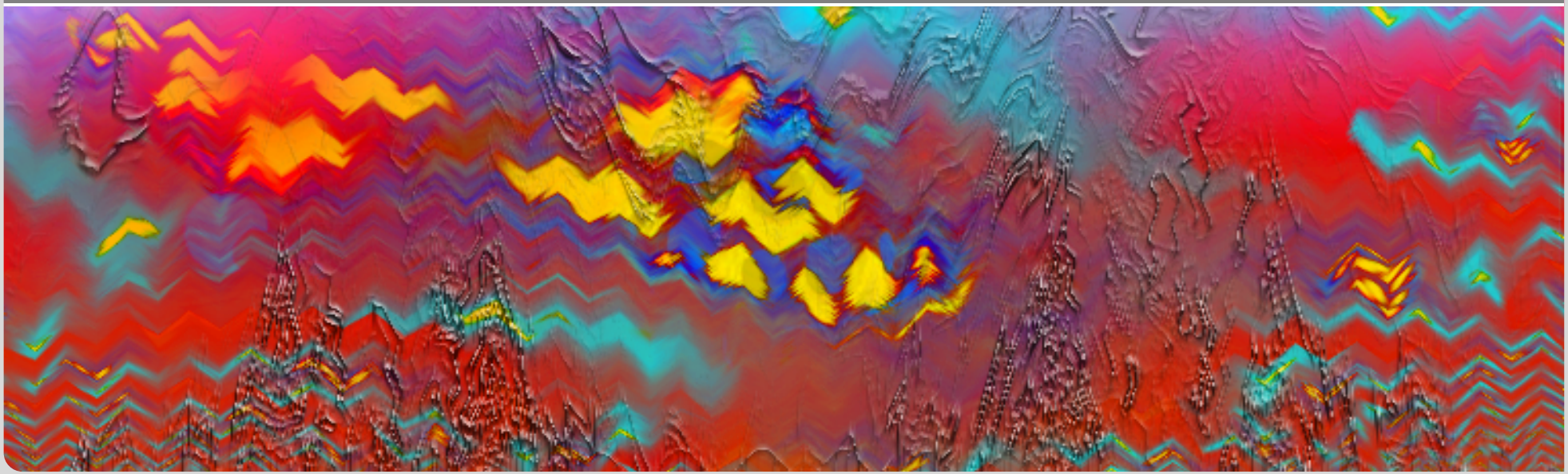
Karlsruhe Institute of Technology

CandyHorde: Candy Kingdom meets HordeSAT

Markus Iser

KARLSRUHE INSTITUTE OF TECHNOLOGY (KIT)



Candy

`https://github.com/Udopia/candy-kingdom`

Candy

- Fork of Glucose (MiniSAT)
- Modern C++
- Modularity: Separation of Concerns
- Simplicity: Use of Standard Library

New Sub-Systems

- Structure Analysis and Random Simulation (Gates, Tseitin)
- Structure-based Branching (RSIL)
- Structure-based Abstraction (RSAR)
- Structure-based Model Minimization

Candy Systems

- Clause Database (Clause, Clause Allocation)
- Current Assignment (Trail)
- Propagation System (Watchers)
- Clause Learning System (Conflict Analysis, 1-UIP Learning)
- Branching System (VSIDS, LRB, RSIL)

Candy Static Polymorphism

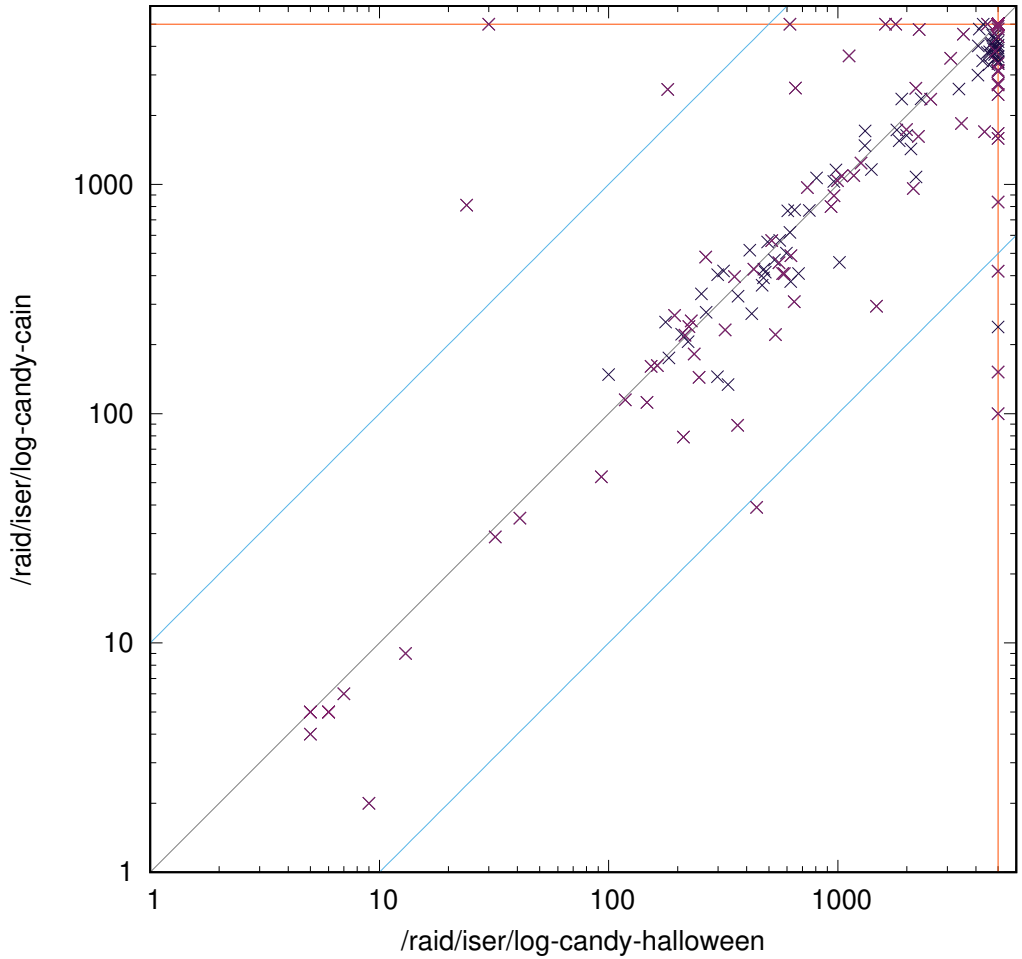
Candy Initialization

```
TDb*      clauses    = new TDb     ();
TTrail*   assignment = new TTrail  ();

TProp*    propagate  = new TProp   (db, assignment);
TLearn*   learning   = new TLearn  (db, assignment);
TBranch*  branching  = new TBranch(db, assignment);

Candy* solver =
  new Solver<TDb, TTrail, TProp, TLearn, TBranch> (
    clauses, assignment, propagate, learning, branching
  );
```

- TBranch: VSIDS, LRB, RSIL, ...
- TTrail and TPropagate now have “ThreadSafe” Variants

Recent Candy Development

“Halloween” Version vs Recent Version “Cain”

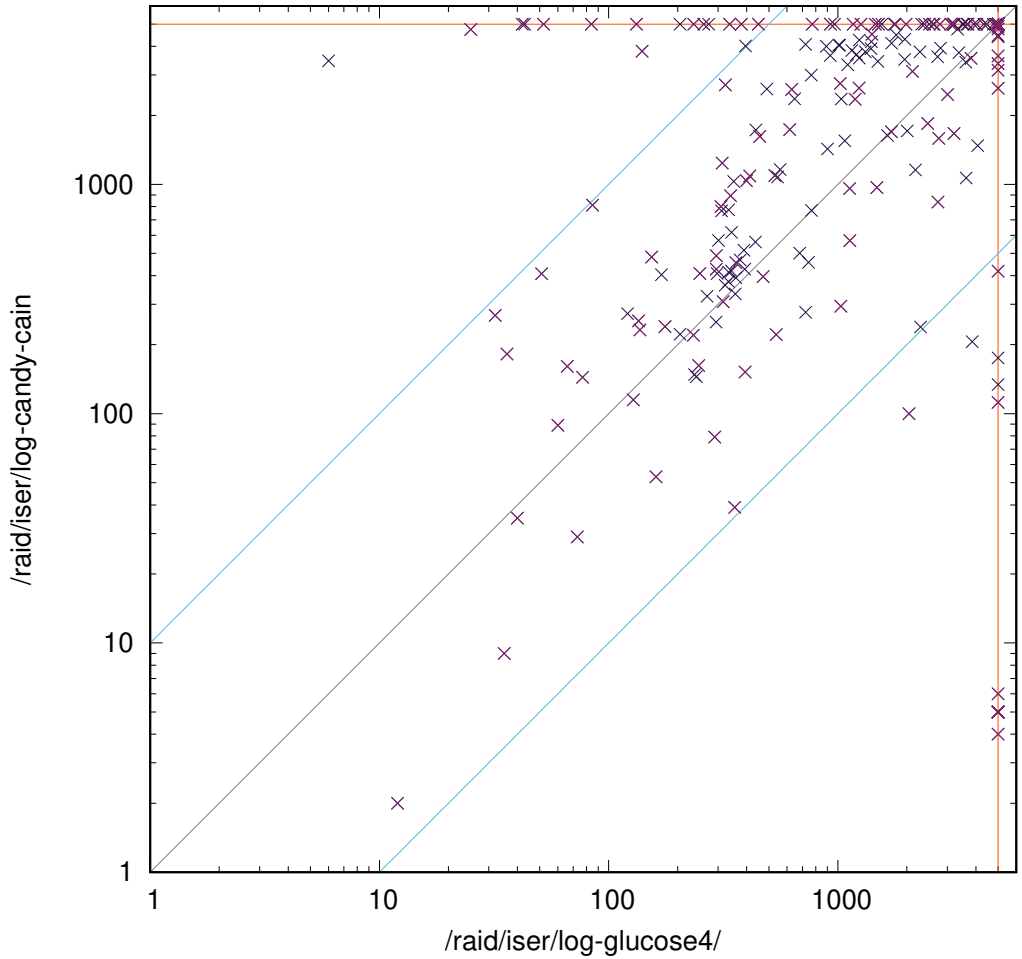


Halloween:
Solved: 129/400
PAR-2: 2909362

Cain:
Solved: 146/400
PAR-2: 2772256

Compared to “Mother” Glucose

Still Worse Performance



Glucose:
Solved: 171/400
PAR-2: 2502445

Cain:
Solved: 146/400
PAR-2: 2772256

HordeSAT

https://github.com/biotomas/hordesat

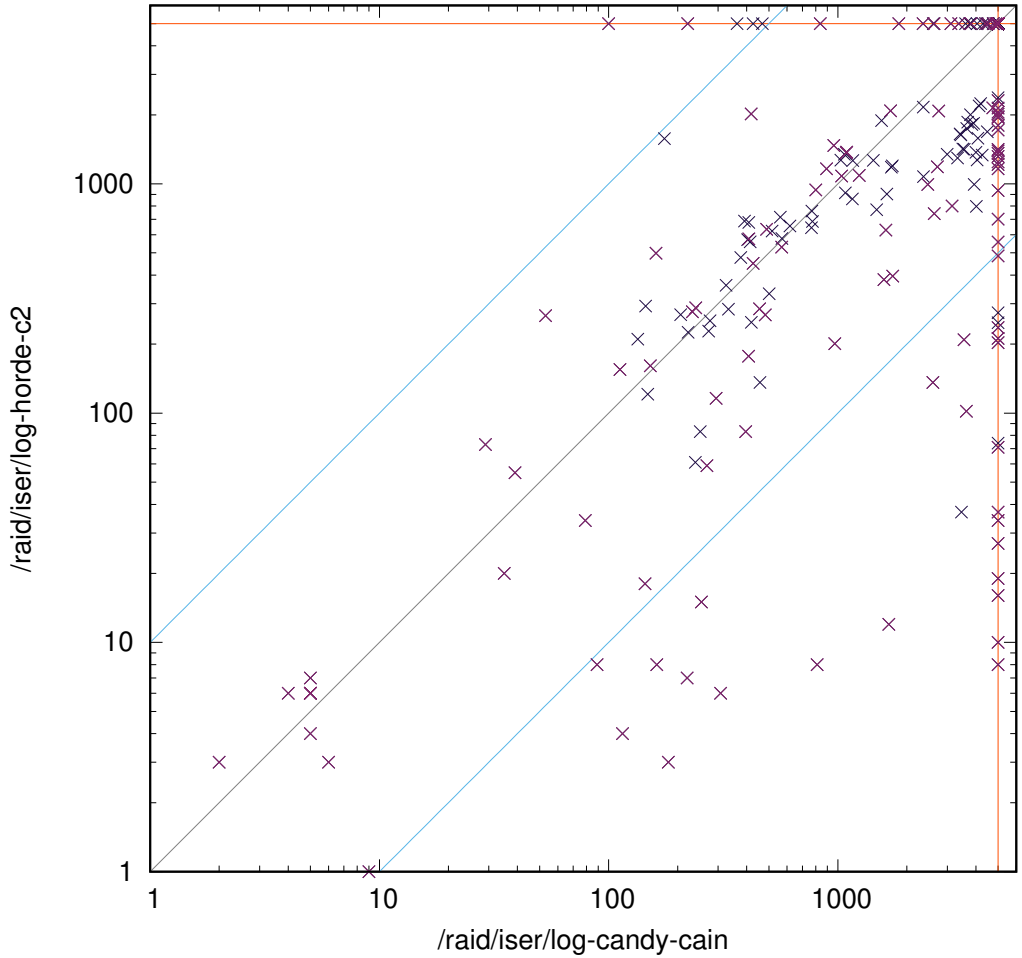
HordeSAT Parallel Interface

- Highly configurable parallel solver implementation
- Abstracts away details of single solver implemenation
- Can run portfolio of solvers
- Implementation of Horde interface: Lingeling, Minisat and now . . .
- . . . **Candy!**

Candy Horde

First Results

One vs. Two Instaces of Candy Version “Cain”



Single Cain (VSIDS):
Solved: 146/400
PAR-2: 2772256

Parallel Cain (VSIDS & LRB):
Solved: 156/400
PAR-2: 2559229

no clause-sharing or additional diversification so far

Solving the Memory Problem

Sharing of Clause Database

- Modularity of Candy makes it easy to share one Clause Database across solver instances
- However, propagate and clause-learning sub-systems utilize order of literals in clauses
- New implementations of propagate and clause-learning sub-systems must operate without changing or exploiting literal-order

Literal Order

First Two Literals in Clause are Watched

- The first literal is the the asserted (true) literal
- The second literal is the propagation trigger / the other watched literal
- Advantage: Watchers are independent (one watcher for each of the two literals)

Refactoring the Solver

Step-by-step

- Make all non-const methods in clause “private”
- Mark classes that access these methods as “friends”
- Reduce the number of friends to clauses (until only propagate and clause-analysis remain)
- Write new propagate and clause-analysis that are not friends of clause

Building a literal-order invariant Conflict Analysis

Advantage: Can be tested indepentent of Propagator

Step-by-step

- Make all non-const methods in clause “private”
- Mark classes that access these methods as “friends”
- Reduce the number of friends to clauses (until only propagate and clause-analysis remain)
- Write new propagate and clause-analysis that are not friends of clause

Conflict Analysis becomes Easier

Legacy Clause Analysis

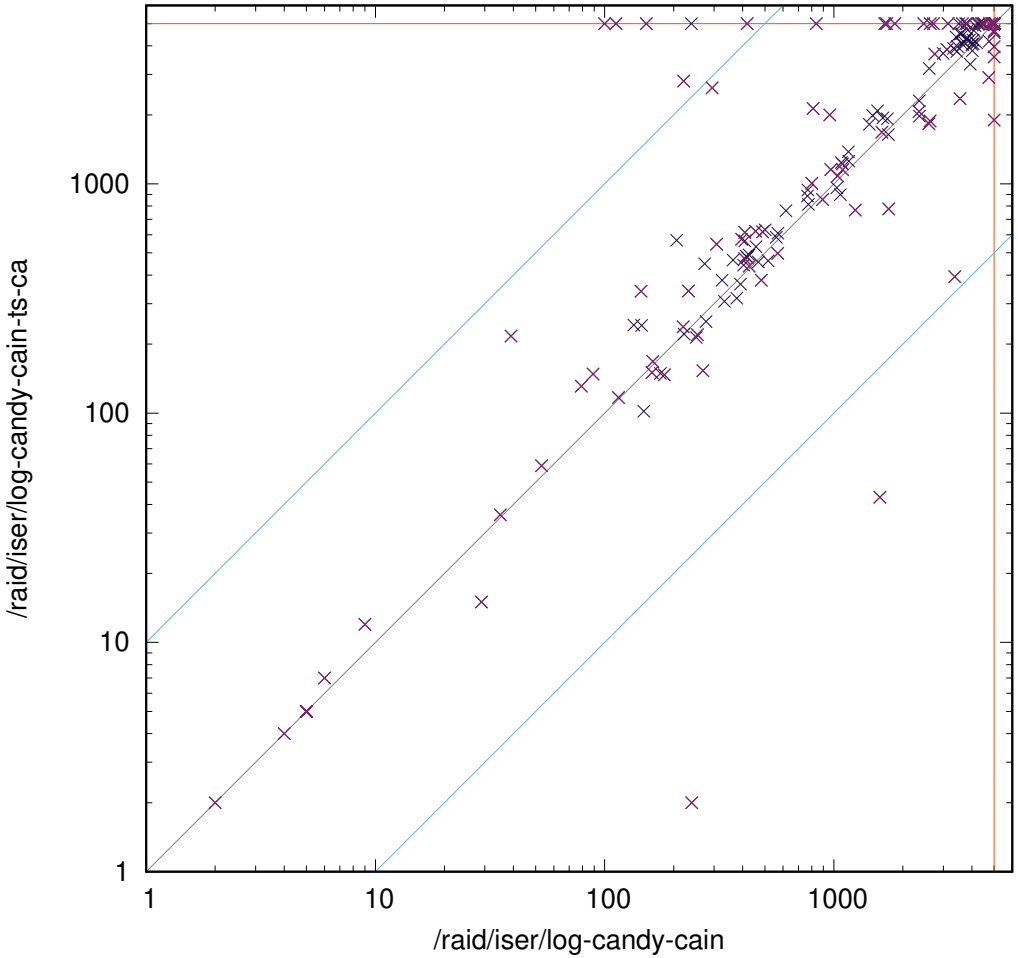
```
if (confl->size() == 2 && trail.value(confl[0]) == l_False) {
  confl->swap(0, 1);
}
auto it = isbegin() ? confl->begin() : confl->begin() + 1;
for (; it != confl->end(); it++) {
  Var v = var(*it);
  if (!stamp[v] && trail.level(v) != 0) { ... }
}
```

New Clause Analysis

```
for (Lit lit : *confl) {
  Var v = var(lit);
  if (lit != aslit && !stamp[v] && trail.level(v) != 0) { ... }
}
```

First Results

Candy Version “Cain” with Classic vs. New Conflict Analysis



Classical Cain:
Solved: 146/400
PAR-2: 2772256

New Cain:
Solved: 128/400
PAR-2: 2923940