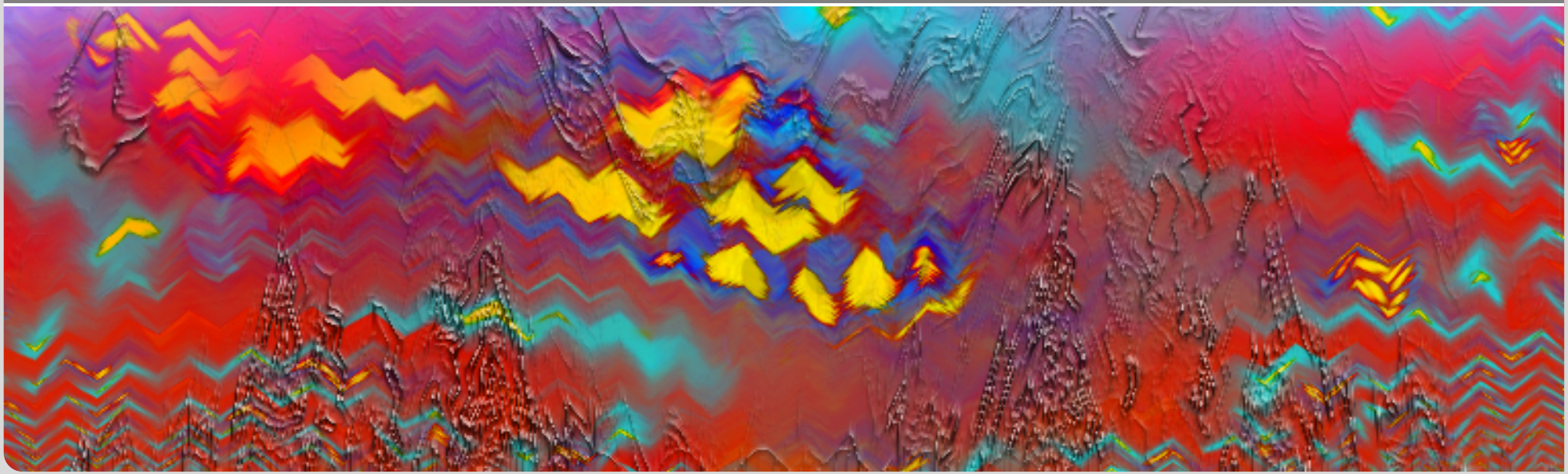
Karlsruhe Institute of Technology

Collective Management of Benchmark Metadata

Markus Iser, Carsten Sinz

KARLSRUHE INSTITUTE OF TECHNOLOGY (KIT)



Motivation

Dealing with large amounts of benchmark data

- Distributed over different groups and computers
- Names are not unique, duplicate benchmark problems
- Searching for problems with specific properties

How do we store properties such that they can be safely assigned to the problem?

Roadmap

- Classes of Benchmark Data
- Metadata Usage
- Benchmark Fingerprinting
- Reference Implementation – Global Benchmark Database (GBD)
- Presentation

Classes of Benchmark Data

Meta-data (non-calculable)

- Author
- Generator
- Encoding
- Application Domain
- Local Path
- Online Source
- Inclusion in Competition Set
- ...

Classes of Benchmark Data

Feature-data (calculable, easy)

- Number of Variables / Clauses
- Maximum clause-length
- Number of connected components
- Tree-width
- Problem class (Horn, 2-SAT, etc.)
- ...

Classes of Benchmark Data

Heavyweight Data (calculable, hard)

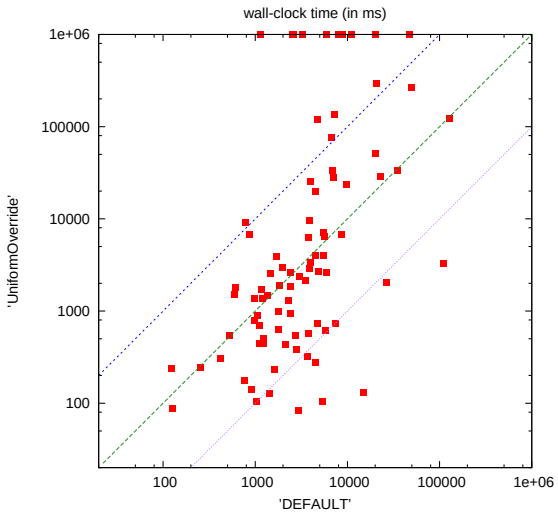
- Solution to SAT-Problem
- Number of Solutions
- Isomporphic Problems
- Size of shortest (recorded) Proof
- Runtimes
- Best (recorded) runtime
- ...

Use Case: Correlation Analysis

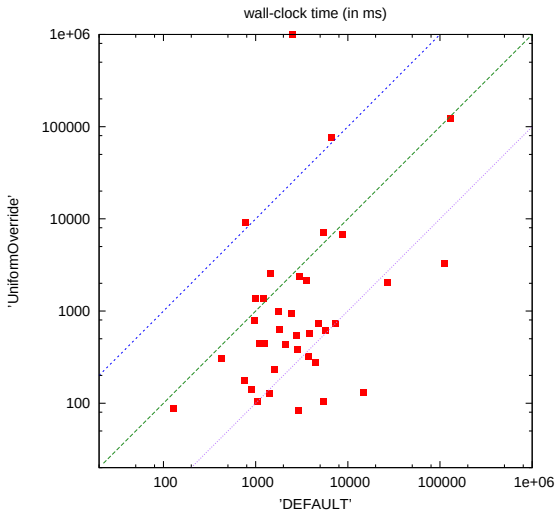
Karlsruhe Institute of Technology

Very common to analyze runtimes with respect to SAT result.

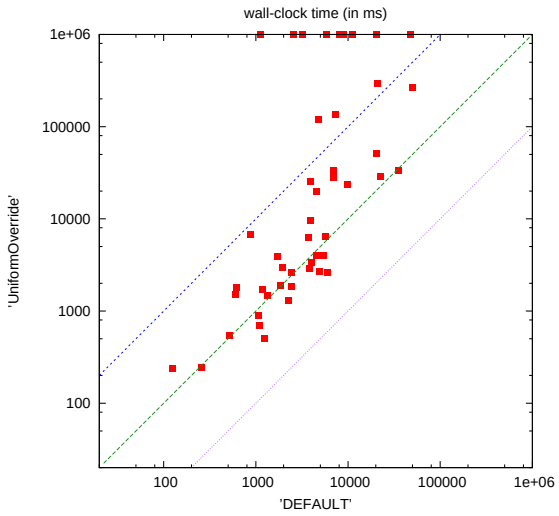
Mixed Result



SAT Result



UNSAT Result



Use Cases

Correlations

- Experimentation $\rightarrow$ Observation $\rightarrow$ Conclusion
- Come to conclusions with previously “inconclusive” results
- Possibility of correlation-based hypothesis

Elimination of Duplicate Benchmarks

- Clause and Literal Ordering
- Variable Renaming
- Create Equivalence Class $\rightarrow$ Choose Representant $\rightarrow$ Union-Find

Benchmark Fingerprinting

Problems with storing and sharing benchmark meta-data

- Filenames can change
- Existence of duplicate problems
- Problem file-size can be huge

Fingerprinting as Fundamental Requirement for Sharing

- Specification of Common Hash Function
- Benchmark Normalization (comments, whitespace, etc)

Benchmark Fingerprinting

Solution in our Reference Implementation (GBD)

- Removal of comments and additional whitespace
- Normalization of new-line characters
- md5sum (integration)

Benchmark Fingerprinting – Further Ideas

For Discussion: Increased Dedication, Loss of Integration

- Hash Dimacs without the header (header information can be incorrect)
- Improve Collision Avoidance by Appending e.g. “Number of Variables”
- Use other hash-function (sha-1)
- Develop DIMACS-specific hash-function . . .
- . . . with invariance guarantees (e.g. w.r.t. clause order)

Global Benchmark Database: GBD

Reference Implementation GBD

- based on Python and SQLite
- uses a two-column hash/value table for each attribute
- uses attribute types *integer*, *text*, *double*
- distinguishes “unique” and “non-unique” attributes
- possibilty to specify a (queryable) default-value for certain attributes
- pipe-based: query for hash value and pipe them to other gbd commands
- automatic bootstrapping: initializes a table containing hashes and paths to locally available problems (local path is just another attribute)

GBD – Future Work

- Continue to provide a reference implementation for the specification of a commonly used hash-function
- include a REST webservice, easily expose database in the web and run queries against public URLs
- Will be used to aggregate all the results of my thesis
- Automatic import of header comments (specification of format)

Tool Presentation

Karlsruhe Institute of Technology

GBD is maintained and available at

`https://github.com/Udopia/gbd`