Chair of Scientific Computing in Computer Science
Univ.-Prof. Dr. Hans-Joachim Bungartz

Technische Universität München

Seminar Final Presentation

Algorithm Selection and Auto-Tuning in AutoPas

Manuel Lerchner
manuel.lerchner@tum.de

Table of Contents

Molecular Dynamics Simulation Challenges

- Complex interaction models
- Enormous numbers of particles
- High computational requirements
- Need for highly optimized algorithms

Traditional MD Engines vs AutoPas

Traditional Engines (GROMACS, LAMMPS, ls1 mardyn):

- Single, highly optimized implementation
- Static optimizations
- Hardware-specific tuning required

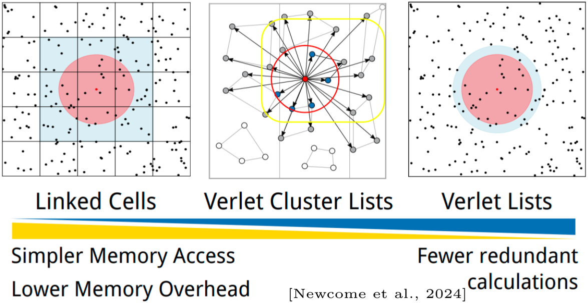
AutoPas Approach:

- Dynamic optimizations
- Adapts to simulation state
- Hardware-independent

Table of Contents

What is AutoPas?

- Library for arbitrary N-body simulations
- Optimal performance by switching implementations
  - Container: Finding neighboring particles
  - Traversal: Parallel force calculations
  - Data Layout: Memory access optimization
  - Newton 3: Force calculation optimization



[Newcome et al., 2024]

Structure of AutoPas

- Three main areas:
  - User Application
  - Algorithm Library
  - Tuning Strategies
- Algorithm Library:
  - Huge Search Space[1]
- Tuning Strategies:
  - Full Search
  - Random Search
  - Predictive Tuning
  - Bayesian Search
  - Rule Based Tuning

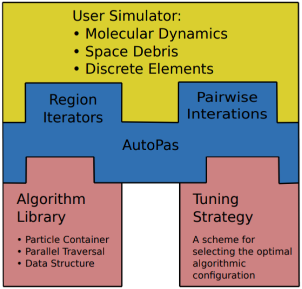


Figure: [Newcome et al., 2023]

[1] Container × Traversal × Data Layout × Newton 3 × Load Estimator × Cell Size Factor

Container Types

- Linked Cells
- Verlet Lists
- Verlet Cluster Lists

Traversal Patterns

- C01: Independent cell processing
- C18: Color-based with Newton 3
- C08: Reduced synchronization points

Table of Contents

Tuning Strategies

- FullSearch: Evaluates all configurations
- RandomSearch: Random sampling
- BayesianSearch: Optimization-guided selection
- PredictiveTuning: Performance prediction
- RuleBasedTuning: Expert knowledge rules
- FuzzyTuning: Fuzzy logic evaluation

Benefits of Auto-Tuning

- Significant performance improvements
- Out-of-the-box optimization
- Hardware independence
- User-friendly interface
- Easy integration with other frameworks

Challenges of Auto-Tuning

- Overhead from evaluating suboptimal configurations
- Unnecessary periodic re-tuning
- Complex scenarios needed to show benefits

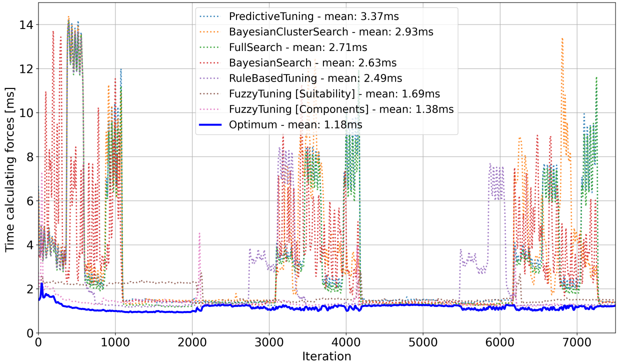
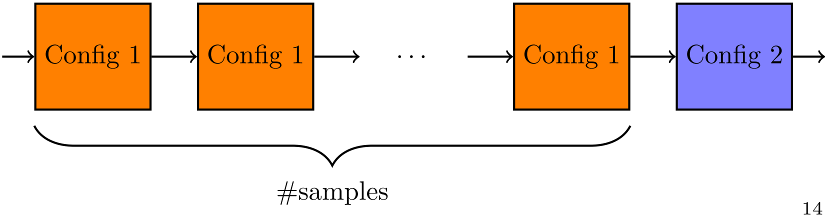


Table of Contents

Early Stopping Idea

- Each configuration gets evaluated multiple times
- Average performance is used for the final decision
- Optimization: Abort evaluation early if performance is poor
    - Stop after first poor sample
    - Stop during evaluation



Implementation in AutoPas

- Keep track of best performance seen so far
- Stop resampling if current performance exceeds optimal by a certain factor
- Hyperparameter: allowedSlowdownFactor
- Which allowedSlowdownFactor should be used?
  - allowedSlowdownFactor $\to 1$: Many spurious aborts
  - allowedSlowdownFactor $\to \infty$: No early stopping

Early Stopping Results [PredictiveTuning]

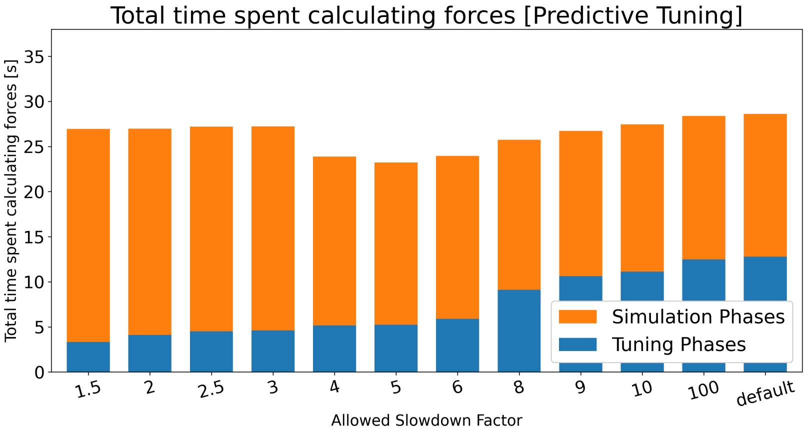


Figure: Total Simulation Time for Exploding Liquid Simulation for different

Early Stopping Results Analysis

- Reduction in total simulation time:
  - FullSearch: 14.8% reduction
  - PredictiveTuning: 18.9% reduction
- Optimal threshold around 4.5
- Never significantly increased simulation time

Conclusion

- AutoPas offers flexible, adaptive optimization
- Early stopping mechanism shows promise
- Trade-off between:
  - Specialized static optimization
  - Dynamic adaptability
- Complementary approaches possible

Thank you for your attention!

Questions?

References I

Newcome, S. J., Gratl, F. A., Muehlhaeusser, M., Neumann, P., and Bungartz, H.-J. (2024).
Autopas: Dynamic algorithm selection in molecular dynamics for optimal time and energy.
In SIAM Conference on Parallel Processing (PP24). SIAM.

Newcome, S. J., Gratl, F. A., Neumann, P., and Bungartz, H.-J. (2023).
Towards the smarter tuning of molecular dynamics simulations.
Amsterdam, The Netherlands.

Backup:

- A
- B