Automated Algorithm Selection for Particle Simulations in Engineering Applications with AutoPas

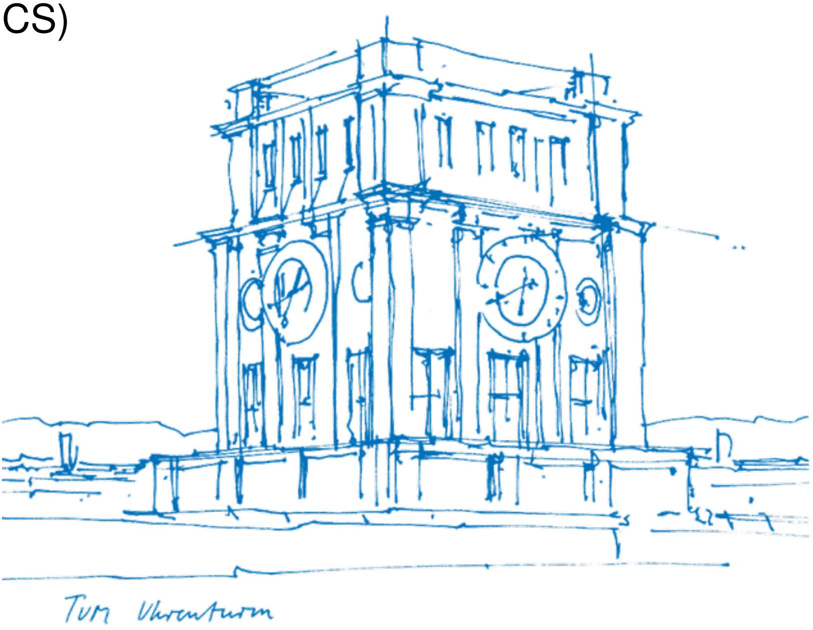
Fabio Gratl, Steffen Seckler, Hans-Joachim Bungartz, Philipp Neumann

Technical University of Munich

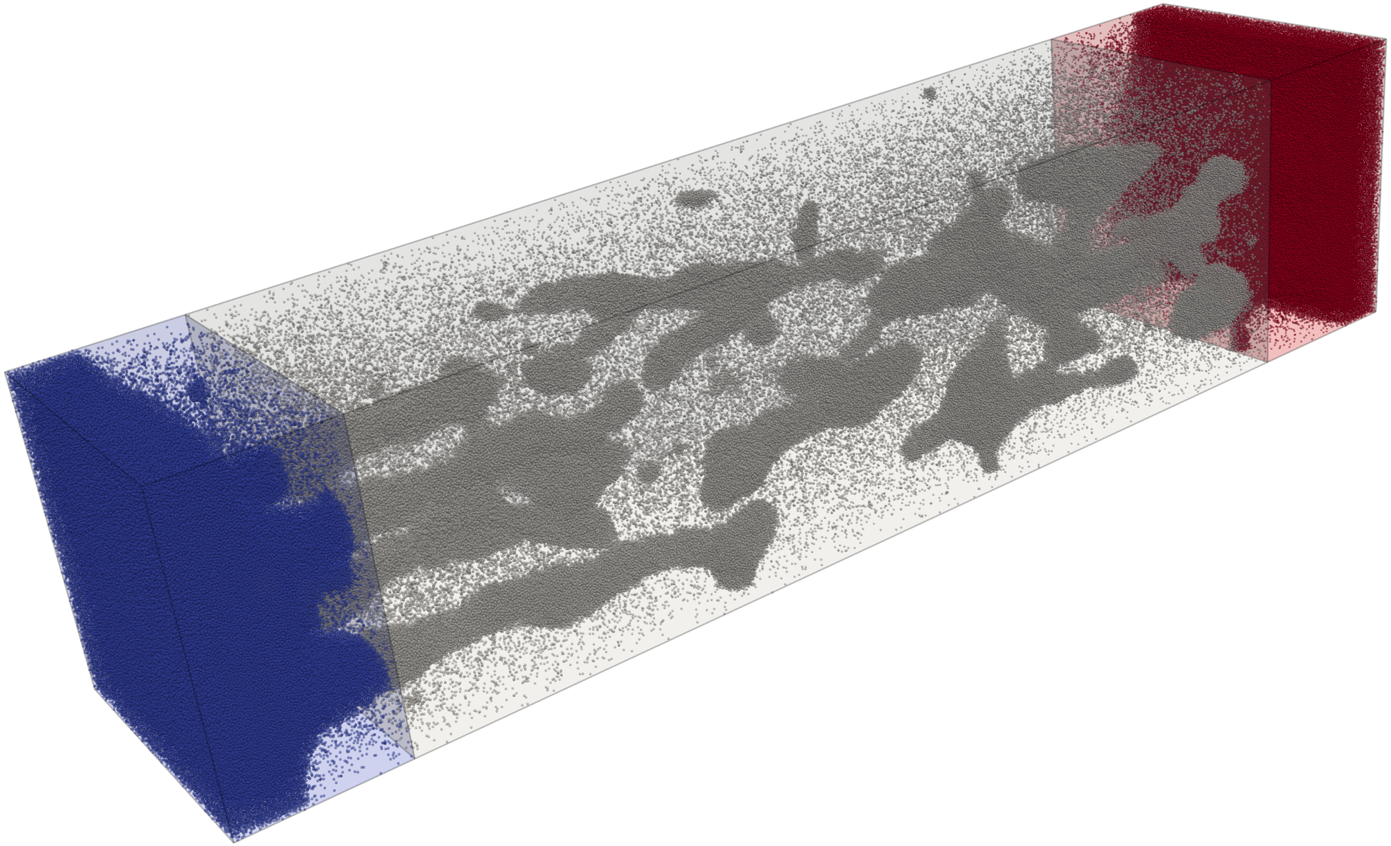
Department of Informatics

Chair of Scientific Computing in Computer Science (SCCS)

Hamburg, 04.10.2021

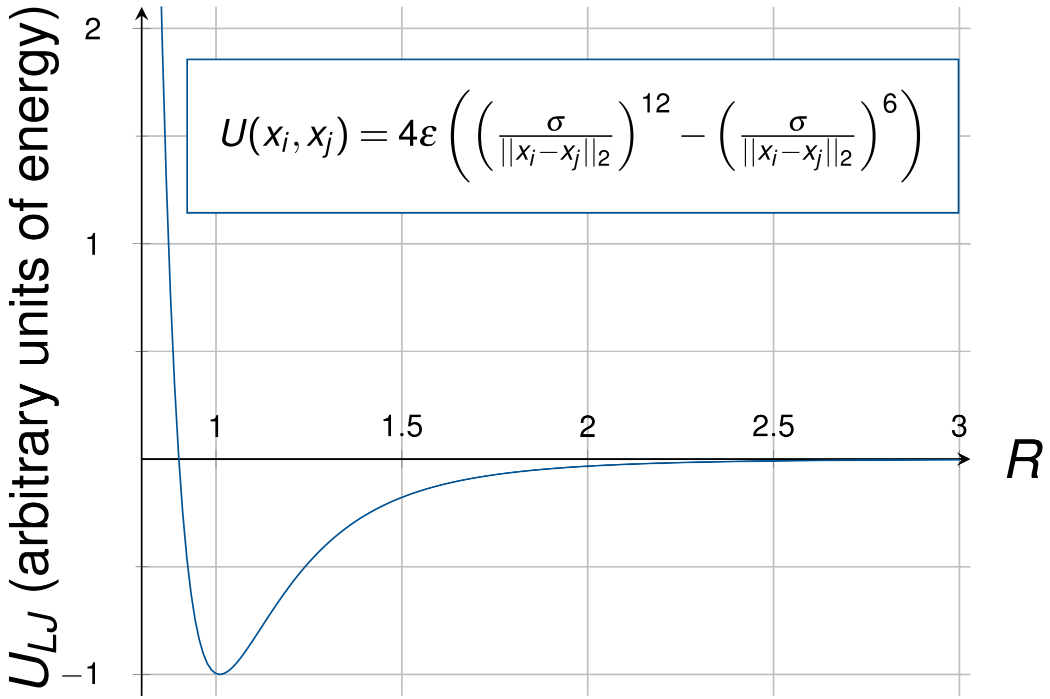


Motivation



Molecular Dynamics - Short Range

- Here: small rigid molecules as points without geometry
- Simulation of movement of particles
- Computation of pairwise forces
- Newton's Laws of Motion
- N-Body problem $\Rightarrow O(N^2)$
- Force cut-off $\Rightarrow O(N)$

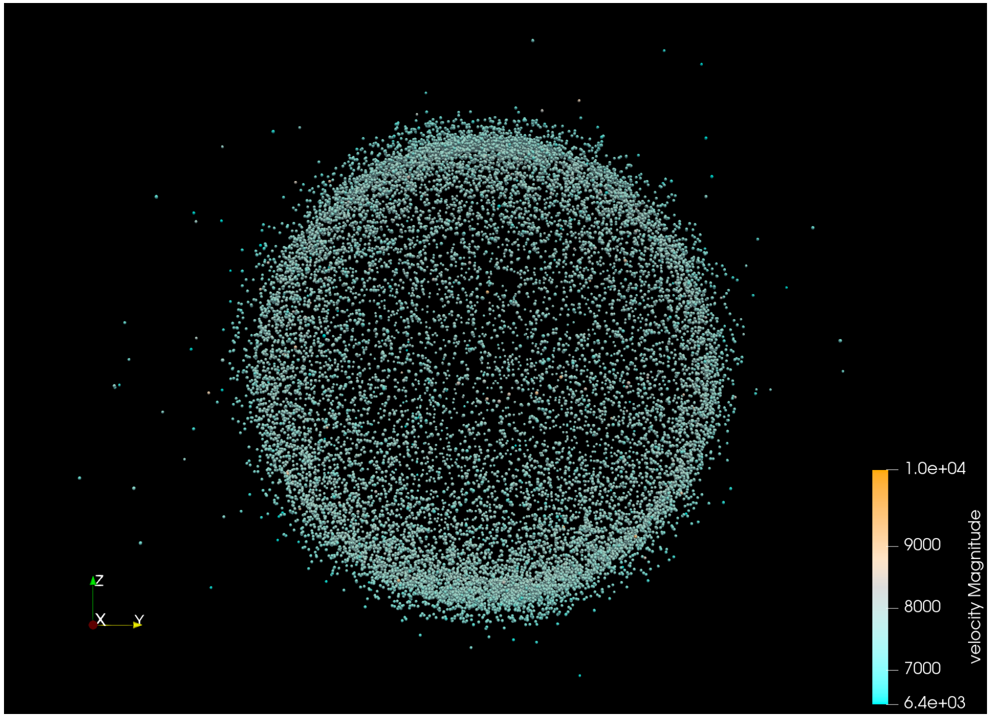


$$U(x_i, x_j) = 4\varepsilon\left(\left(\frac{\sigma}{||x_i - x_j||_2}\right)^{12} - \left(\frac{\sigma}{||x_i - x_j||_2}\right)^{6}\right)$$

Figure: Lennard Jones Potential for $\varepsilon = 1$ and $\sigma = 0.9$

Challenges

- Total number of particles
- Particle density
- (In-)Homogeneity
- Systems changing over time
- Many possible algorithms
- Arbitrary simulation types: MD, SPH, DEM, ...
- Overall goal: Minimize time to solution!

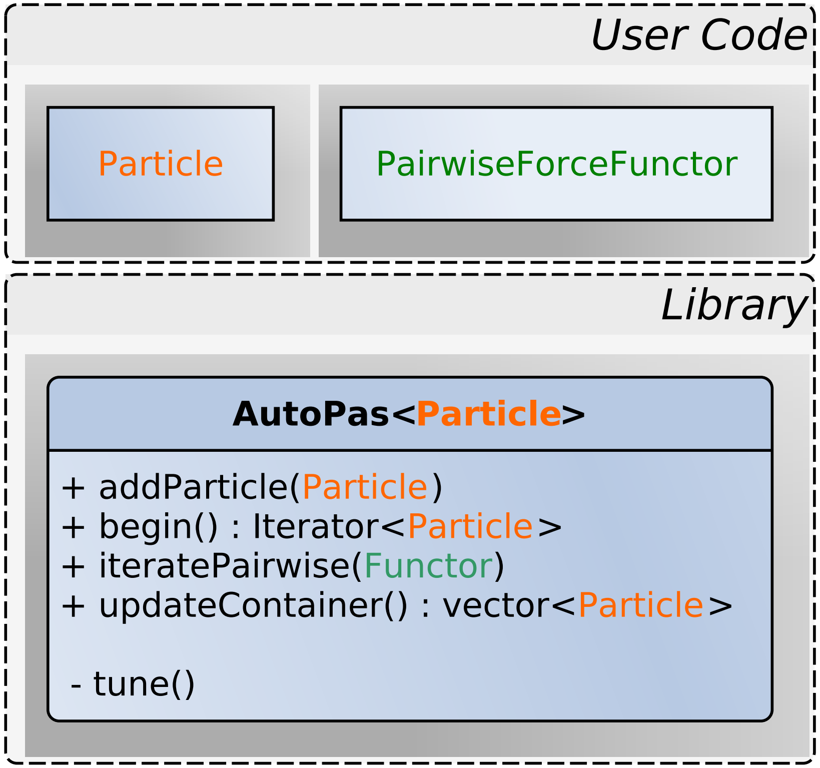


Introducing AutoPas

- Node-Level `C++17` library
- Black-box particle container
- Facade-like software pattern
- User defines:
  - Properties of particles
  - Force for pairwise interaction
- AutoPas provides
  - Containers, Traversals, Data Layouts, ...
  - Dynamic Tuning at run-time

⇒ General base for N-Body simulations

https://autopas.github.io/



User Code

Particle

PairwiseForceFunctor

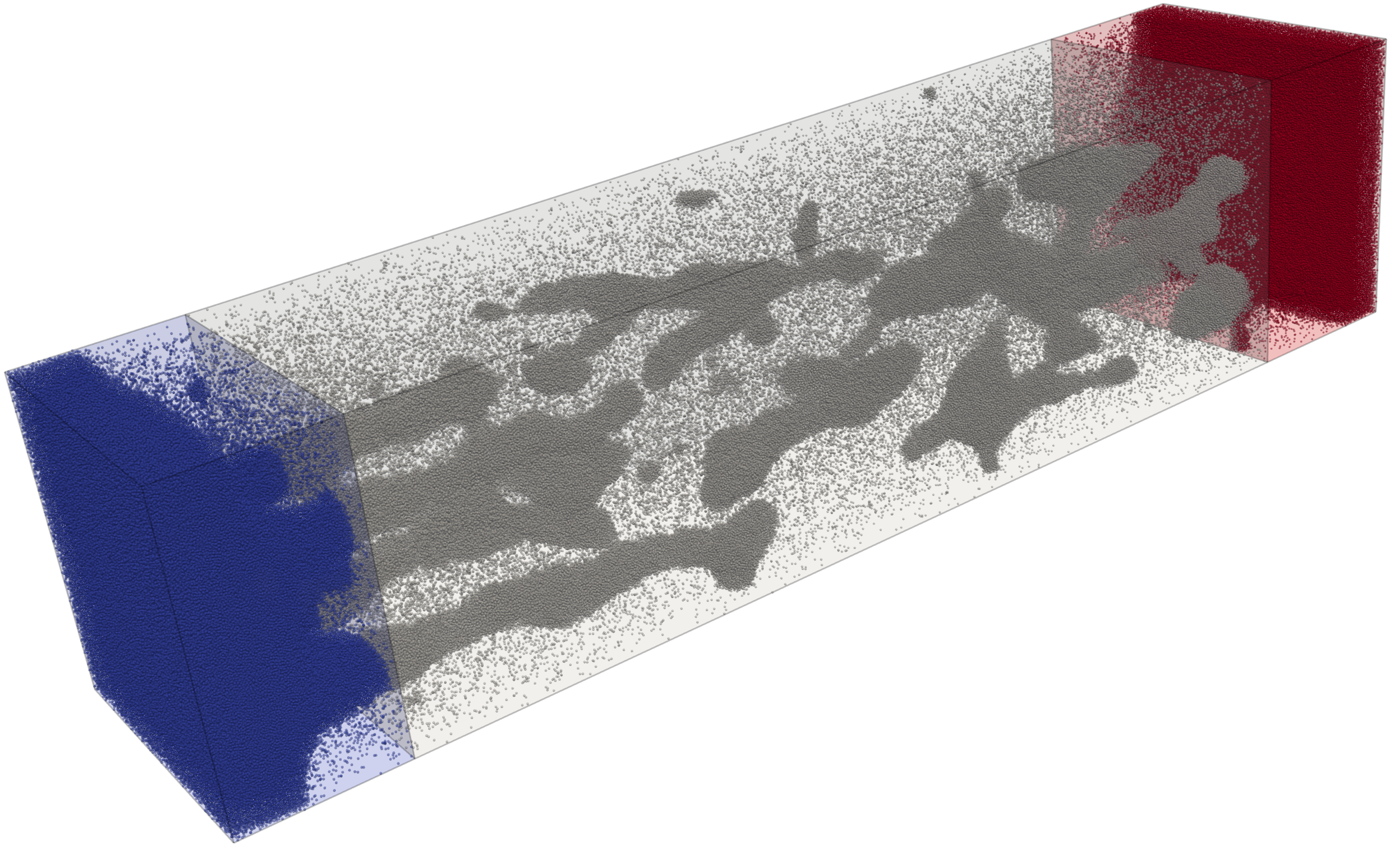
Library

AutoPas<Particle>

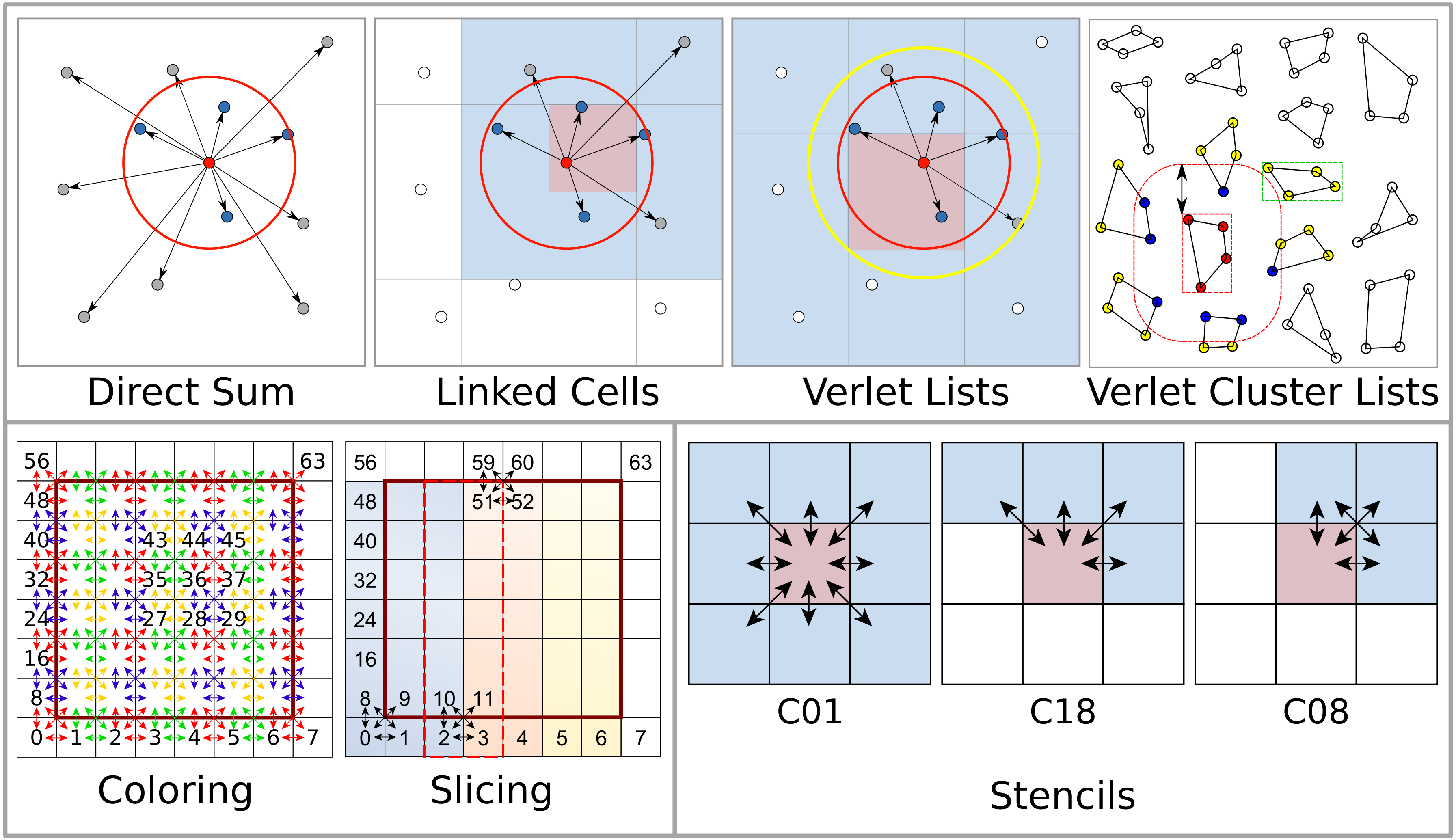
+ addParticle(Particle)
+ begin() : Iterator<Particle>
+ iteratePairwise(Functor)
+ updateContainer() : vector<Particle>

- tune()

ls1 mardyn + AutoPas $\Rightarrow$ MPI + Tuning



Algorithmic Choices



Direct Sum

Linked Cells

Verlet Lists

Verlet Cluster Lists

Coloring

Slicing

C01

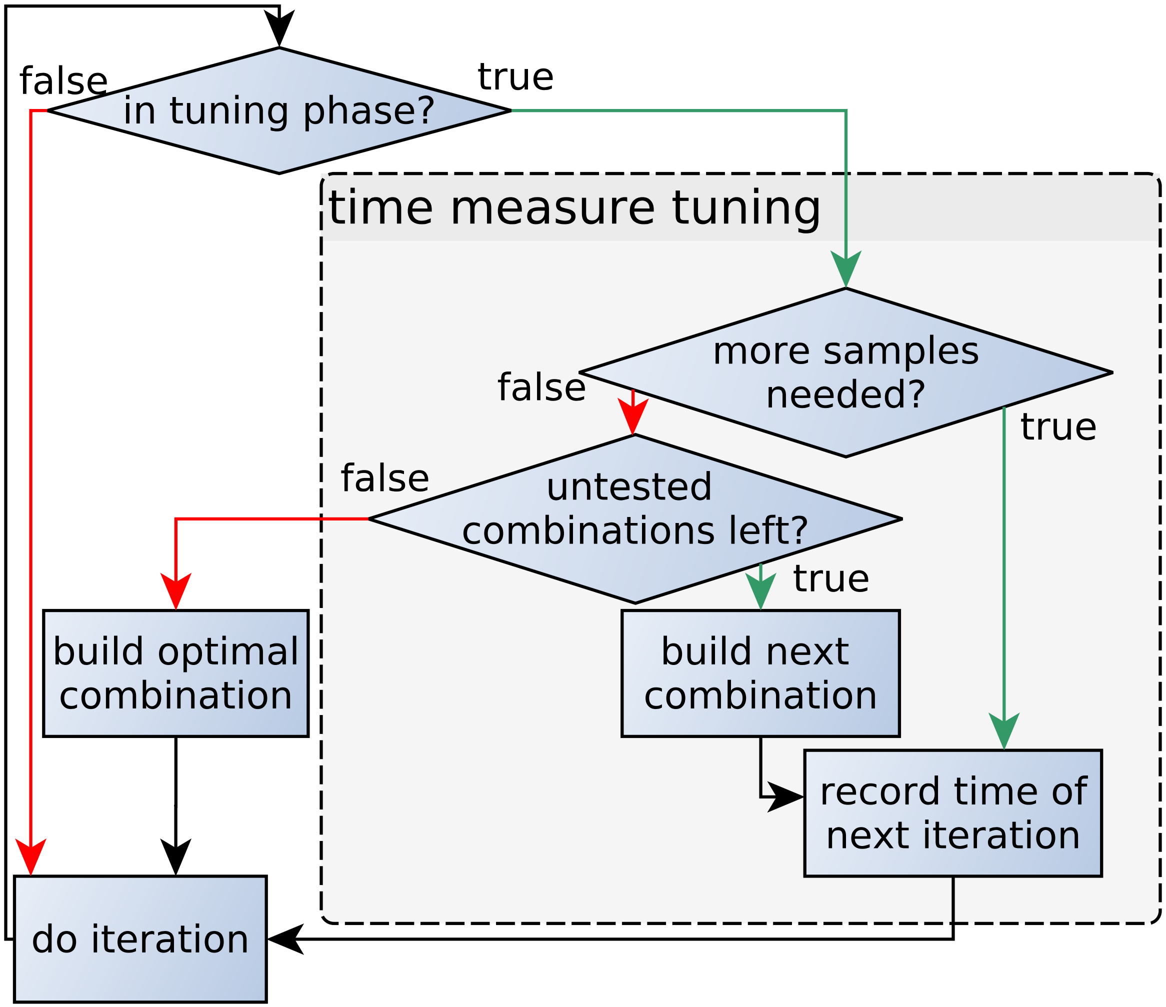
C18

C08

Stencils

Tuning Cycle

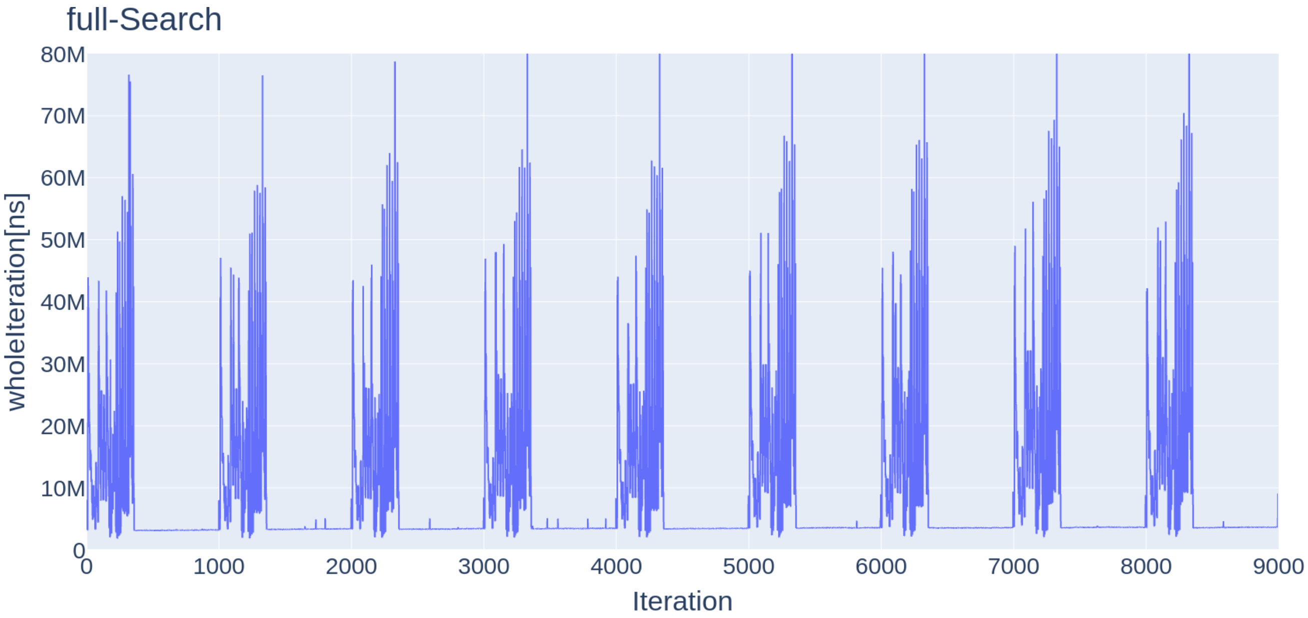
- Common interfaces for containers, traversals, etc
  ⇒ Strategy pattern
  ⇒ "Verlet-like" approach
- Repeated periodically
- User can restrict search space



Naive Approach

- Unordered search space
- Test every possible configuration
- Reevaluate periodically

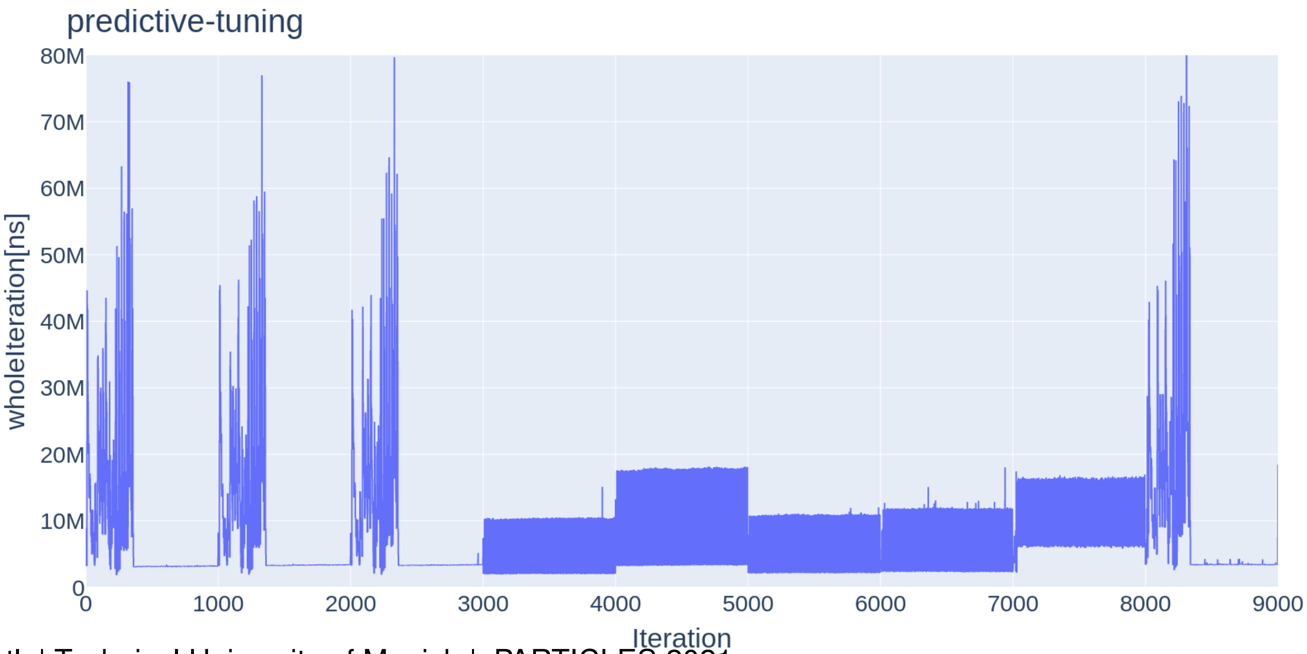
⇒ "Full Search"



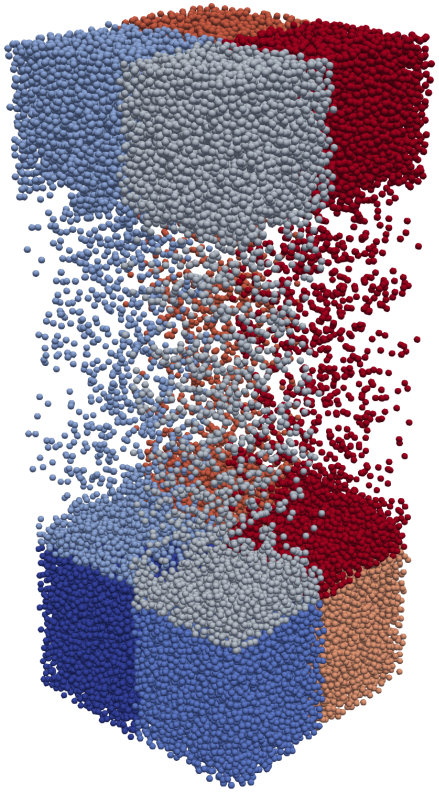
Improving Full Search

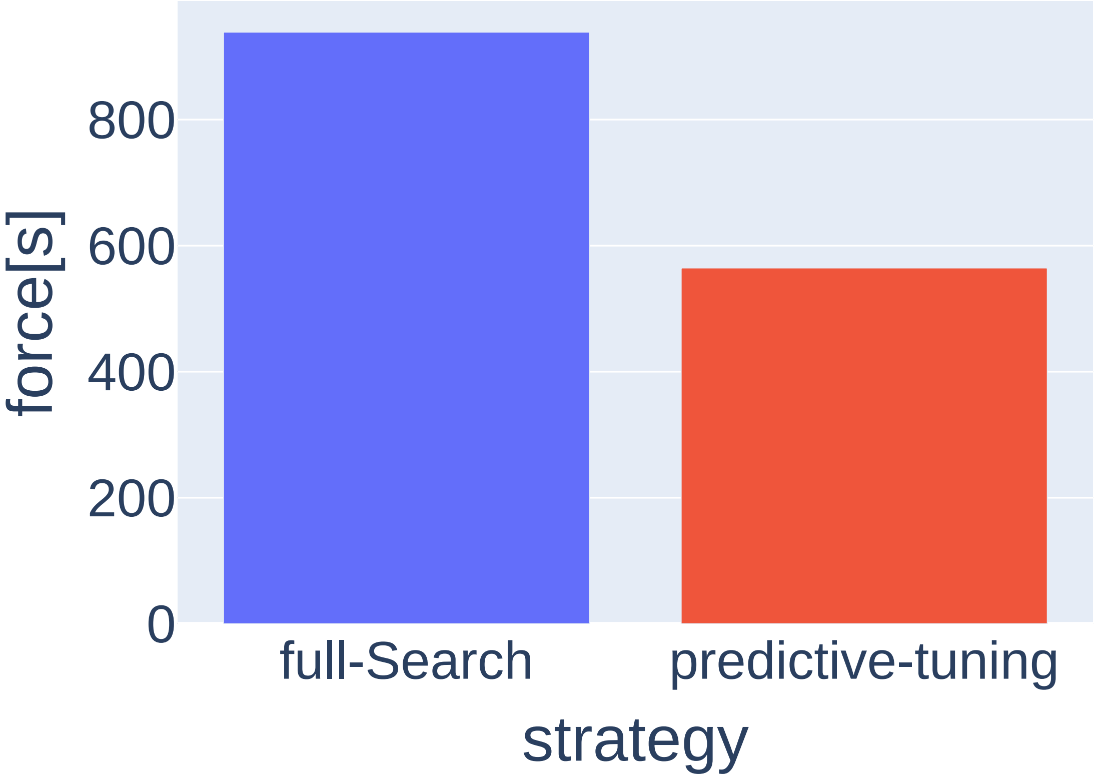
- Use first tuning phases for FS
- Extrapolate performance for rest
- Periodically re-test everything (to check divergence)
- Blacklist (for particularly slow methods)

⇒ Predictive Tuning



Predictive Tuning

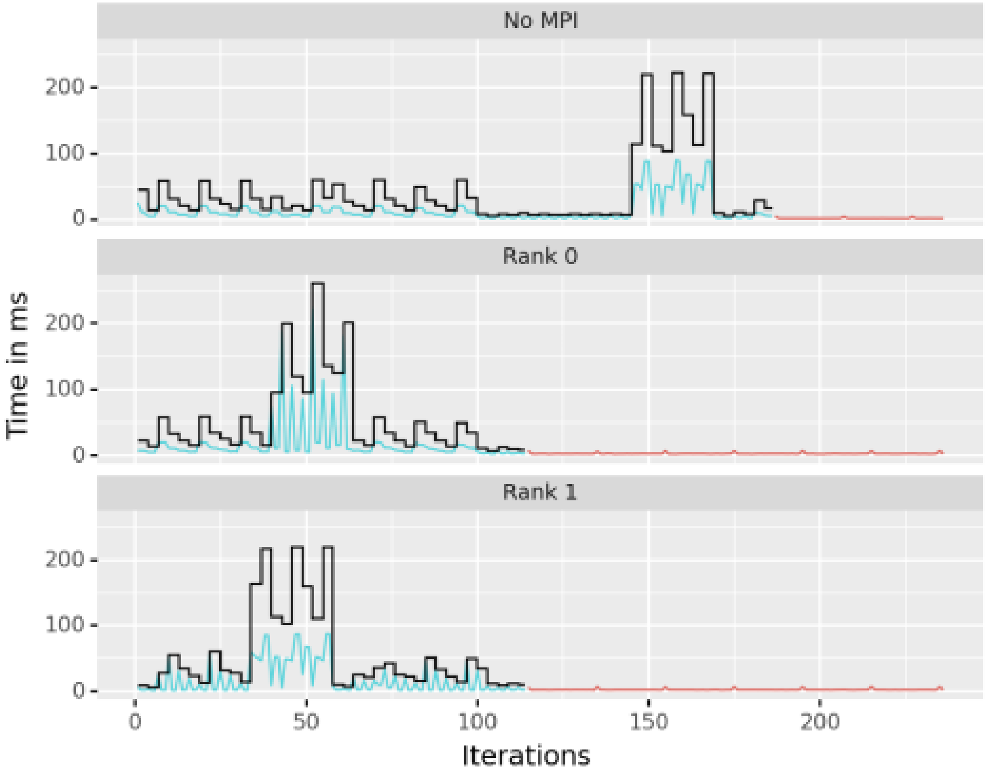




- Testing only promising configurations reduces runtime

WIP: Parallel Tuning

- Different configurations on all ranks and exchange measurements
- Problem: Inhomogeneous domains
- Solution: Bucketing by homogeneity and maximum density



WIP: Rule-based Tuning

- Use knowledge about algorithms or domain
- Formulate rules to restrict the search space
- Applicable to all Tuning Strategies
- Rules can be supplied by the user at run-time

Summary

- AutoPas enables automated algorithm selection on node level.
- Independent tuning of all MPI ranks.
- Optimal algoritm choices also for non-expert users.

... and future work

- Exploiting MPI parallelism.
- Reduction of search space.