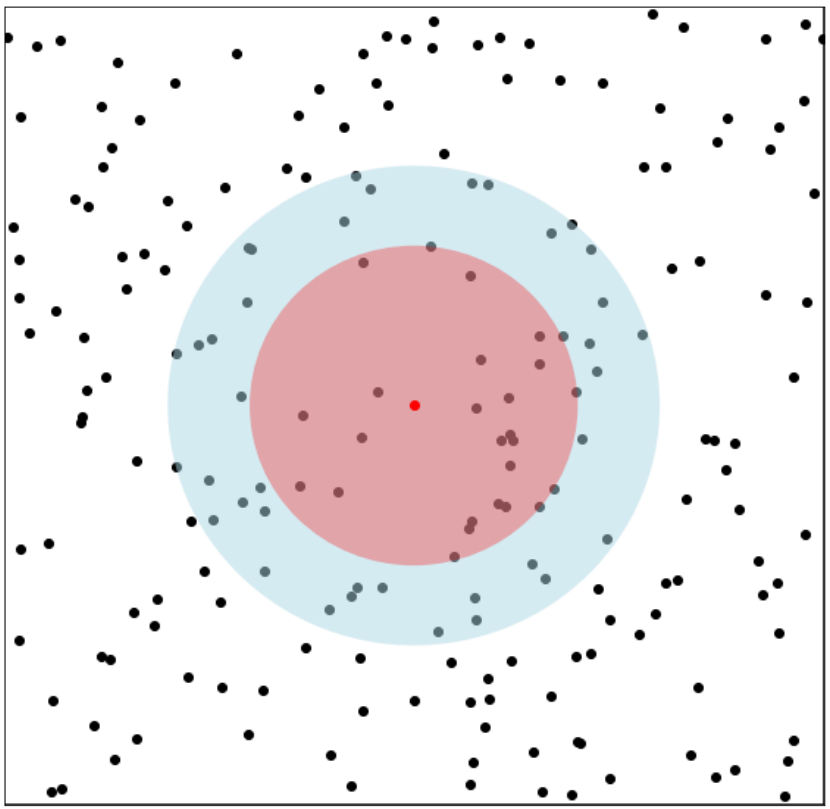
# Short range particle containers: Verlet Lists

- Keep neighbour list of every other molecule within a $r_c + s$ cutoff.
- Only calculate distance to neighbours to determine cutoff.
- Rebuild the neighbour list every $n_{VL}$ timesteps.



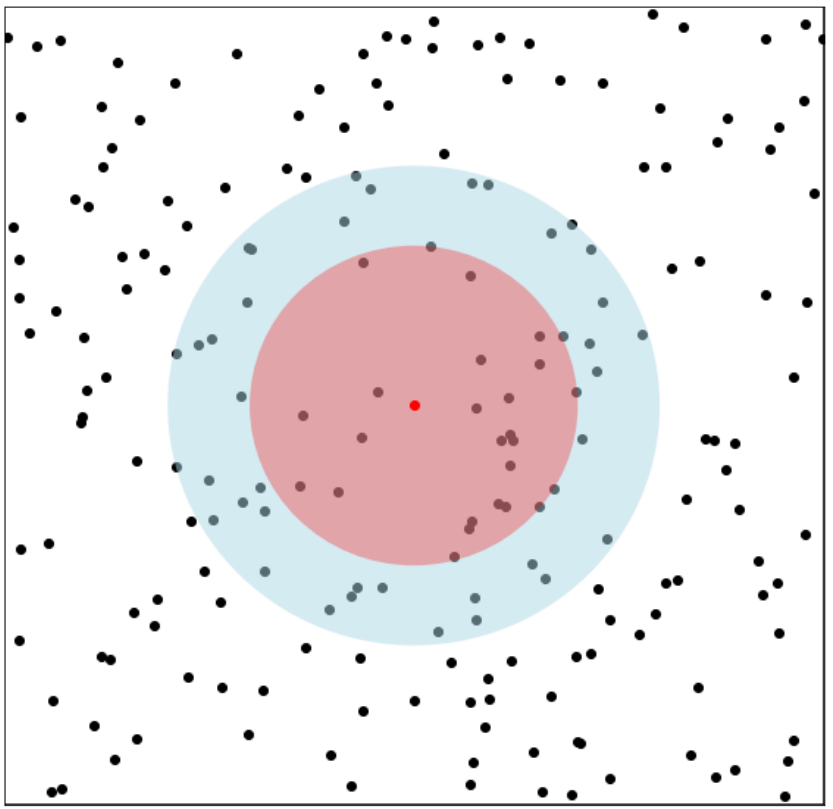
# Short range particle containers: Verlet Lists

- Keep neighbour list of every other molecule within a $r_c + s$ cutoff.
- Only calculate distance to neighbours to determine cutoff.
- Rebuild the neighbour list every $n_{VL}$ timesteps.



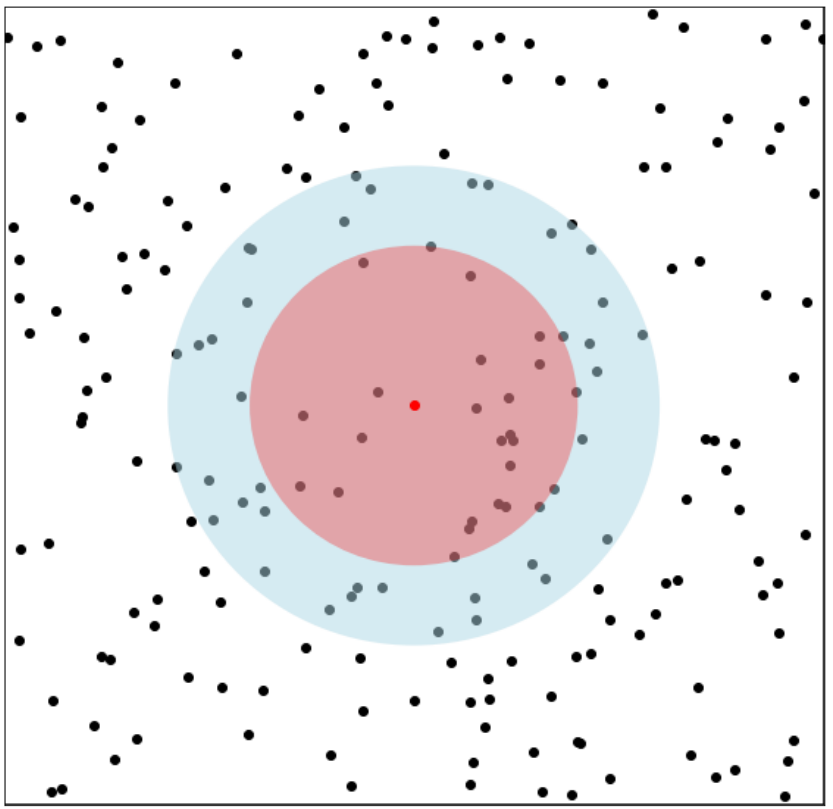
# Short range particle containers: Verlet Lists

- Keep neighbour list of every other molecule within a $r_c + s$ cutoff.
- Only calculate distance to neighbours to determine cutoff.
- Rebuild the neighbour list every $n_{VL}$ timesteps.



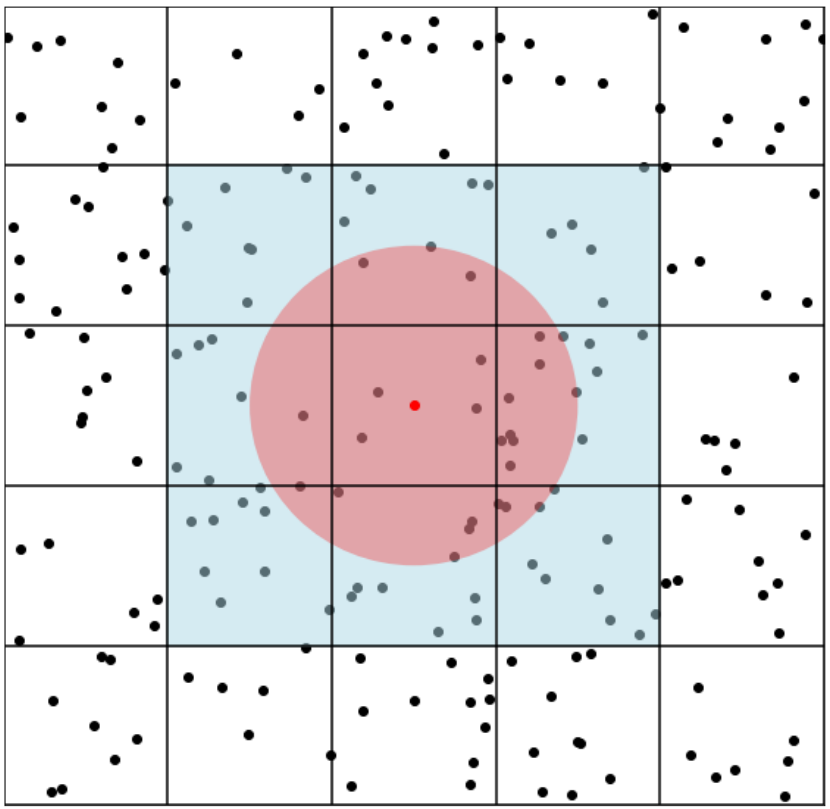
# Short range particle containers: Verlet Lists

- Keep neighbour list of every other molecule within a $r_c + s$ cutoff.
- Only calculate distance to neighbours to determine cutoff.
- Rebuild the neighbour list every $n_{VL}$ timesteps.



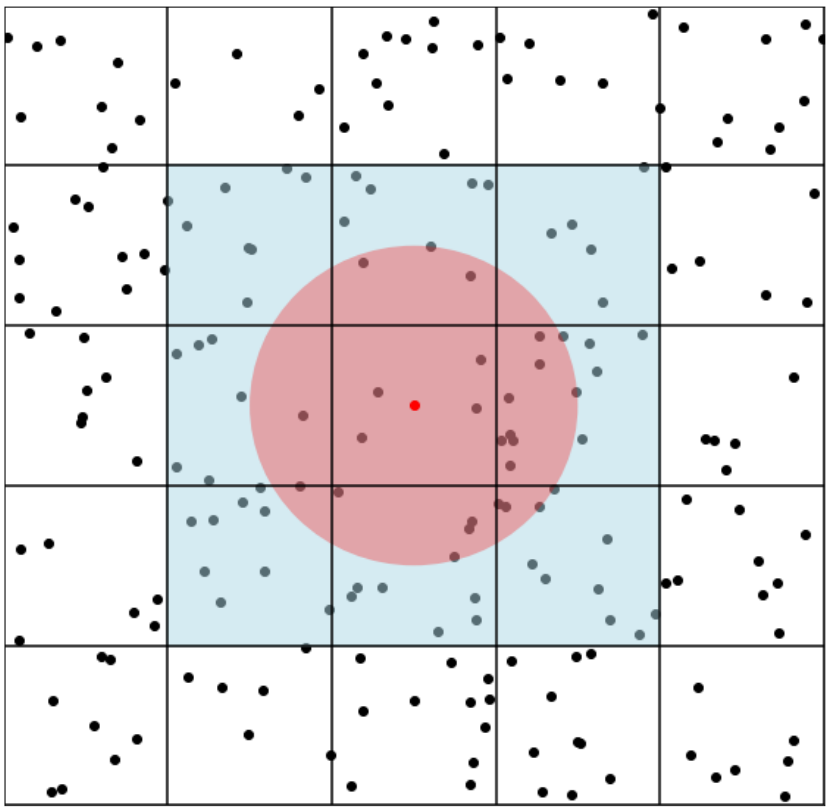
# Short range particle containers: Linked Cells

- Divide molecules in cells.
- Calculate cutoffs only with neighbouring cells.



# Short range particle containers: Linked Cells

- Divide molecules in cells.
- Calculate cutoffs only with neighbouring cells.



# Further algorithmic options: Newton's Third Law

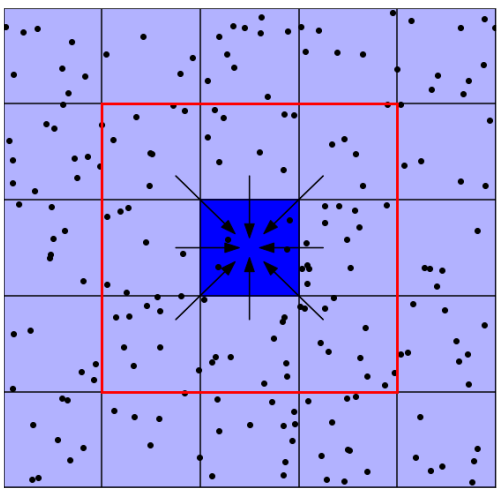
Apply the same force calculation to both particles in a pair.

- Pro: Halves force calculations.
- Con: Introduces race conditions.

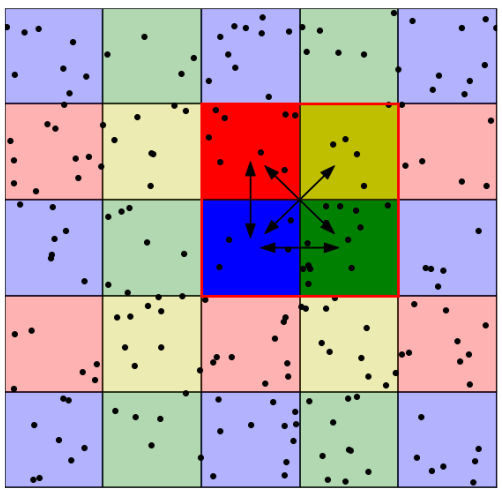
# Further algorithmic options: Traversals

For cells-based algorithms, there are many ways to traverse all cells.

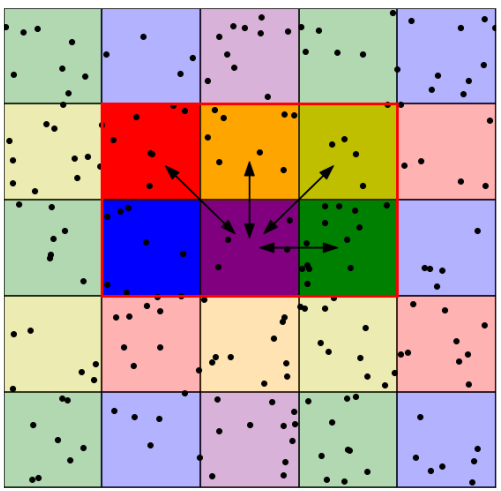
c01



c08

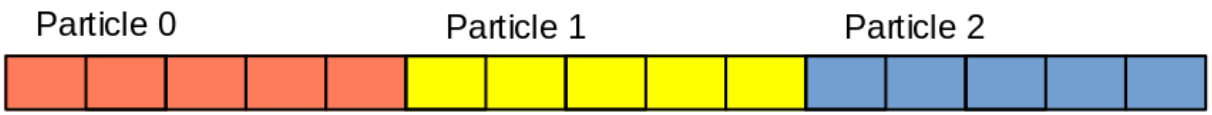


c18

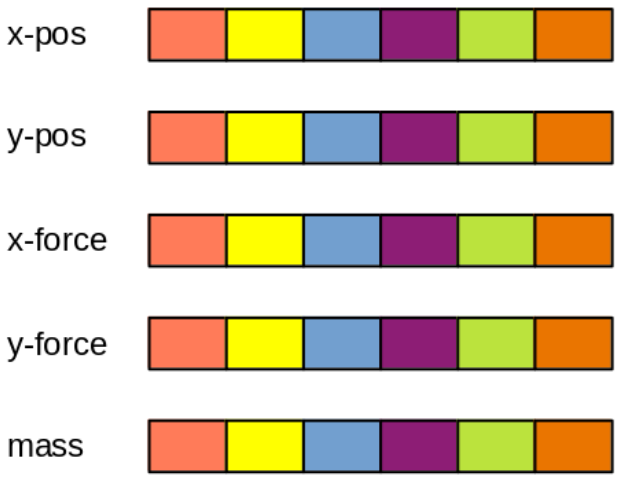


# Further algorithmic options: Data Structure

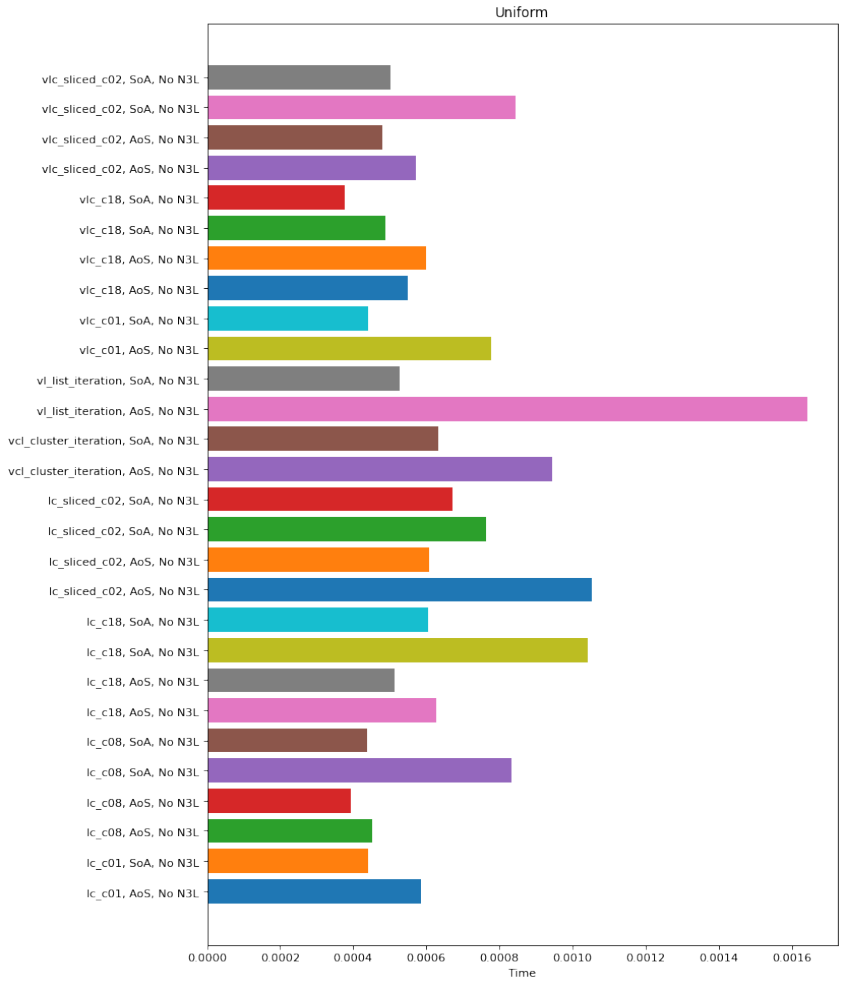
## Array-of-Structures (AoS)

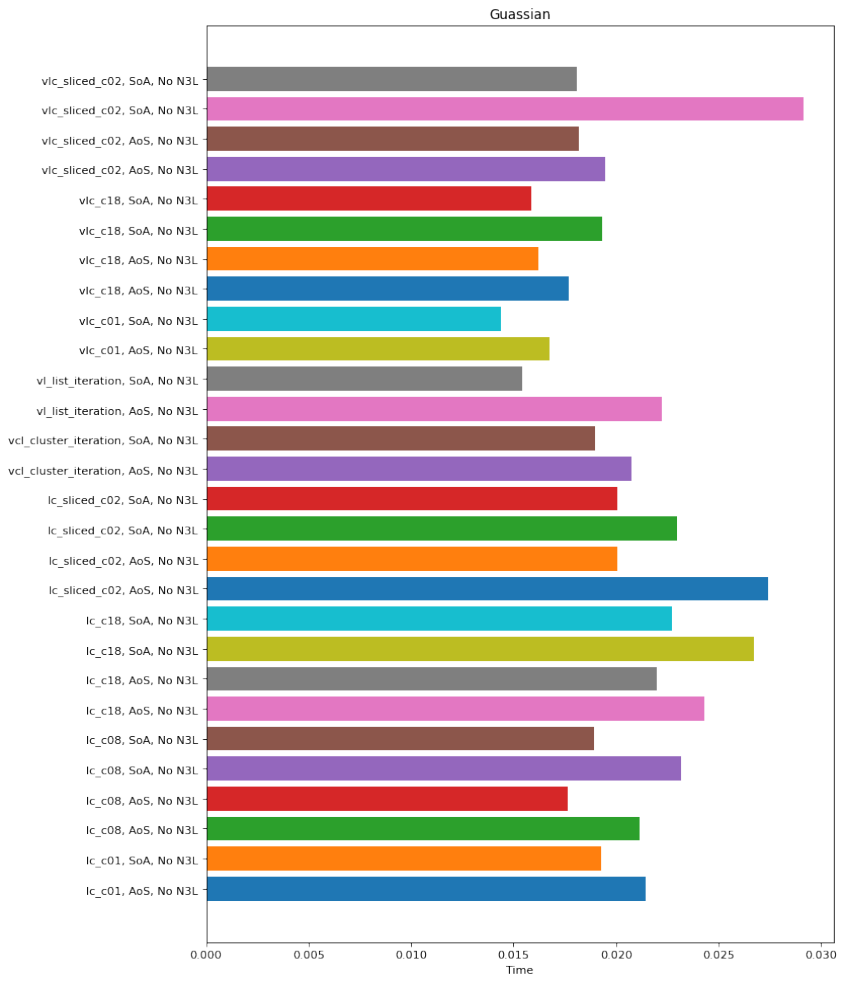



## Structure-of-Arrays (SoA)



# Motivation for Auto-tuning in MD Simulations: Homogeneity & Density



# Motivation for Auto-tuning in MD Simulations: Homogeneity & Density

Uniform:

- Best: Verlet List with AoS & No N3L optimisation
- Time: 0.01373 s

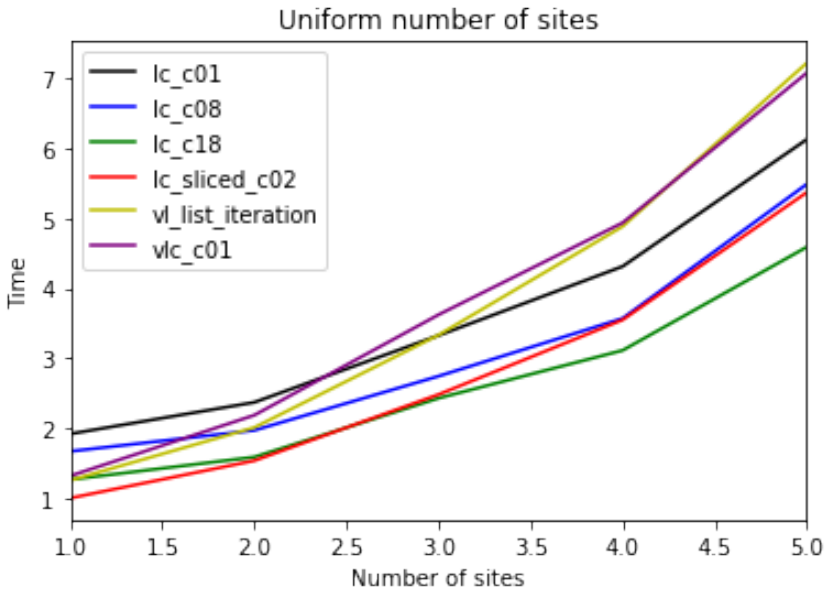
- Time for VLC C18, SoA, N3L: 0.01586 s

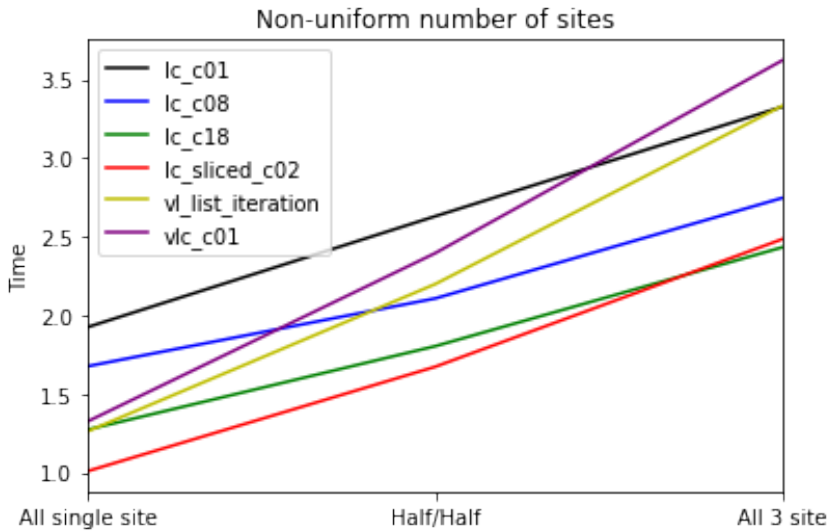
Guassian:

- Best: Verlet List Cells with SoA & N3L optimisation
- Time: 0.0003771 s

- Time for VL, AoS, No N3L: 0.0004145 s

# Motivation for Auto-tuning in MD Simulations: Complexity of Functor



# Motivation for Auto-tuning in MD Simulations

- There is no optimal 'silver bullet' algorithm in all scenarios.
- Choosing which container, traversal, and data structure to use is often beyond the knowledge of many domain scientists who build MD simulators.
- The optimal algorithm can vary throughout a simulation.

# Motivation for Auto-tuning in MD Simulations

- There is no optimal 'silver bullet' algorithm in all scenarios.
- Choosing which container, traversal, and data structure to use is often beyond the knowledge of many domain scientists who build MD simulators.
- The optimal algorithm can vary throughout a simulation.

# Motivation for Auto-tuning in MD Simulations

- There is no optimal 'silver bullet' algorithm in all scenarios.
- Choosing which container, traversal, and data structure to use is often beyond the knowledge of many domain scientists who build MD simulators.
- The optimal algorithm can vary throughout a simulation.

# AutoPas

- AutoPas is an auto-tuning library for the handling of particle interactions.
- It contains a collection of algorithms.
- It tests potential algorithms during tuning phases and uses chosen ‘best’ during subsequent non-tuning phases.

# AutoPas

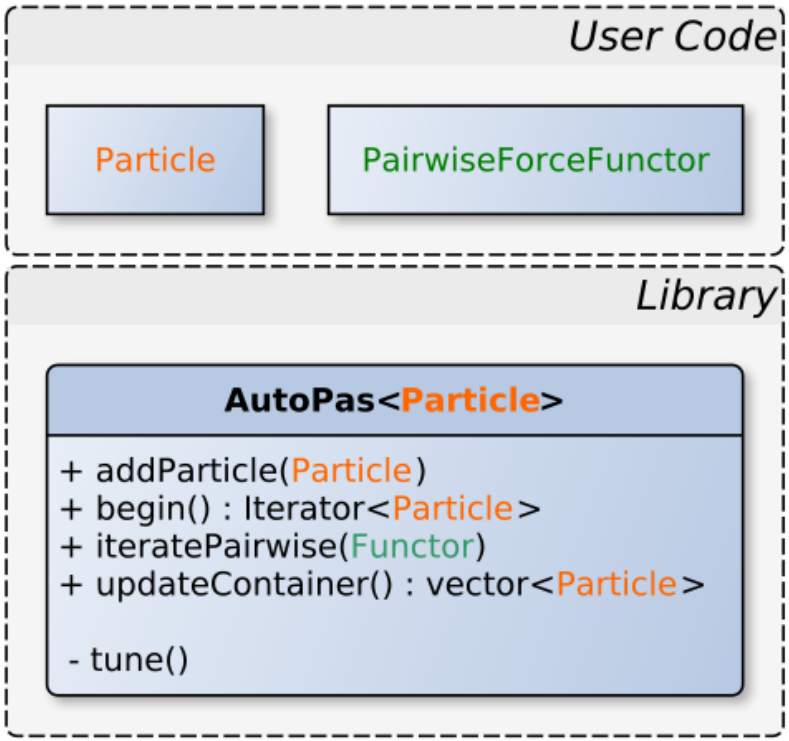
- AutoPas is an auto-tuning library for the handling of particle interactions.
- It contains a collection of algorithms.
- It tests potential algorithms during tuning phases and uses chosen ‘best’ during subsequent non-tuning phases.

# AutoPas

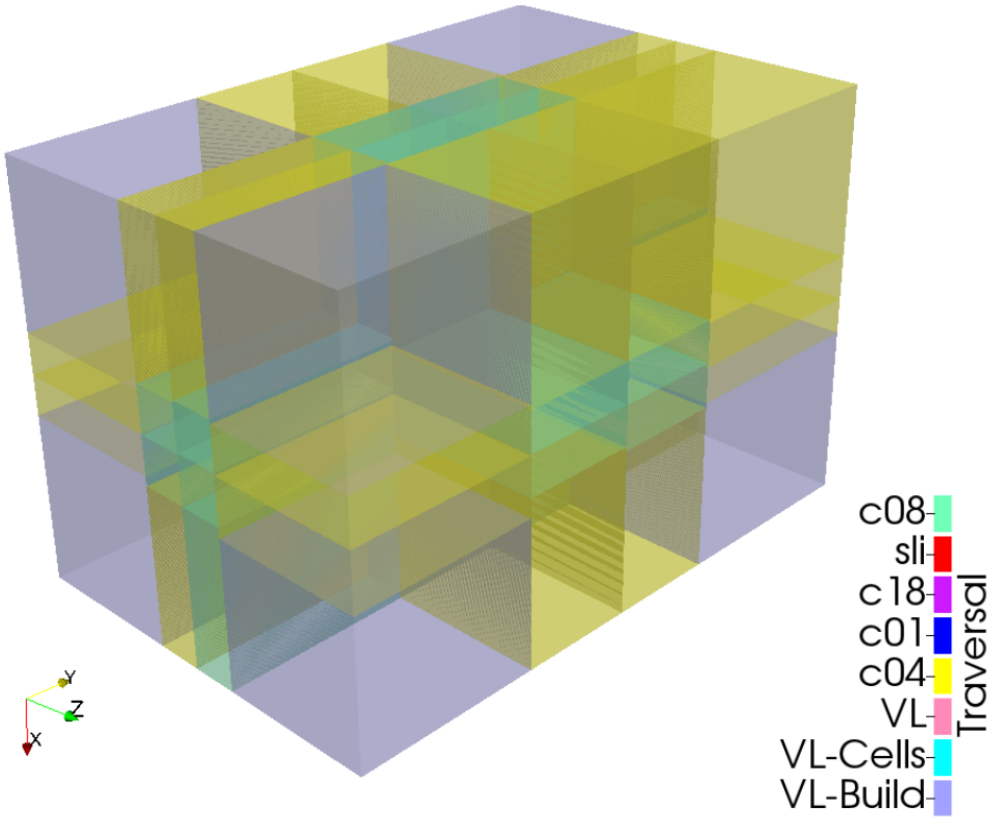
- AutoPas is an auto-tuning library for the handling of particle interactions.
- It contains a collection of algorithms.
- It tests potential algorithms during tuning phases and uses chosen ‘best’ during subsequent non-tuning phases.

# AutoPas

- All handling of algorithms and tuning is handled internally by AutoPas.
- The user must provide particle and pairwise force functor classes, that inherit from AutoPas'.



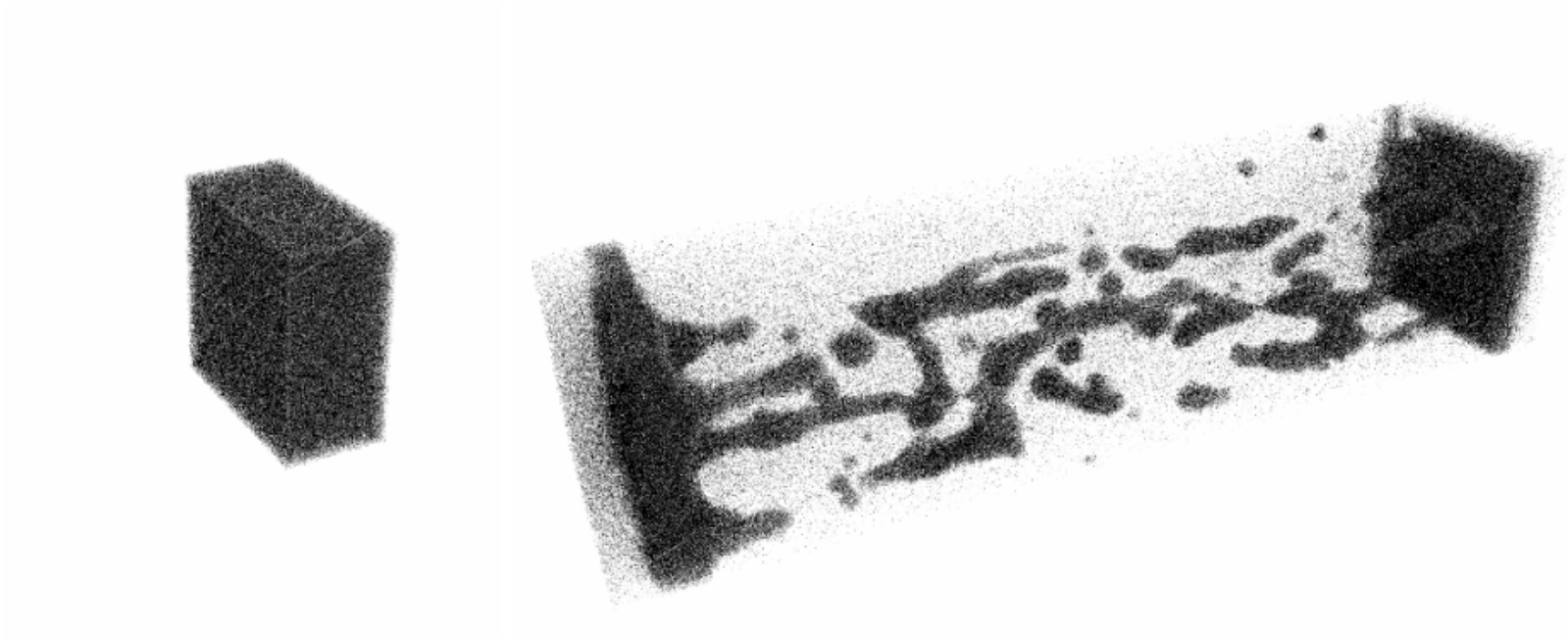
# AutoPas: Distributed Memory



# AutoPas: Use Cases

- ls1-mardyn: MD Simulator
- LADDS: Large-scale Deterministic Debris Simulation

# Single site experiment: Exploding Liquid



# Single site experiment: Exploding Liquid

- Single node, 16 threads
- 40000 time steps
- Comparing tuning every 2500 steps Vs Once at beginning

# Single site experiment: Exploding Liquid

| | Multiple Full-Search | Single Full-Search |
|---|---|---|
| Total | 9.130 | 9.474 |
| Tuning | 2.861 | 0.713 |
| (per iteration) | 0.001144 | 0.0044562 |
| Non-tuning | 6.269 | 8.761 |
| (per iteration) | 0.0001672 | 0.0002199 |

- With tuning throughout simulation, we get better performance during non-tuning.
- But we lose a lot of time to tuning.

# The problem

- AutoPas can perform significantly worse than sticking to a single value.
- Full-search tuning requires testing many sub-optimal algorithms.
- Some smart tuning algorithms exist, but they are still relatively poor.

# Acknowledgments

Contributors to the AutoPas project:

- Fabio Gratl, Technical University of Munich
- Hans-Joachim Bungartz, Technical University of Munich
- Philipp Neumann, Helmut-Schmidt University

Developments are supported by the project: “MaST: Macro/Micro-Simulation of Phase Separation in the Transcritical Regime” of the Digitalization and Technology Research Center of the Bundeswehr (dtec.bw)

# Citations

[1] F. A. Gratl, S. Seckler, H.-J. Bungartz and P. Neumann: N Ways to Simulate Short-Range Particle Systems: Automated Algorithm Selection with the Node-Level Library AutoPas, In Computer Physics Communications, Volume 273, 2022. [2] F.

A. Gratl, S. Seckler, N. Tchipev, H.-J. Bungartz and P. Neumann: AutoPas: Auto-Tuning for Particle Simulations, In 2019 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), Rio de Janeiro, May 2019. [3] S. Seckler, F. Gratl, M. Heinen, J. Vrabec, H.-J. Bungartz,

P. Neumann: AutoPas in ls1 mardyn: Massively parallel particle simulations with node-level auto-tuning, In Journal of Computational Science, Volume 50, 2021.

Any Questions?

# Kokkos

- GPUs can provide a lot of benefits to particle simulations.
- We are a team (currently) of 2 doctorate candidates ⇒ We cannot maintain different versions for different parallel frameworks.
- Kokkos provides a hardware agnostic solution - by providing a layer of abstraction.
- This is not yet fully implemented.