Chair of Scientific Computing
Department of Informatics
Technical University of Munich

Scientific Computing
Department of Informatics
Universität Hamburg

Auto-Tuning for Short-range Particle Simulation

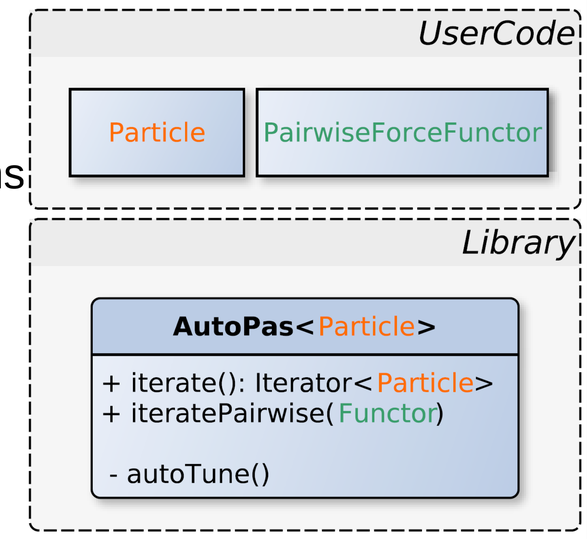
Fabio Gratl, Steffen Seckler, Nikola Tchipev, Philipp Neumann, Hans-Joachim Bungartz
{f.gratl,s.seckler,n.tchipev,bungartz}@tum.de, philipp.neumann@uni-hamburg.de

Abstract

In N-Body simulations, the time-to-solution can change drastically depending on simulation parameters and the configuration of the scenario. Different algorithms or parallelization patterns can be applied to improve the performance for a specific scenario. However, no combination is optimal for every situation. Highly impactful characteristics like the particle distribution or local densities can change significantly over the course of simulations. This means that the optimal algorithm combination can change anytime during the simulation.
To tackle this problem of specialization, we created the C++ library *AutoPas* [1]. It is designed to act as a base layer for arbitrary N-Body simulations while obtaining optimal node-level performance through auto-tuning. The library determines the best performing combination of algorithms for finding particle pairs, data structures, and OpenMP parallelization patterns. Should the optimum change, the library alters the combination at runtime.
We integrated *AutoPas* into the software *ls1 mardyn* [2,3], a molecular dynamics simulator for large numbers of small rigid molecules.
This poster showcases the concepts behind *AutoPas* and how to integrate it into existing code bases. Furthermore, we show run time results for a spinodal decomposition scenario in which we achieve and sustain optimal node-level performance by this auto-tuning approach.

AutoPas Overview

- Node-Level C++ header lib
- User defines:
  - Properties of particles
  - Pairwise particle interactions
- *AutoPas* provides:
  - Containers: Direct sum, linked cells, Verlet lists, cluster lists
  - (Non-)Load balanced traversals: Coloring, locks, buffers
  - Data layouts: AoS, SoA
- Dynamic tuning at run-time

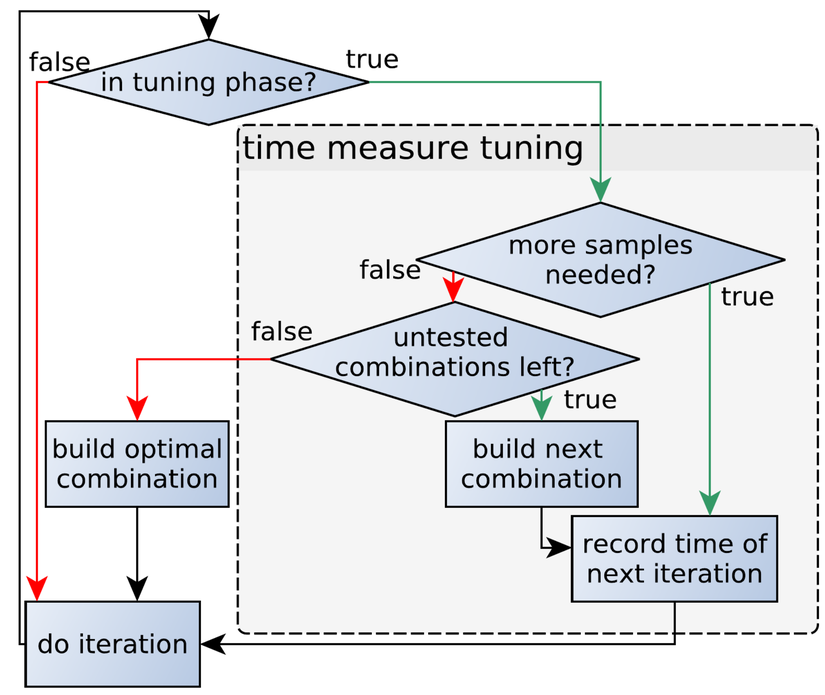


Integration in *ls1 mardyn*

- *ls1 mardyn*: simulation software for large systems of rigid molecules
- New molecule class to interface *ls1 mardyn* and *AutoPas* particle descriptions
- Wrapper for main *AutoPas* interface to replace original particle container
- Challenge: Distributed-memory support (potentially different *AutoPas* containers on different ranks)
  → subject to current work

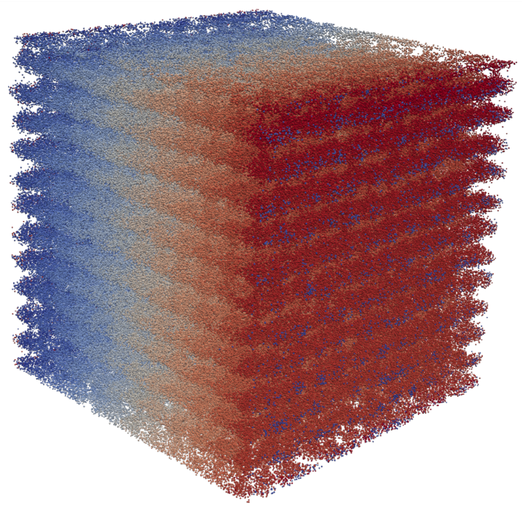
References:
[1] F. Gratl, S. Seckler, N. Tchipev, H.-J. Bungartz, P. Neumann. *AutoPas: Auto-Tuning for Particle Simulations*. Accepted for IPDPS 2019 proceedings, 2019
[2] C. Niethammer et al. *ls1 mardyn: The massively parallel molecular dynamics code for large systems.* Journal of Chemical Theory and Computation 10(10):4455-4464, 2014
[3] N. Tchipev et al. *TweTriS: Twenty trillion-atom simulation*. International Journal of High Performance Computing Applications, published online, 2019

Auto-Tuning

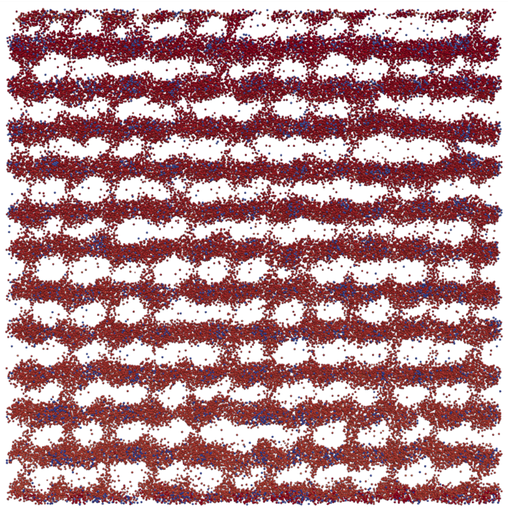


- Common interfaces for particle containers, particle-pair traversals, OpenMP parallelization scheme, etc.
  → strategy pattern
- Periodic testing (every N steps) of various combinations
- Restriction of test space via user
- Outlook: Machine learning, Bayesian statistics

Spinodal Decomposition

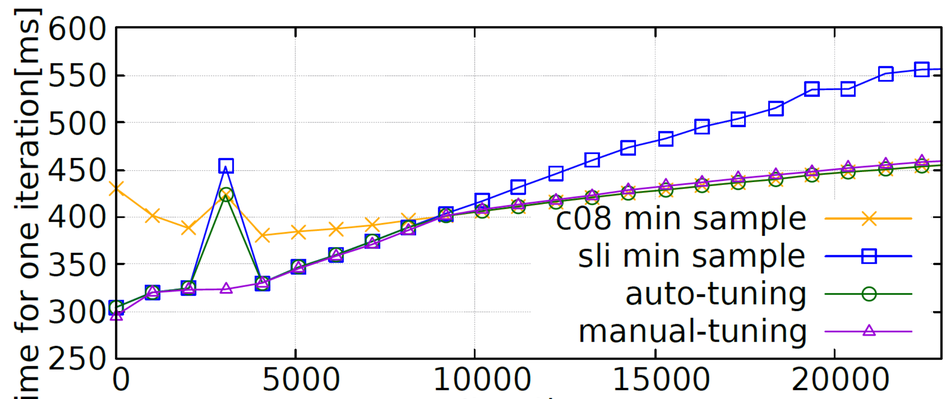


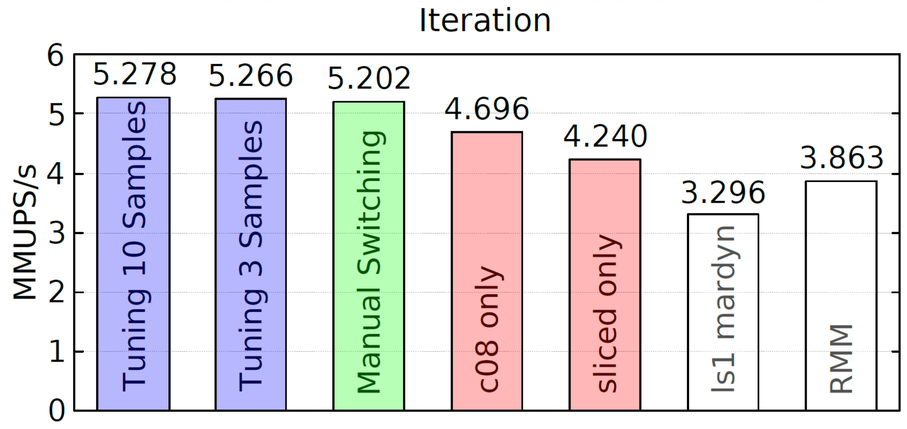
3D view



2D slice

- 4M particles, 240x240x240 domain, cutoff radius rc=2.5
- Hardware: Haswell (SuperMUC, Phase 2), 28 threads
- *AutoPas* configuration: linked cells/ 8-way coloring (c08), lock-based slicing (sli)





We thank the Intel Parallel Computing Center “ExScaMIC-KNL” and the Federal Ministry of Education and Research, Germany, project “Task-based load balancing and auto-tuning in particle simulations” (TaLPas), grant number 01IH16008 for financial support of this research.

SPONSORED BY THE
Federal Ministry of Education and Research