
Bachelor Thesis Final Presentation

Exploring Fuzzy Tuning Technique for Molecular Dynamics Simulations in AutoPas

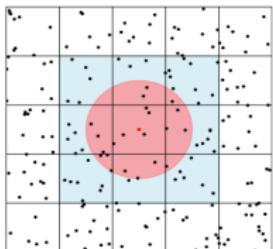
Manuel Lerchner
manuel.lerchner@tum.de

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

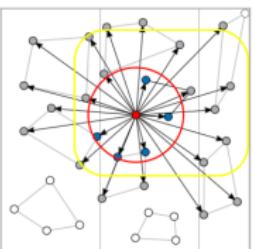
AutoPas

What is AutoPas?

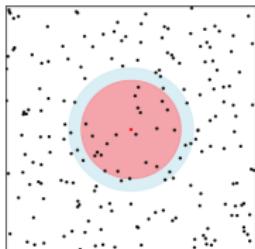
- Library for arbitrary N-body simulations
- Optimal performance by switching implementations
 - **Container:** Finding neighboring particles
 - **Traversal:** Parallel force calculations
 - **Data Layout:** Memory access optimization
 - **Newton 3:** Force calculation optimization



Linked Cells



Verlet Cluster Lists



Verlet Lists

Simpler Memory Access

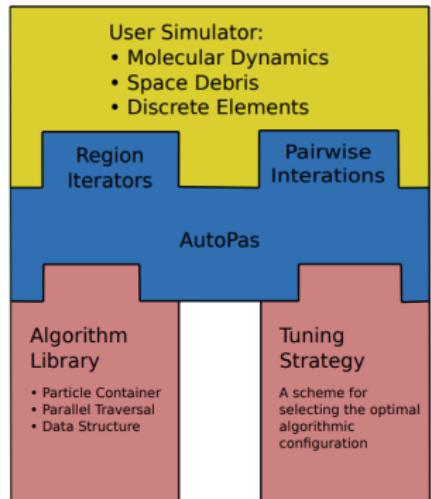
Lower Memory Overhead

Fewer redundant
calculations

[Newcome et al., 2024]

Structure of AutoPas

- Three main areas:
 - User Application
 - Algorithm Library
 - Tuning Strategies
- Algorithm Library:
 - Huge Search Space¹
- Tuning Strategies:
 - Full Search
 - Random Search
 - Predictive Tuning
 - Bayesian Search
 - Rule Based Tuning

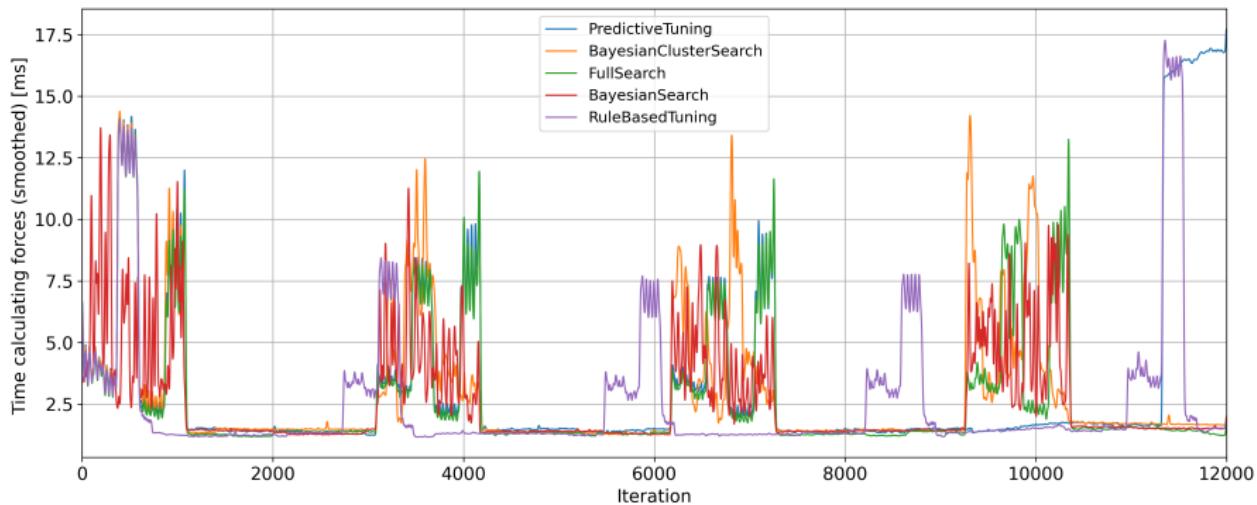


Source: [Newcome et al., 2023]

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

Auto-Tuning

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



Fuzzy Logic Systems

- Human-like reasoning to model systems $f : \mathbb{R}^n \rightarrow \mathbb{R}$
- (Fuzzy) Rule based systems
 - Interpolation effect between conflicting rules
 - No hard boundaries → robust against noise
- Smooth transition instead of hard boundaries
 - E.g. *cold, warm, hot* instead of $20^\circ C, 30^\circ C, 40^\circ C$

Example (Heater Control)

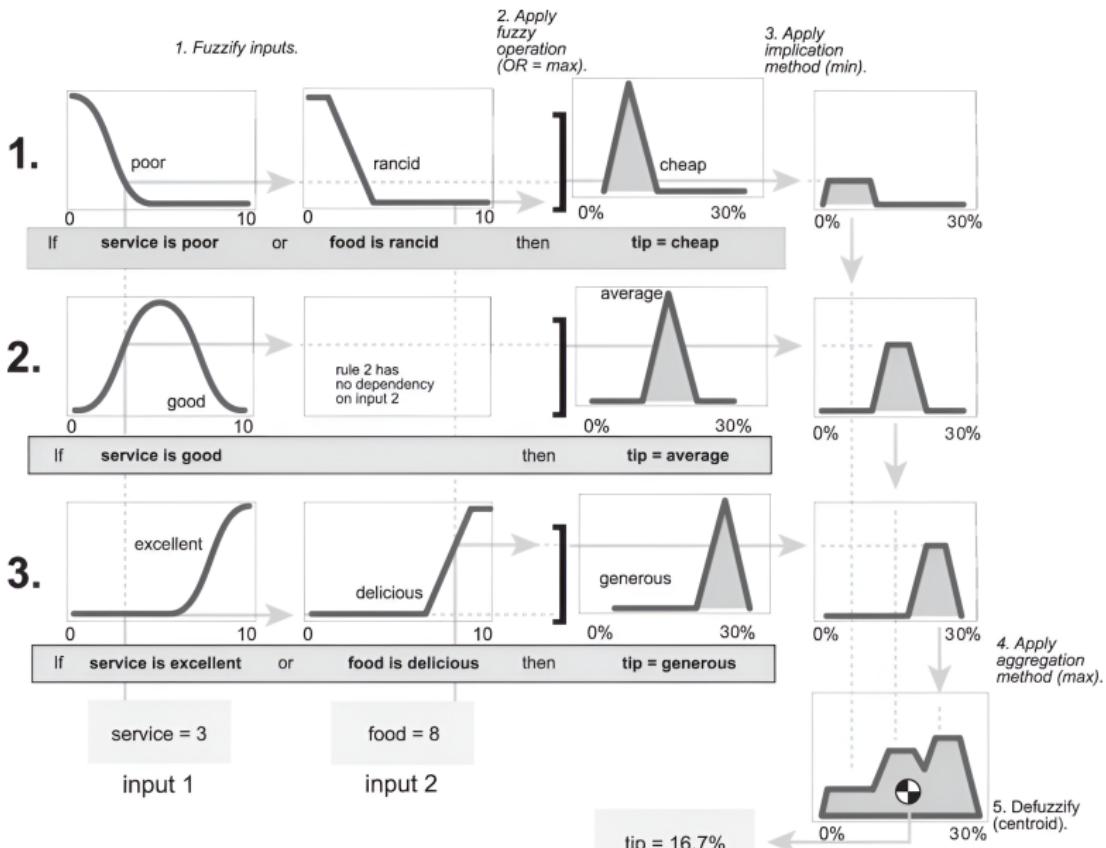
Input: temperature (e.g. $20^\circ C$), humidity (e.g. 60%)

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>

Source: MathWorks - Fuzzy Inference Process



Fuzzy Tuning for AutoPas

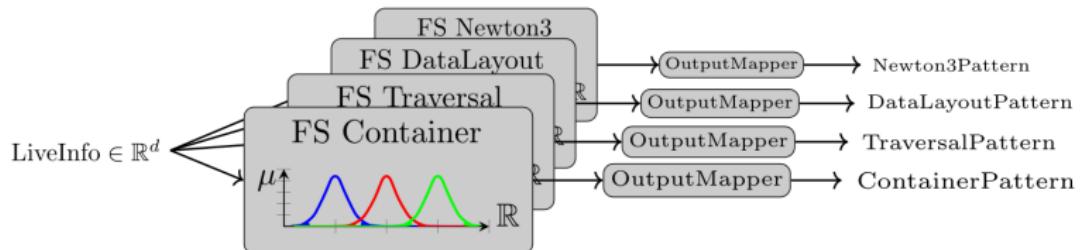
- Main Idea: Use Fuzzy Logic to tune AutoPas
- Expected Benefits:
 - Expert knowledge allows efficient tuning phases
 - ✓ Minimal tuning overhead
 - ✓ Allows frequent tuning phases
 - Comes with *interpolation* effect
 - ✓ Fewer rules than Rule-Based tuning
 - ✓ Generalization to new environments
 - Simplicity of the model
 - ✓ Easy to understand and interpret
 - ✓ Easy to maintain

Challenges and Questions

- Fuzzy Tuning not directly applicable to AutoPas
 - Tuning is a *discrete* optimization problem
 - Fuzzy tuning provides functions $f : \mathbb{R}^n \rightarrow \mathbb{R}$
 - Big Question: How to interpret the numerical output?
 - *Component Tuning* Approach
 - *Suitability Tuning* Approach
- 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

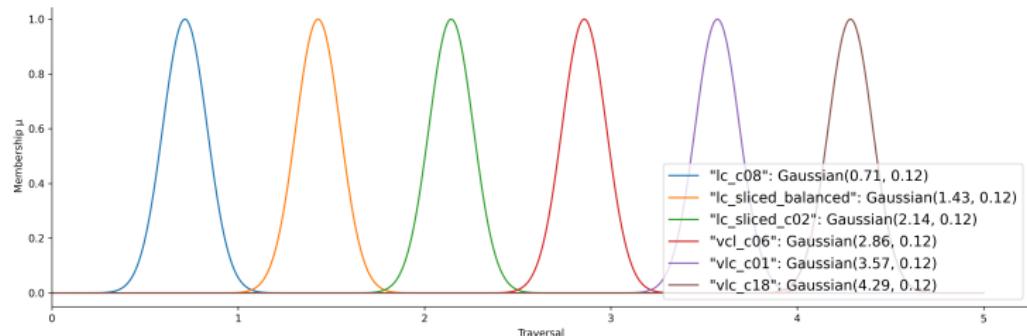
Approach 1: Component Tuning

- Predict good values for each tunable parameter separately
 - Explicit Fuzzy System for each tunable parameter
 - Output Variable: Continuous representation of the parameter
 - *Discrete* implementations are placed in the output space
 - Map numerical output to the *closest* value
- (Inspired by: [Mohammed et al., 2022])
- Combine individual predictions to form final configuration(s)



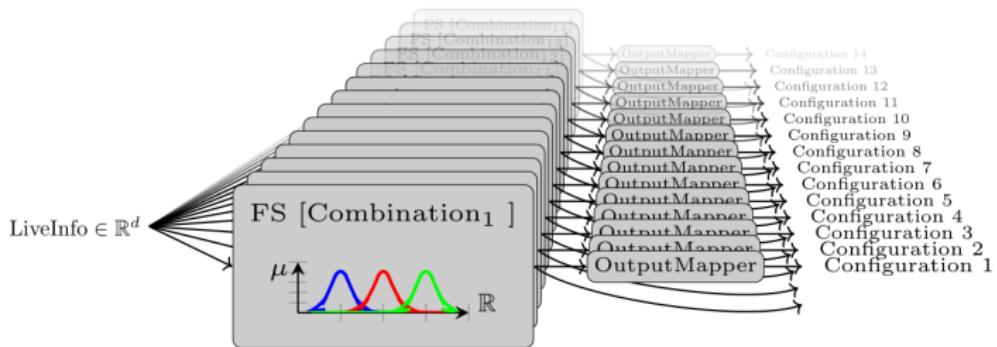
Approach 1: Component Tuning (Example)

- Style: **IF** avgParticlesPerCell is *low* **AND** threadCount is *low*
THEN traversal is *vcl_c06*
- **Benefits:**
 - Few fuzzy systems
 - Few and natural rules
- **Drawbacks:**
 - Limited interpolation effect (MoM defuzzification)
 - Independence assumption



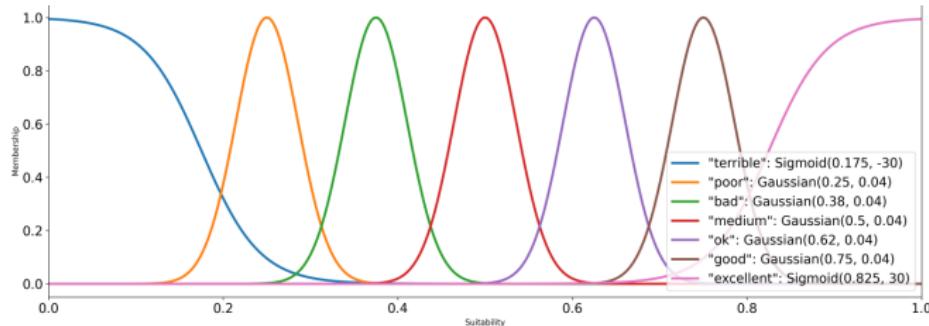
Approach 2: Suitability Tuning

- Predict the *suitability* of **each** configuration separately
- Explicit Fuzzy System for all possible configurations
 - Output Variable: Numerical suitability of the configuration
- Results in (*suitability, configuration*) pairs
 - Best configurations are selected



Approach 2: Suitability Tuning (Example)

- Style: IF `threadCount` is *high* AND `avgParticlesPerCell` is *low*
THEN `suitability_LinkedCells_AoS_lc_c18_disabled` is *bad*
- Benefits:
 - Utilizes the full power of fuzzy logic (CoG defuzzification)
 - Dependencies and incompatibilities can be modeled
- Drawbacks:
 - Huge number of fuzzy systems
 - Impossible to maintain by hand

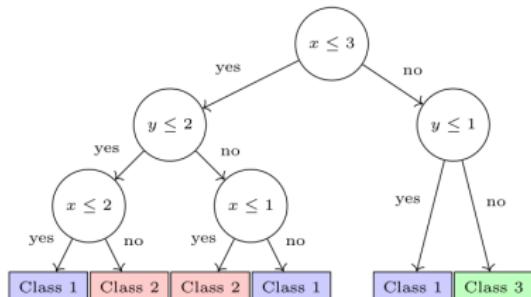


Data-Driven Rule Extraction

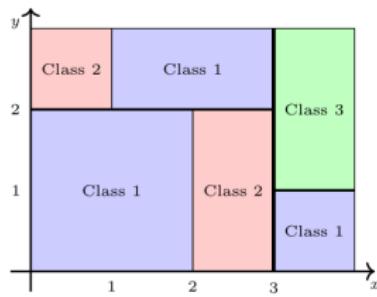
- Creating the Fuzzy Rules and Linguistic Terms manually is difficult
 - Expert Knowledge is required
 - Formalization of knowledge is difficult
 - Potentially many rules required
- Use Machine Learning to extract rules from data
 - Decision Tree → Fuzzy Decision Tree → Fuzzy Rules
[Crockett et al., 2006]
 - Does not require expert knowledge
- Human experts can still validate the rules

Decision Trees

- **Idea:** Split data with axis-aligned splits to best separate classes
- Corresponds to nested *if-then-else* rules
- Easy to understand and interpret
- TFinal tree contains the entire expert knowledge
- Can be trained automatically, given enough data
 - No expert knowledge required!



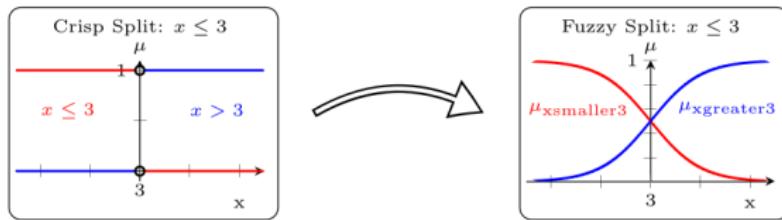
(a) Example decision tree

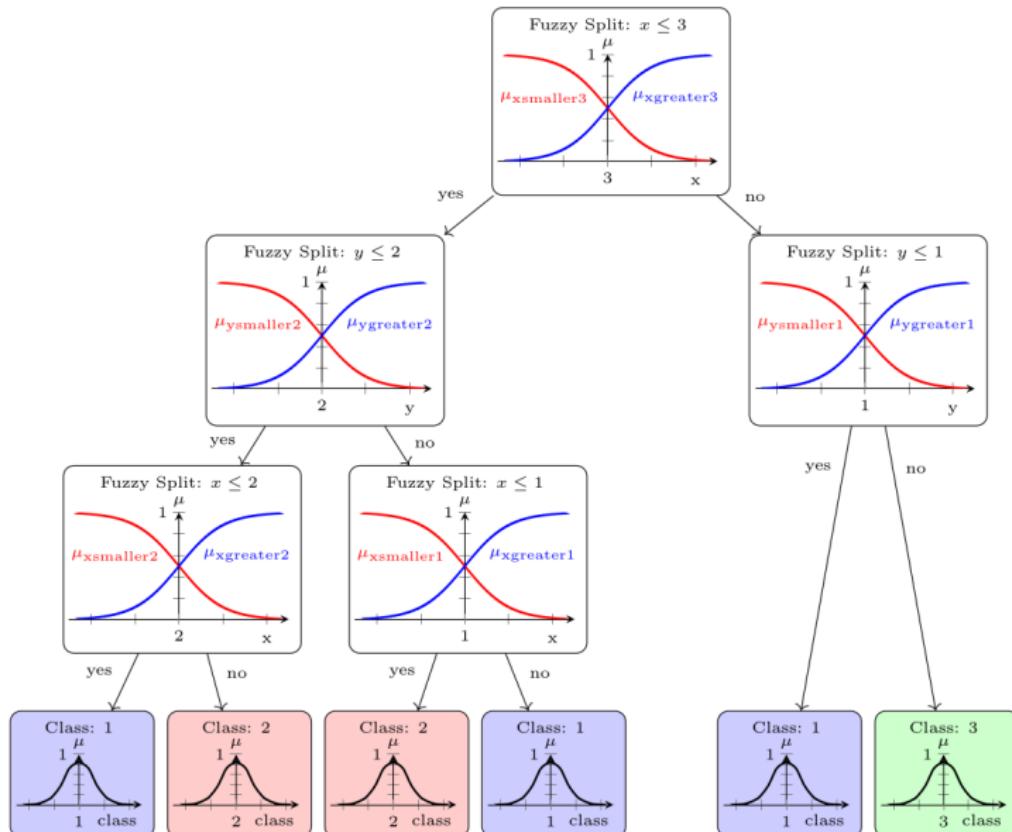


(b) Decision surface over $\mathcal{D} = [0, 4] \times [0, 3]$ 13 / 28

Decision Trees → Fuzzy Decision Trees

- Fuzzy Decision Trees
 - Each decision is a linguistic term
 - E.g. Traverse left if temperature is *cold*
- Conversion: Each (crisp) split is turned into two fuzzy sets
 - Fuzzy sets should maintain the semantics of the split
 - Provides robustness against noise
- Leaf nodes are represented with Linguistic terms





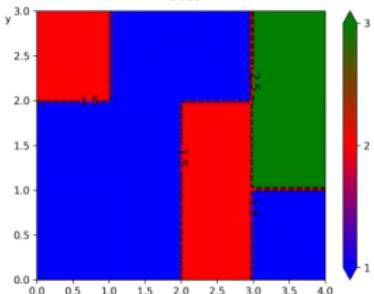
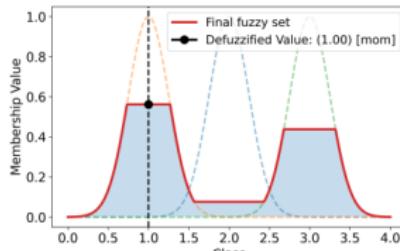
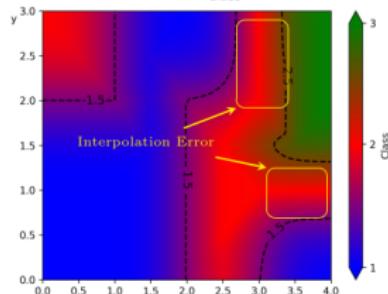
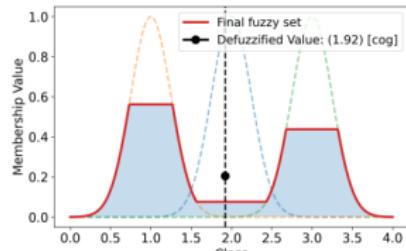
Fuzzy Decision Trees → Fuzzy Rules

- Depth-First traversal of the Fuzzy Decision Tree
 - Each node corresponds to a condition of the antecedent
 - Leaf node corresponds to the consequent
- One rule per path from root to leaf
- Corresponds to *unnesting* the decision tree

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

Fuzzy Decision Surfaces - CoG vs. MoM

- CoG: Interpolation effect + errors, smooth boundaries
- MoM: Hard boundaries, similar to Decision Trees



Fuzzy Rule Extraction for `md_flexible`

- Collect huge dataset of `md_flexible` simulations
- **LiveInfoData:** `maxDensity`, `homogeneity`, `threadCount`, ...
- **TuningData:** `Container`, `Traversal`, `Newton3`, ..., `Time`
- Introduce notion of *relative speed* for each configuration
 - $t_{best}^{(i)}$: Best configuration time in tuning phase i
 - $t_{config}^{(i)}$: Time of configuration in tuning phase i

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

- Allows for fair comparison between different tuning phases

Resulting Dataset

- Performance of configurations in different environments
- Can be used to extract rules for both approaches
- Goal: Find configurations with high relative speed

ParticlesPerCell			Miscellaneous			Configuration			Relative Speed
avg	max	stddev	homogeneity	max-density	threads	Container DataLayout	Traversal	Newton3	
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
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Generate Rules: Approach 1

- Naively remove bad configurations (relative speed < 70%)
- Group remaining dataset by tuning-phase (same live-info)
- Aggregate present (*good*) values for each parameter
- Apply Rule Extraction algorithm to 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	"vlc.c06, vlc.c18, vlc_sliced_c02"
:	:	:	:	:

Generate Rules: Approach 2

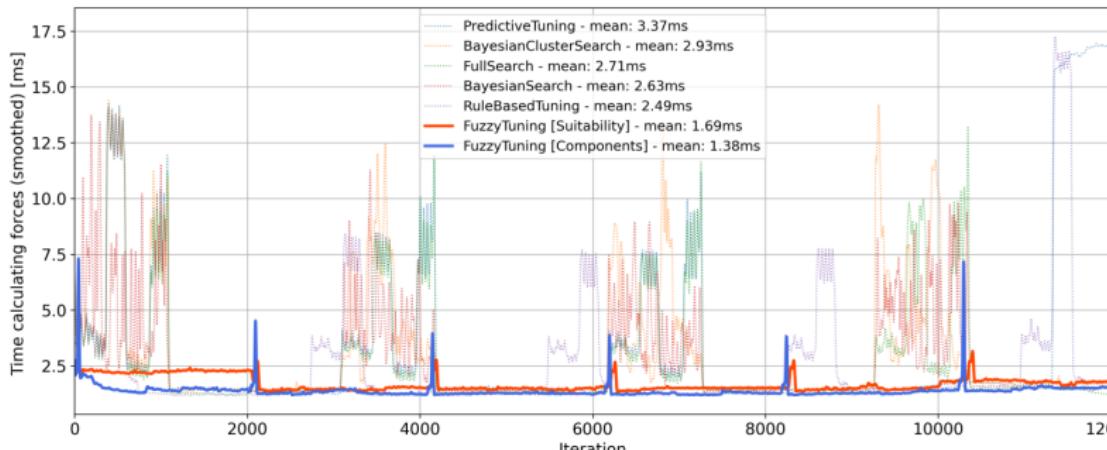
- Assign suitability-classes to each entry in the dataset
- Classes represent ranges of relative speed
- Apply Rule Extraction algorithm for the suitability class

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
⋮	⋮	⋮	⋮	⋮

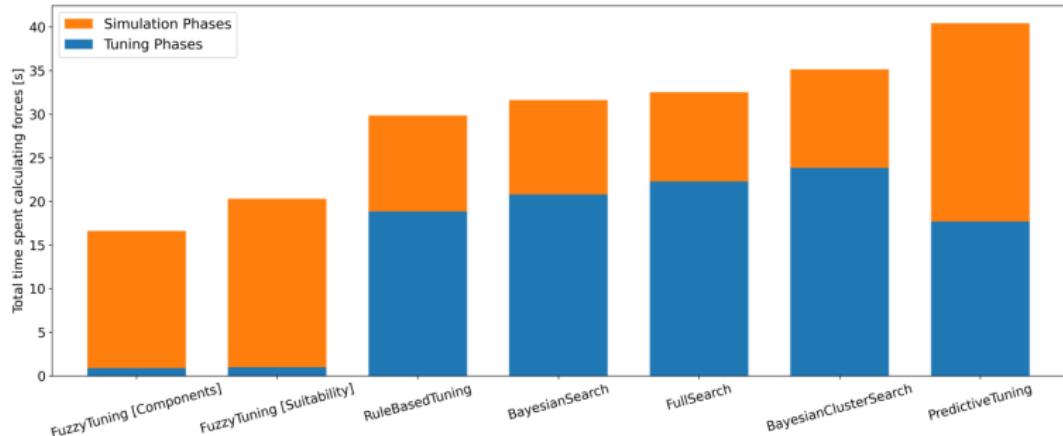
Benchmark 1: Exploding Liquid

- Exploding Liquid Benchmark (Included in Dataset)
 - Both fuzzy approaches look promising
 - Very short tuning phases (not much noise)
 - Selected configurations perform well (tiny spikes)
 - Winning configurations are (mostly) equivalent



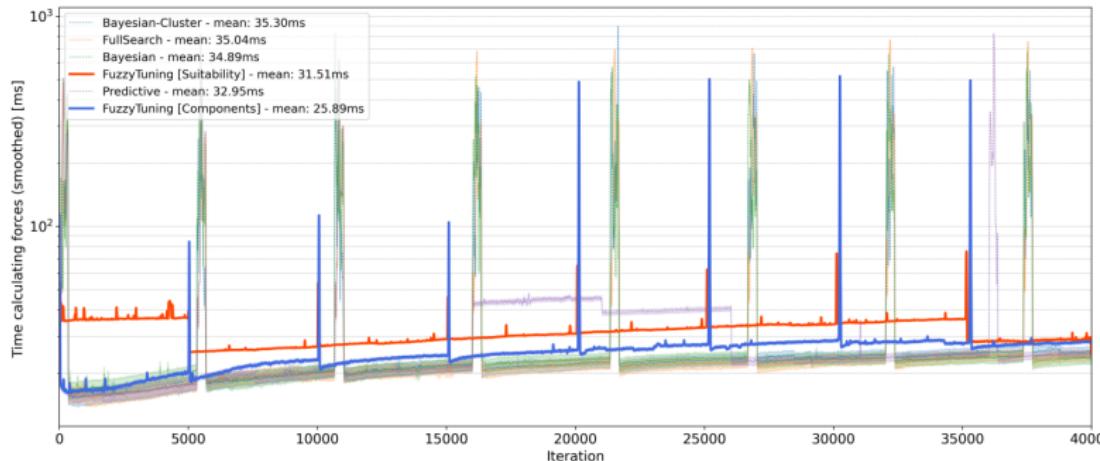
Total Time: Exploding Liquid

- Fuzzy tuning has lowest total time
- Mostly due to the very efficient tuning phases
 - Few configurations evaluated
 - Evaluated configurations are expected to perform well
- Benefits of tuning with tiny overhead



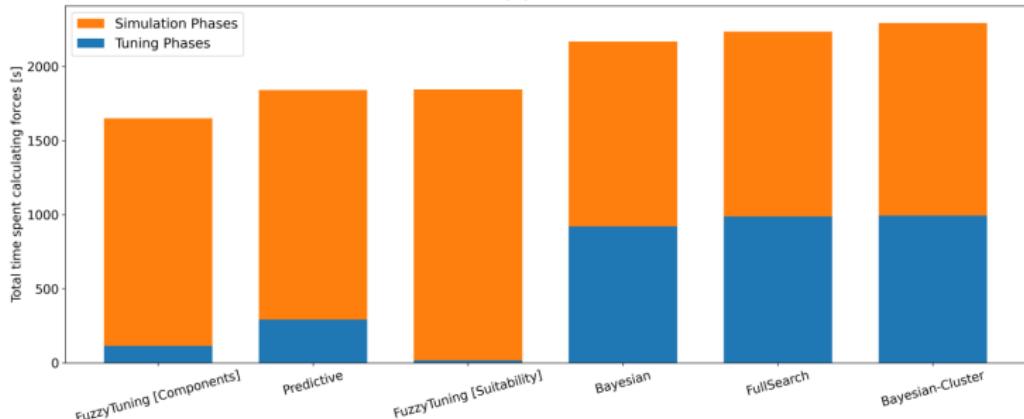
Benchmark 2: Spinodal Decomposition MPI

- Spinodal Decomposition MPI (Indirectly included in Dataset)
 - Fuzzy tuning promising again
 - Suitability approach struggles, often misses the best configuration
 - However, both approaches have efficient tuning phases



Comparison and Evaluation: Spinodal Decomposition

- Component approach performs well
- Suitability approach still quite fast
 - Fast tuning phases compensate for suboptimal configurations!
 - Misses caused by too optimistic suitability threshold
 - Too few configurations are selected for evaluation
 - **Solution:** Increase the suitability threshold (Top 10% → Top 30%)



Future Work

- Dynamic Rule Generation
 - Update the expert knowledge on the fly
 - Adapt to new scenarios
 - No need for giant datasets
- Improving Tuning Strategies
 - Implement early stopping mechanism
 - Stop evaluating extremely bad configurations early
- Simplification of the Fuzzy System to Decision Trees
 - Use Decision Trees instead of Fuzzy Systems
 - Could potentially perform equally well, while being easier to understand

Conclusion

- Fuzzy Logic is very promising for tuning AutoPas
- Data-Driven Rule Extraction is a powerful tool
- However:
 - Requires a lot of prior data (current training data is 1.1GB)
 - Users cannot be expected to collect this data beforehand
 - A more user-friendly approach is needed for broader adoption
- Future work could focus on simplifying/streamlining the rule extraction process
- Alternatively: Investigate Early Stopping, to solve the tuning overhead once and for all

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.
-  Mohammed, A., Korndörfer, J. H. M., Eleliemy, A., and Ciorba, F. M. (2022).
Automated scheduling algorithm selection and chunk parameter calculation in openmp.
IEEE Transactions on Parallel and Distributed Systems, 33(12):4383–4394.
-  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.

References II

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