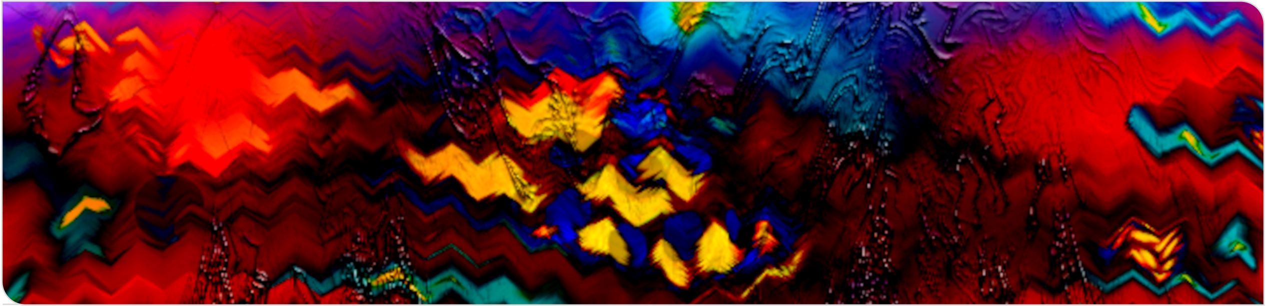
Karlsruhe Institute of Technology

Empirical Evaluations for Hard Algorithmic Challenges

Algorithm Engineering

Markus Iser | March 7, 2025



Professional Background

GitHub

Scholar

- PostDoc at the Algorithm Engineering Group of Peter Sanders
- Organizer of International SAT Competitions
- Research Focus:

  Boolean Satisfiability and Optimization:
  Algorithms, Data Structures, Heuristics, Implementations, Applications, Encodings

  Empirical Algorithmics:
  Global Benchmark Database, Informed Benchmarking, Explainable Algorithm Selection, Research Automation

- Focus of this Talk:

  Empirical Evaluations for Hard Algorithmic Challenges

Some Algorithmic Challenges

Efficiency: Asymptotic runtime and memory usage.

Scalability and Communication: Parallelizability and overheads in parallel and distributed computing.

Correctness: Verification and correctness guarantees.

Empirical Evaluations:

Analytical methods are insufficient to predict the performance of algorithms in practice.

Empirical evaluations are essential for understanding algorithm performance.

Runtime experiments form the cornerstone of **measurement** in empirical algorithmics.

Hard Problems: Boolean Satisfiability and Optimization

Karlsruhe Institute of Technology

Problems are given as a set of (linear) constraints of the form $\sum_{i=1}^{n} c_i x_i \circ b$.

$\rightarrow$ Boolean variables x_i, integer coefficients c_i, bound b, and operator $\circ \in \{\leq, \geq, =\}$

Goal: Find an assignment to the variables x_i that satisfies all constraints (or prove that none exists).

Problem Variants: Pseudo-Boolean Satisfiability (PBS), Propositional Satisfiability (SAT), Pseudo-Boolean Optimization (PBO), Maximum Satisfiability (MaxSAT), . . .

Applications: Verification, Placement/Routing, Product Configuration, Explainability, Planning, Scheduling, . . .

Complexity: NP-complete / NP-hard

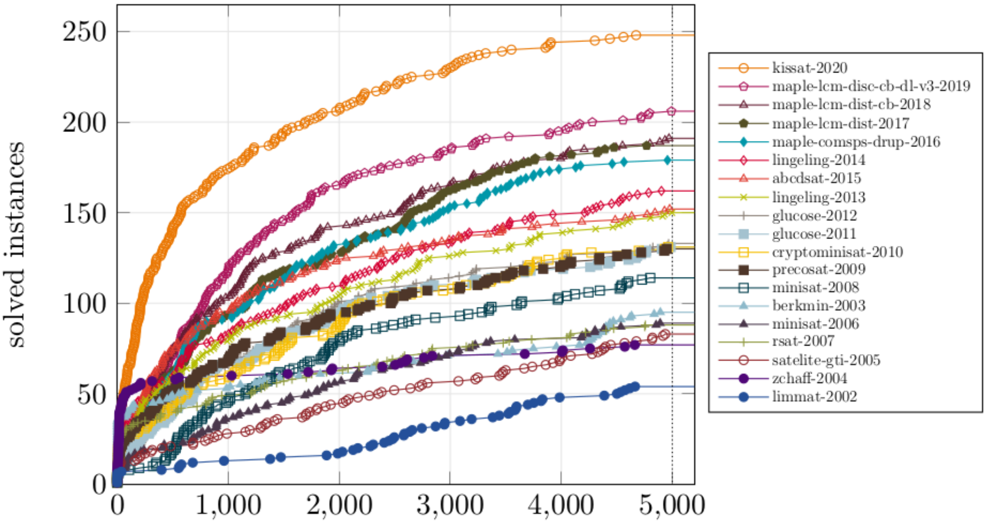
NP = Nondeterministic Polynomial Time: The class of problems where a given solution can be verified efficiently (in polynomial time), but finding a solution may not be feasible in polynomial time.

Boolean Satisfiability and Optimization
Ingredients of Efficient Solvers

Relevance has stimulated research to the extend that modern solvers can efficiently solve real-world instances.

- *Paradigms*: Local Search, Backtrack Search, Resolution-based Learning, ...
- *Algorithms and Data Structures*: Solvers are complex systems with many interacting components.
- *Heuristics and Configuration*: combinatorial explosion of choices
- *Implementations*: Bare-metal implementation skills are crucial, take the hardware architecture into account
- *Encoding*: devise efficient representation as set of constraints

SAT Competition Winners on the SC2020 Benchmark Suite



data produced by Armin Biere and Marijn Heule

Empirical Evaluations
Challenges in Benchmarking

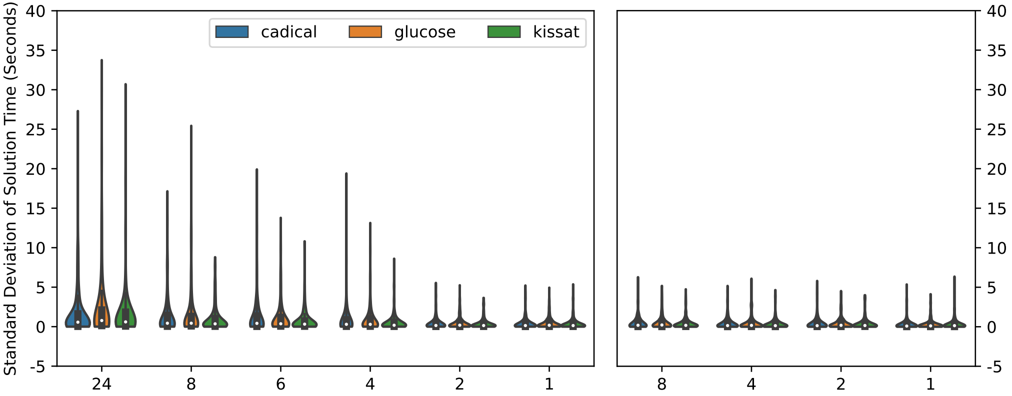
Karlsruhe Institute of Technology

- **Reproducibility**: Ensuring that results can be reproduced by others.
- **Consistency**: Ensuring that results are consistent across different runs.
- Evaluations in the past ran on a large amount of nodes (one job per CPU)
- Modern CPUs have a hierarchical memory structure
  - Non-uniform memory access (NUMA) in multicore architectures
  - Shared caches among cores on the same socket
  - Sharing of resoruces can lead to interference and unpredictable memory access patterns.

Empirical Evaluations

Reproducible and Reliable Measurements

Result of enforcing non-overlapping caches using Resource Control (resctrl) on Intel Xeon E5-2650v4.



From: Parallel Empirical Evaluations: Resilience despite Concurrency, Fichte et al., AAAI-24

Global Benchmark Database (GBD)

Turn to the Extension of an Algorithmic Problem

- Overcome working with sets of mostly anonymous benchmark instances
  $\rightarrow$ with 'meaningful' names like: `44-114583.cnf`
- Dataset interoperability and integrability through computable instance IDs
  $\rightarrow$ Computable ID serves as primary key for GBD datasets
  - Extensible, Decentralized Architecture
  - Identify identical benchmark instances
  - Join instance metadata from different sources
- Track domain, or author of instances, relationships between instances, isomorphism classes, etc.
- Tools for the seamless integration of benchmark data into existing workflows

Richer Evaluations of Runtime Data

Evaluation per Instance Domain (Example: SAT Competition 2023 Dataset)

Family	Count	BreakID-Kissat	SBVA-Cadical	Kissat-MAB-prop
interval-matching	20	10000.00	10000.00	**0.15**
or_randxor	5	103.93	**21.82**	10000.00
hashtable-safety	20	10000.00	797.46	**194.75**
satcoin	15	10000.00	**1395.53**	10000.00
set-covering	20	**262.39**	722.01	5761.57
cryptography-ascon	20	2673.77	**356.82**	5628.35
grs-fp-comm	17	8258.64	3649.90	**3435.82**
reg-n	5	10000.00	10000.00	**6295.16**
mutilated-chessboard	12	5135.77	3194.54	**1656.50**
production-planning	20	5031.37	**2470.42**	3946.63
hardware-verification	8	3964.51	2832.05	**1558.52**
register-allocation	20	2016.39	101.20	**5.50**
...	...	...	...	...

Goals

Systematize Algorithmic Ground Truth in NP-hard Problem Domains

- User-defined Domains, Feature Extractors, and Instance Transformers
- Relations between Domains, Relations between Encodings
- Track Instance Equivalence Classes, e.g., for Isomorphism Classes: isomorphism-invariant hashes
- Generic Toolset for Algorithmic Evaluation
- Rolling Competitions and Automated Benchmarking
- Data-driven Algorithmic Research