

---

## Bachelor Thesis Final Presentation

# Exploring Fuzzy Tuning Technique for Molecular Dynamics Simulations in AutoPas

**Manuel Lerchner**  
manuel.lerchner@tum.de

Advisors:  
**Manish Kumar Mishra, M.Sc.**  
**Samuel James Newcome, M.Sc.**

# Table of Contents

1 Introduction

2 Implementation

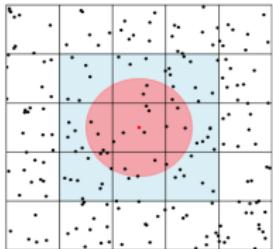
3 Benchmarks

4 Conclusion

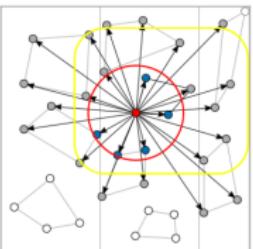
# AutoPas

## What is AutoPas?

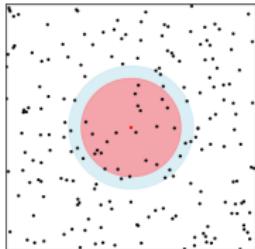
- Library for arbitrary N-body simulations
- Optimal performance by switching implementations
  - **Container:** Finding neighboring particles
  - **Traversal:** Parallel force calculations
  - **Data Layout:** Memory access optimization
  - **Newton 3:** Force calculation optimization



Linked Cells



Verlet Cluster Lists



Verlet Lists

Simpler Memory Access

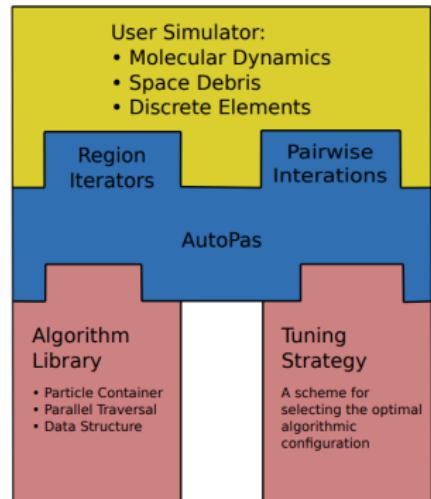
Lower Memory Overhead

Fewer redundant  
calculations

[Newcome et al., 2024]

## Structure of AutoPas

- Three main areas:
  - User Application
  - Algorithm Library
  - Tuning Strategies
- Algorithm Library:
  - Huge Search Space<sup>1</sup>
- Tuning Strategies:
  - Full Search
  - Random Search
  - Predictive Tuning
  - Bayesian Search
  - Rule Based Tuning

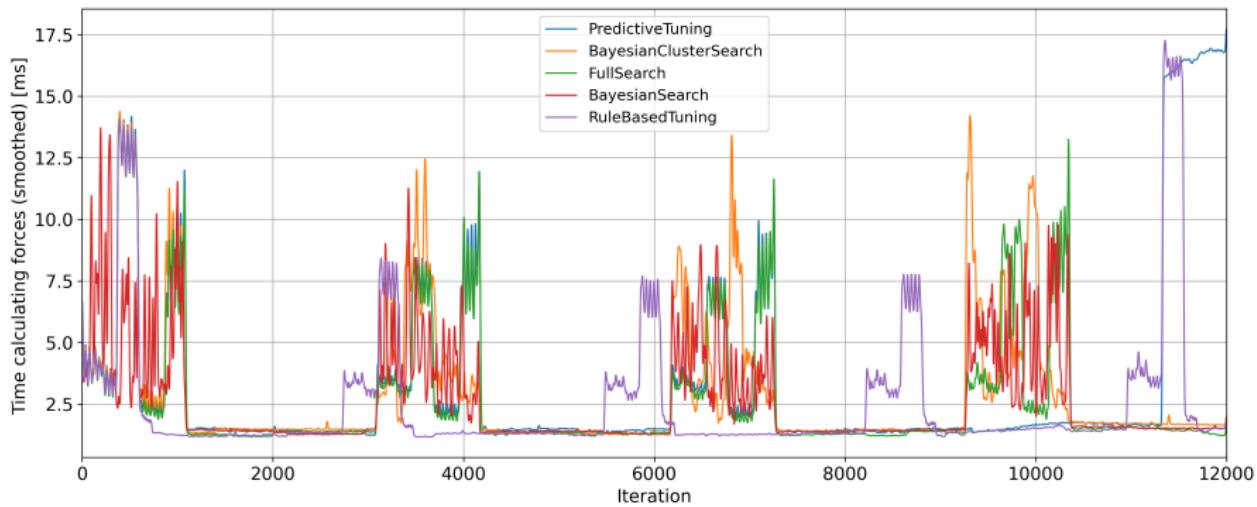


**Figure:** Source:  
[Newcome et al., 2023]

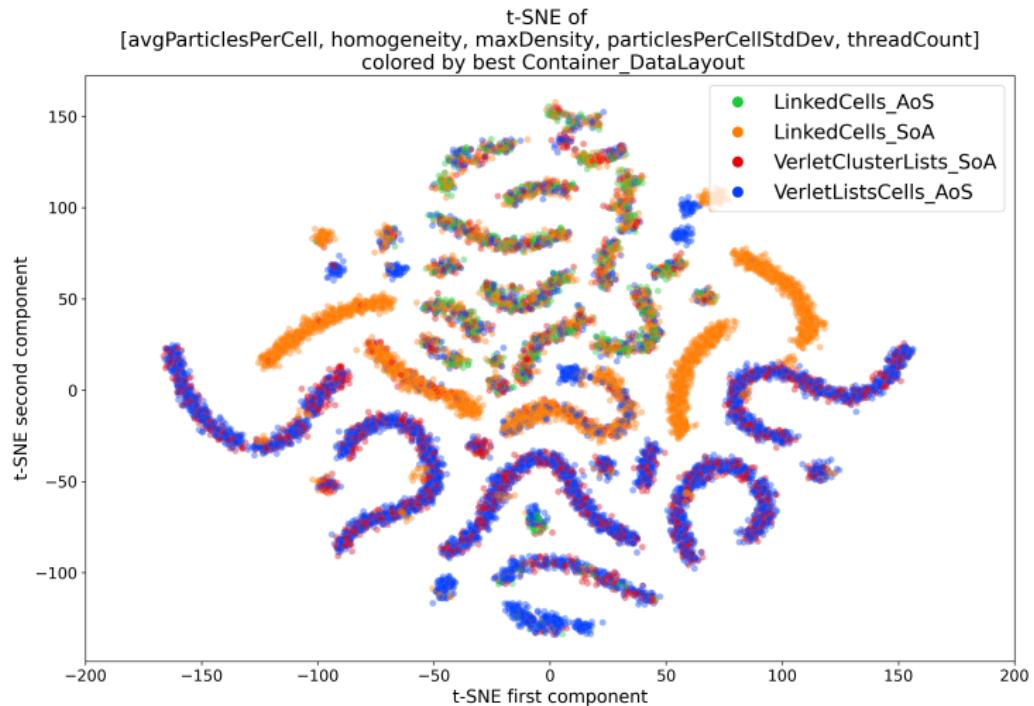
<sup>1</sup>Container × Traversal × Data Layout × Newton 3 × Load Estimator × Cell Size Factor

## Auto-Tuning

- Tuning Phase → Simulation Phase → Repeat
- Potential Tuning Overhead



## Idea: Tuning based on Simulation State



## Fuzzy Logic System

- (Fuzzy) Rule-based system
- Human-like reasoning to model systems  $f : \mathbb{R}^n \rightarrow \mathbb{R}$
- Linguistic terms instead of numerical values
  - What is *hot*? What is *cold*?
  - Allow smooth transitions between terms

### Example (Heater Control)

**Input:** temperature (e.g. 20°C), humidity (e.g. 60%)

**Output:** heater power (e.g. 50%)

**Rules:**

IF temp is <i>cold</i>	AND	humidity is <i>dry</i>	THEN power is <i>high</i>
IF temp is <i>hot</i>	OR	humidity is <i>wet</i>	THEN power is <i>low</i>
IF temp is <i>warm</i>			THEN power is <i>medium</i>

## Table of Contents

1 Introduction

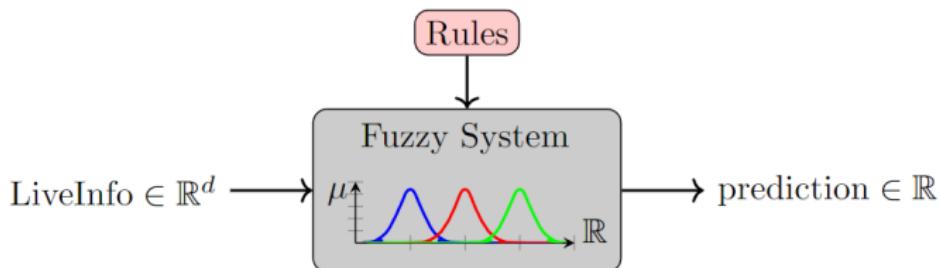
2 Implementation

3 Benchmarks

4 Conclusion

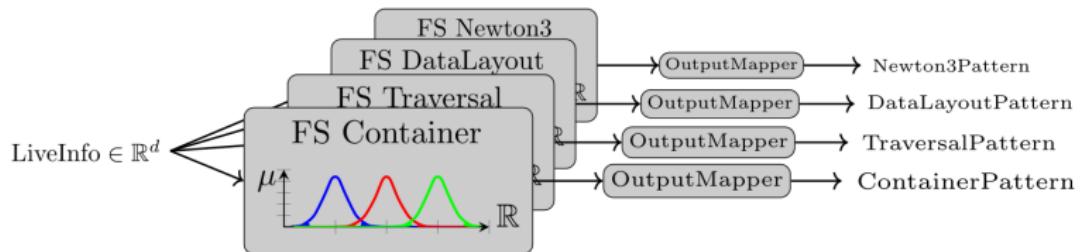
## Challenge 1: Numerical Output

- Fuzzy Tuning not directly applicable to AutoPas
  - Tuning is an *algorithm selection* problem
  - Fuzzy tuning provides functions  $f : \mathbb{R}^n \rightarrow \mathbb{R}$
  - Big Question: How to interpret the numerical output?



## Approach 1: Component Tuning

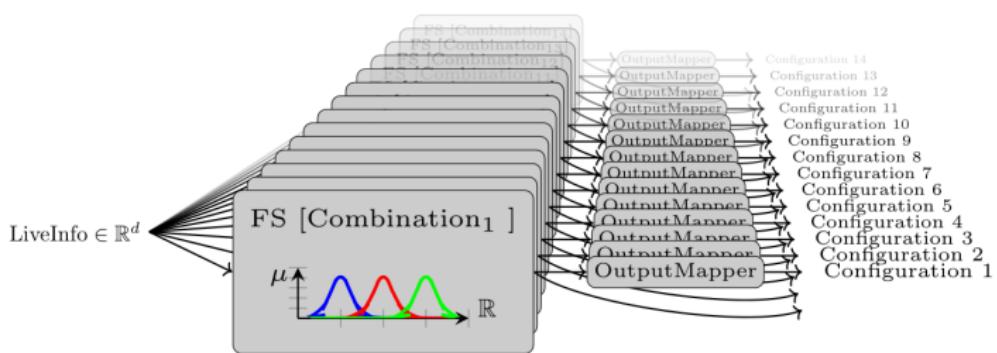
- Explicit Fuzzy System for each tunable parameter
  - Continuous representation of the parameter values



- Example: **IF** avgParticlesPerCell is *low* **AND** threadCount is *low*  
**THEN** traversal is *vcl\_c06*

## Approach 2: Suitability Tuning

- A Fuzzy System for **each** possible configuration
- Predict *suitability* of each configuration



- Example: **IF** `threadCount` is *high* **AND** `avgParticlesPerCell` is *low*  
**THEN** `suitability_LinkedCells_AoS_lc_c18_disabled` is *bad*

## Challenge 2: Expert Knowledge

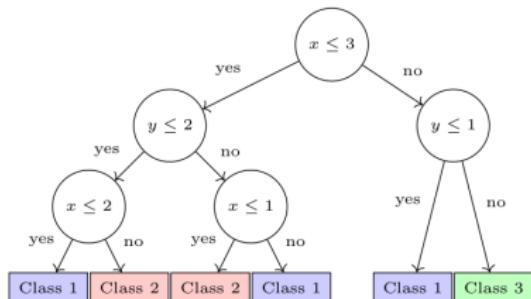
- How to design the Fuzzy System?
  - Creating it manually?
    - ✗ Requires extensive expert knowledge
    - ✗ Difficult to formalize non-trivial knowledge
  - Extracting rules from data?
    - ✓ No prior expert knowledge required
    - ✓ Semi-automated process

## Data-Driven Rule Extraction

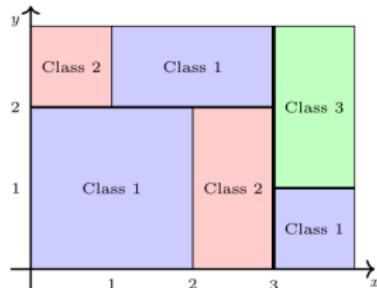
- Machine learning to generate rules [Crockett et al., 2006]
  1. Train Decision Tree on the data
  2. Decision Tree → Fuzzy Decision Tree
  3. Fuzzy Decision Tree → Fuzzy Rules
- Repeat to create overlapping rules (ensemble)

## Data → Decision Tree

- Separate classes recursively with axis-aligned splits
- Corresponds to nested *if-then-else* rules
- Tree contains the entire expert knowledge



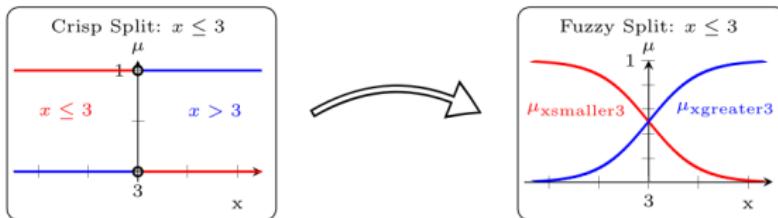
(a) Example decision tree

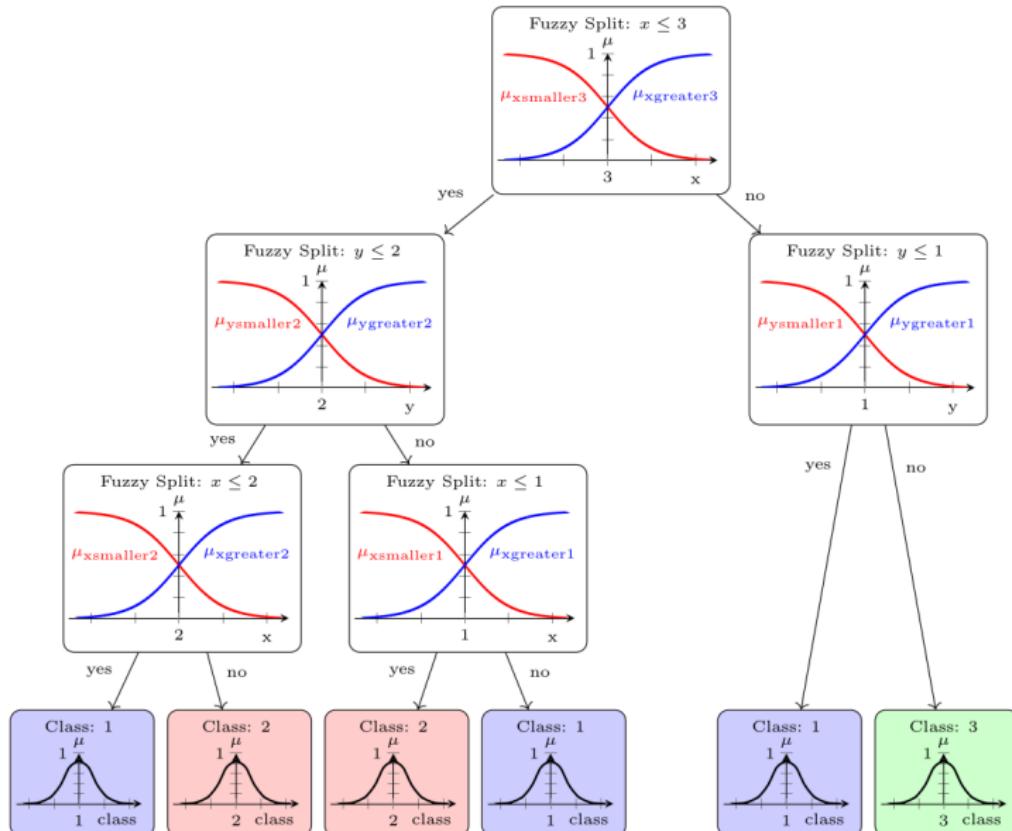


(b) Decision surface over  $\mathcal{D} = [0, 4] \times [0, 3]$

## Decision Trees → Fuzzy Decision Trees

- Conversion: Each (crisp) split is turned into two fuzzy sets
  - Provides robustness against noise
  - Both branches are considered



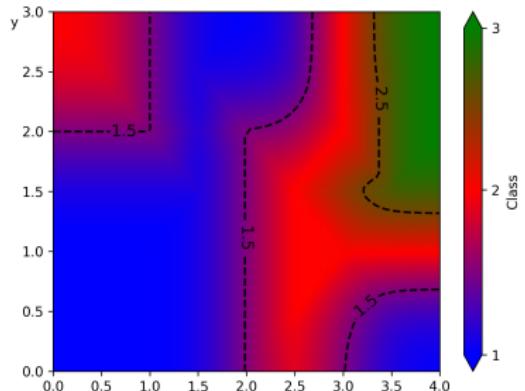


## Fuzzy Decision Trees → Fuzzy Rules

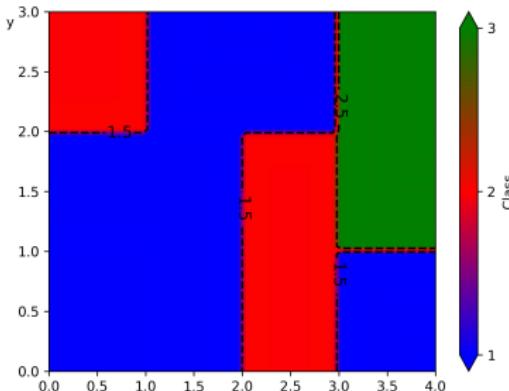
- Unnest the fuzzy decision tree

Rule	Antecedent	Consequent
1	$x \text{ is smaller3} \wedge y \text{ is smaller2} \wedge x \text{ is smaller2}$	$\text{class is 1}$
2	$x \text{ is smaller3} \wedge y \text{ is smaller2} \wedge x \text{ is greater2}$	$\text{class is 2}$
3	$x \text{ is smaller3} \wedge y \text{ is greater2} \wedge x \text{ is smaller1}$	$\text{class is 2}$
4	$x \text{ is smaller3} \wedge y \text{ is greater2} \wedge x \text{ is greater1}$	$\text{class is 1}$
5	$x \text{ is greater3} \wedge y \text{ is smaller1}$	$\text{class is 1}$
6	$x \text{ is greater3} \wedge y \text{ is greater1}$	$\text{class is 3}$

## Decision Surfaces



(a) Center of Gravity - Defuzzification



(b) Mean of Maxima - Defuzzification

Figure: Decision Surfaces of Fuzzy System

## Fuzzy Rule Extraction for `md_flexible`

- Collect dataset of `md_flexible` simulations
- For each tuning phase  $i$ , store:
  - `LiveInfoData`: `maxDensity`, `homogeneity`, `threadCount`, ...
  - `TuningData`: `Container`, `Traversal`, `Newton3`, ..., `Time`
- Introduce *relative speed* metric
  - Absolute time is not meaningful

$$\text{relative speed}_{\text{config}}^{(i)} = \frac{t_{\text{best}}^{(i)}}{t_{\text{config}}^{(i)}}$$

## Resulting Dataset

- Contains expected *relative speed* based on:
  - Simulation state
  - Configuration

ParticlesPerCell			Miscellaneous			Configuration			
avg	max	stddev	homogeneity	max-density	threads	Container DataLayout	Traversal	Newton3	Relative Speed
0.905	23	0.0129	0.0354	0.531	1	LinkedCells_AoS	lc_sliced	enabled	0.450641
2.201	13	0.0144	0.0861	0.627	24	VerletListsCells_AoS	vlc_sliced	disabled	0.594117
0.905	18	0.0136	0.0431	0.319	4	LinkedCells_AoS	lc_sliced_c02	enabled	0.454632
:	:	:	:	:	:	:	:	:	:

## Component Tuning - Rule Extraction

1. Combine *good* values for each simulation state
2. Extract rules for each parameter

ParticlesPerCell			Miscellaneous			Aggregated Configuration Terms		
avg	max	stddev	homogeneity	max-density	threads	Container DataLayout	Traversal	Newton3
0.906	15	0.015	0.055	0.297	4	"LinkedCells_SoA, VerletClusterLists_SoA, VerletListsCells_AoS"	"lc_sliced, lc_sliced_balanced, lc_sliced_c02"	"enabled"
0.945	25	0.041	0.084	0.673	24	"LinkedCells_SoA, VerletClusterLists_SoA, VerletListsCells_AoS"	"lc_c04, lc_c08, lc_sliced, lc_sliced_balanced"	"disabled, enabled"

Antecedent			Consequent	
avgParticlesPC	homogeneity	particlesPCStdDev	threadCount	Traversal
lower than 1.553	higher than 0.047	lower than 0.023	higher than 2.5	"lc_sliced, vlc_c18, lc_sliced_c02"
	lower than 0.037	lower than 0.023	lower than 26.0	"vcl_c06, vlc_c18, vlc_sliced_c02"
:	:	:	:	:

## Suitability Tuning - Rule Extraction

1. Assign suitability classes to each entry in the dataset
2. Group dataset by configuration
3. Rule Extraction for each configuration

ParticlesPerCell			Miscellaneous			Configuration				
avg	max	stddev	homogeneity	max-density	threads	Container DataLayout	Traversal	Newton3	Relative speed	Suitability
0.905	15	0.012	0.035	0.531	1	LinkedCells_AoS	lc sliced	enabled	0.450	"bad"
0.944	25	0.012	0.083	0.691	28	VerletClusterLists_AoS	vcl_c06	disabled	0.319	"poor"
0.944	20	0.012	0.079	0.041	12	LinkedCell_SoA	vlc_ sliced	enabled	0.989	"excellent"

Antecedent				Consequent
avgParticlesPC		homogeneity	particlesPCStdDev	threadCount
		lower than 0.084	higher than 0.029	higher than 26.0
		higher than 0.084	higher than 0.029	higher than 26.0
			higher than 0.02	lower than 2.5
⋮	⋮	⋮	⋮	⋮

## Table of Contents

1 Introduction

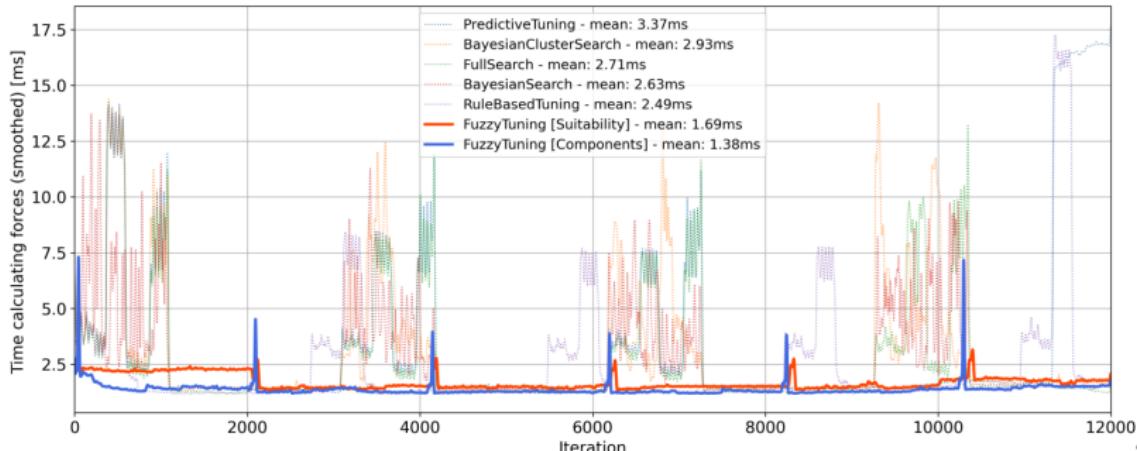
2 Implementation

3 Benchmarks

4 Conclusion

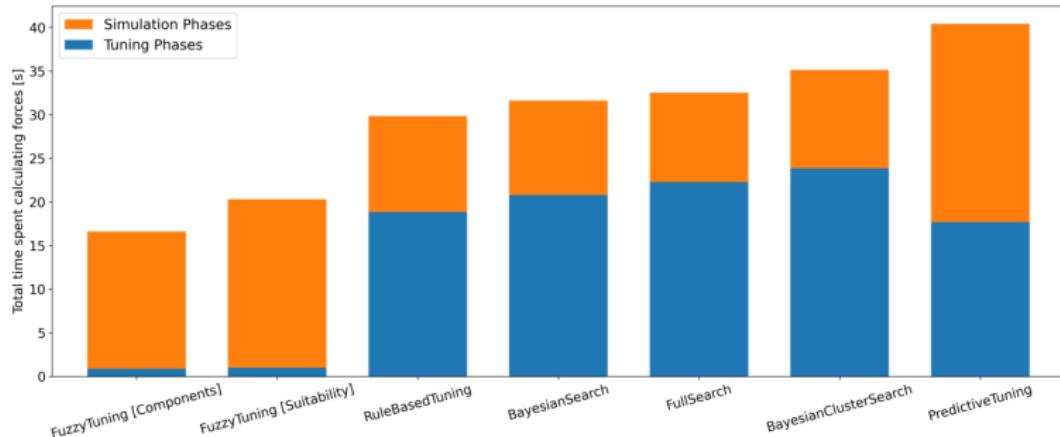
## Benchmark 1: Exploding Liquid

- Fuzzy Tuning Approaches:
  - Very short tuning phases
  - Selected configurations perform well
  - Winning configurations are (mostly) equivalent



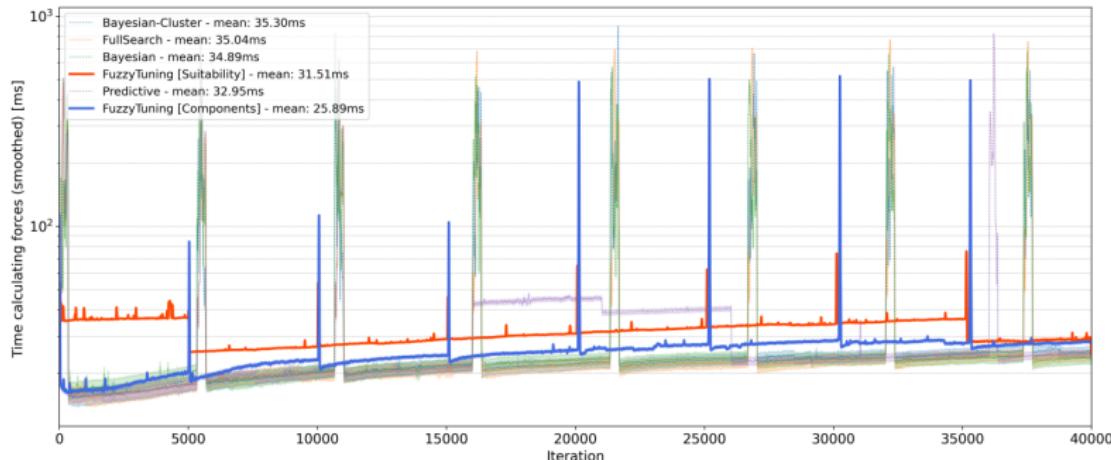
## Total Time: Exploding Liquid

- Fuzzy Tuning is good because:
  - Few evaluated configurations
  - Good evaluated configurations
- Massive reduction in time spent tuning  
→ More frequent tuning phases



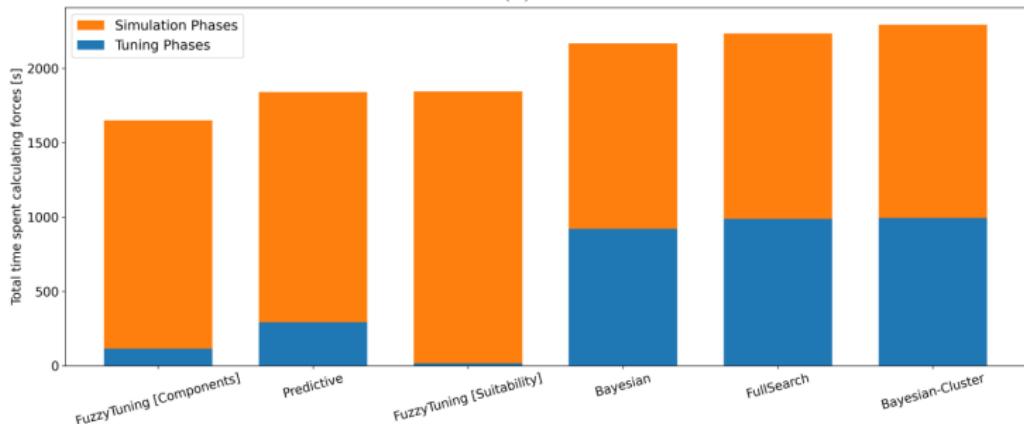
## Benchmark 2: Spinodal Decomposition (MPI)

- Component approach performs well
- Suitability approach misses the optimal configuration
  - However: no tuning overhead
  - Can this make up for the suboptimal configurations?



## Comparison and Evaluation: Spinodal Decomposition

- Component tuning wins again
- Suitability tuning:
  - Fast tuning phases compensate for suboptimal configurations!
  - Improvement: Encourage longer tuning phases
    - Increase suitability threshold (Top 10% → Top 30%)



## Table of Contents

1 Introduction

2 Implementation

3 Benchmarks

4 Conclusion

## Conclusion

- Fuzzy Tuning is very promising for AutoPas
  - Massive reduction in tuning overhead
  - Reduction of total simulation time (up to factor 1.96x)
  - Among the best tuning strategies (but very complex)
- Ongoing Challenge:
  - Rule generation
    - ✗ Requires a lot of data (1.1GB)
    - ✗ Unappealing for users
    - ✗ No universal solution (showed some generalization)

## Conclusion

- Fuzzy Tuning is very promising for AutoPas
  - Massive reduction in tuning overhead
  - Reduction of total simulation time (up to factor 1.96x)
  - Among the best tuning strategies (but very complex)
- Ongoing Challenge:
  - Rule generation
    - ✗ Requires a lot of data (1.1GB)
    - ✗ Unappealing for users
    - ✗ No universal solution (showed some generalization)

## Improvement Ideas

- Dynamic Rule Generation
  - Update expert knowledge on the fly
  - Automatically adapt to new scenarios
- Improvements to the Tuning Process
  - Investigate *early stopping* mechanism
  - Dismiss slow configurations early
  - Solve tuning overhead once and for all?
- Simplification of the Model
  - Ensemble of Decision Trees instead of Fuzzy Systems
  - Maybe comparable results?

**Thank you for your attention!**

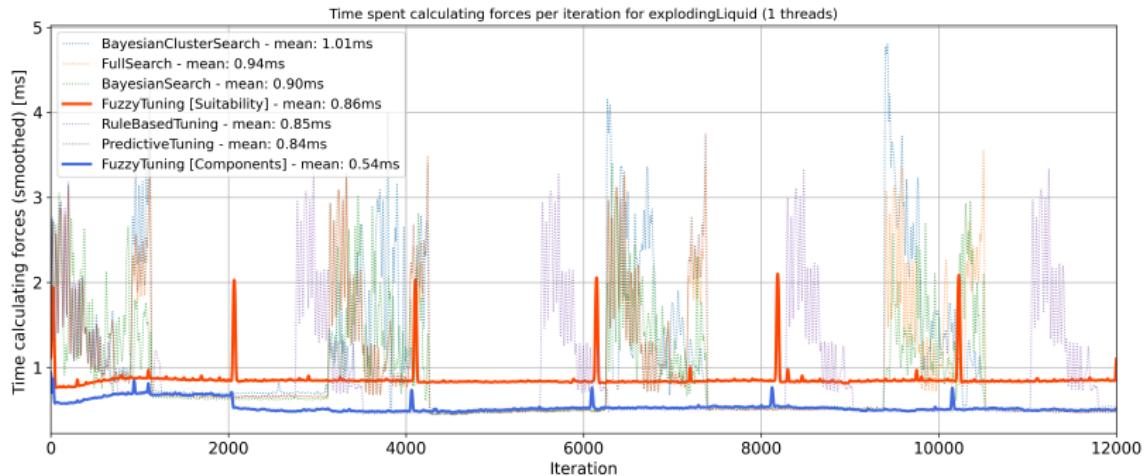
**Questions?**

## References I

-  Crockett, K., Bandar, Z., Mclean, D., and OShea, J. (2006).  
On constructing a fuzzy inference framework using crisp decision trees.  
*Fuzzy Sets and Systems*, 157(21):2809–2832.
-  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.

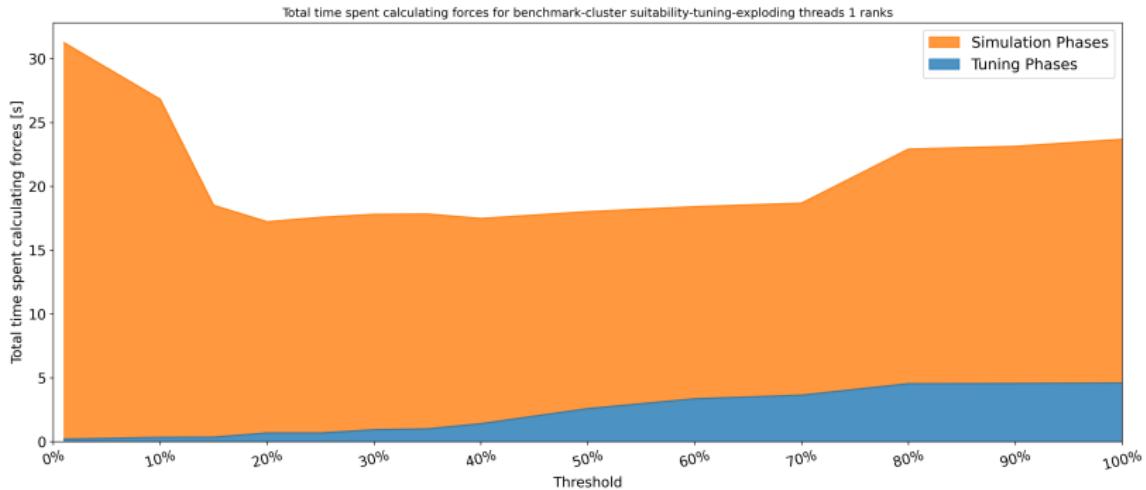
## Backup: What happens on new scenarios?

- Exploding Liquid: bigger  $\varepsilon \rightarrow$  more spacing
- Component Tuning is winning again

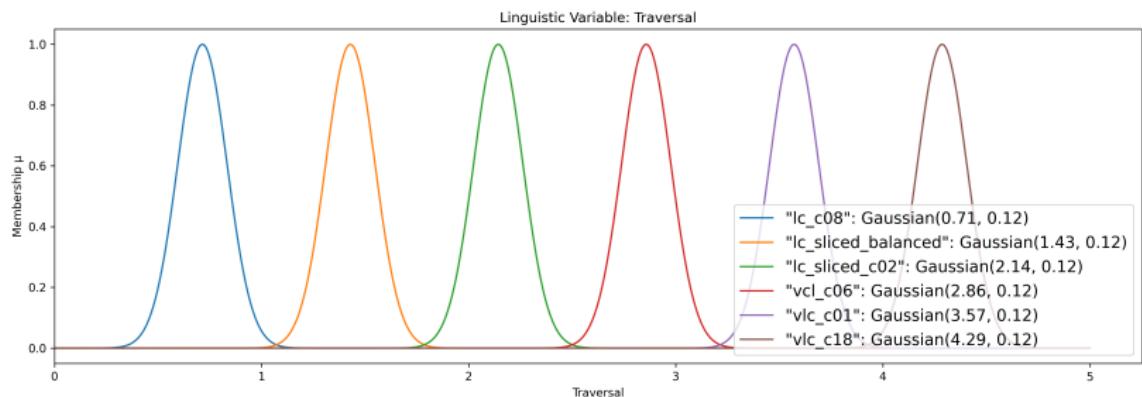


## Backup: Optimal Suitability Threshold

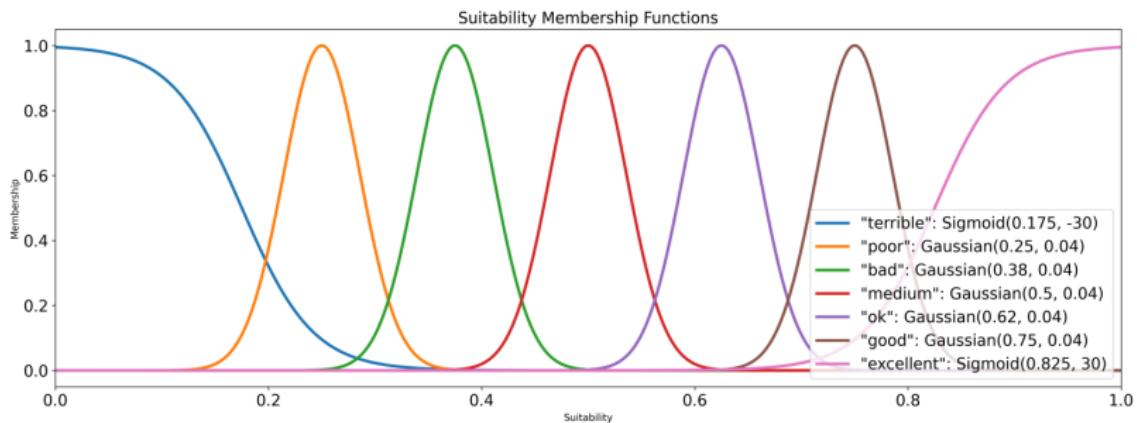
- Goal: find optimal hyperparameters for suitability-approach



## Backup: Linguistic Variables for Component Tuning

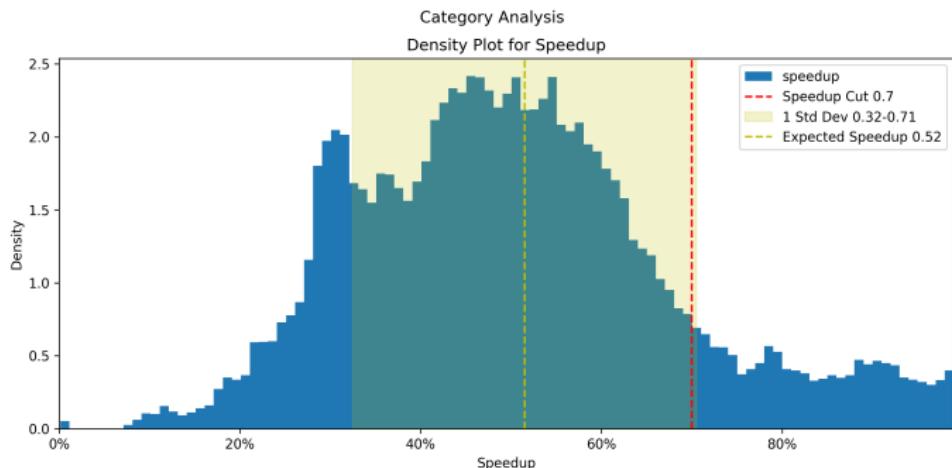


## Backup: Linguistic Variables for Suitability Tuning



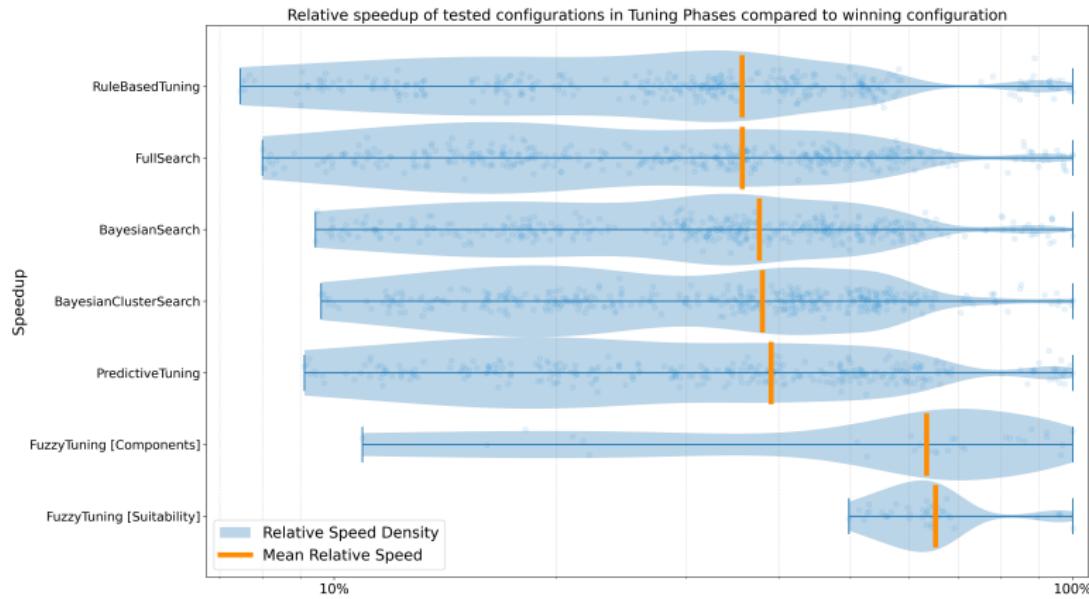
## Backup: Speedup Distribution

- How good is the average configuration?



## Backup: Quality of Predictions

- How well do tuning strategies suggest good configurations?



**Figure:** Source: MathWorks - Fuzzy Inference Process

