Proof Logging with CaDiCaL

Florian Pollitt

universität freiburg

WHOOPS'25: 2nd International Workshop on Highlights in Organizing and Optimizing Proof-logging Systems

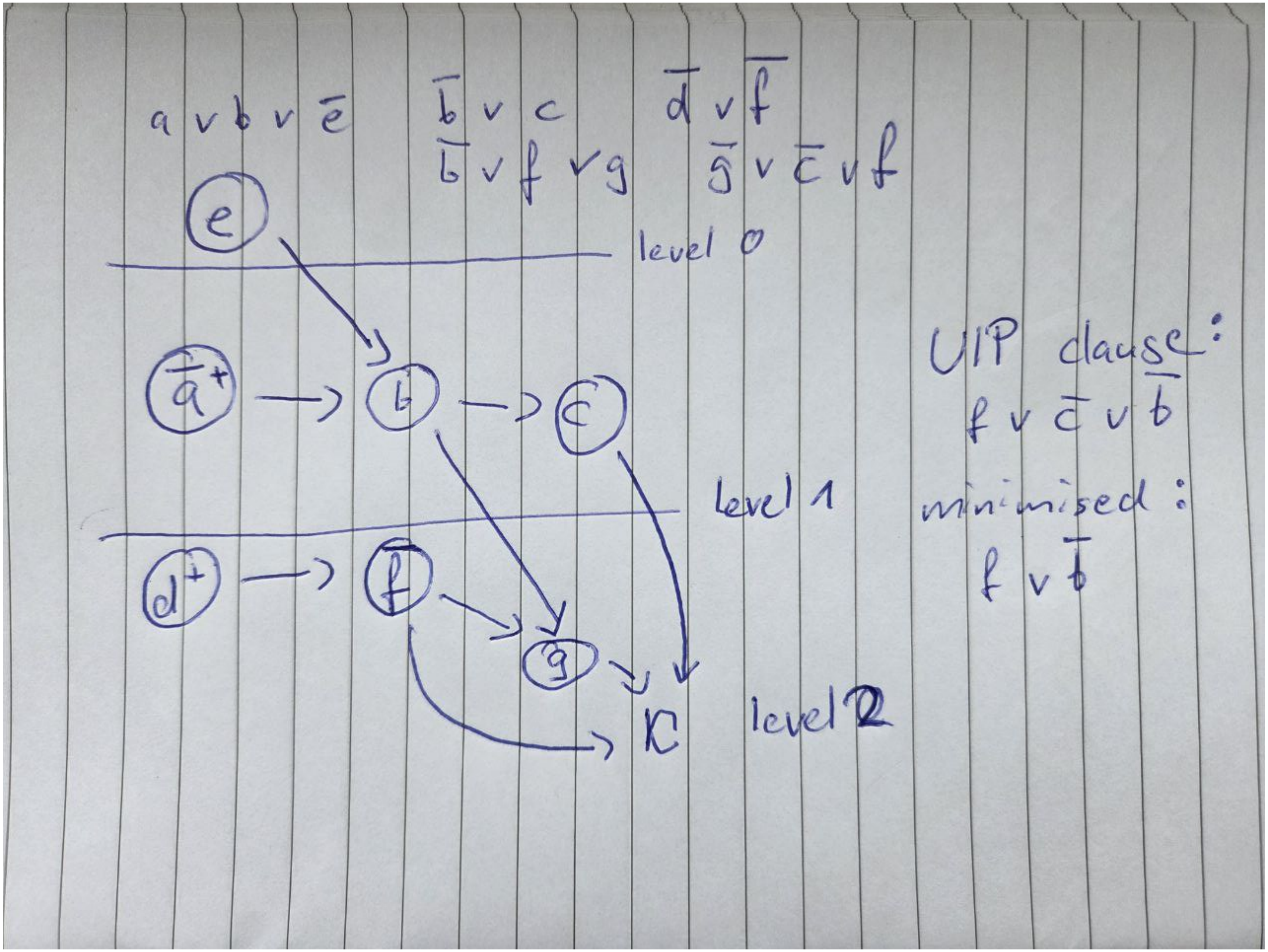
September 13.-14., 2025, Orsay, France

supported by DFG Intel

Proof logging for standalone SAT solvers

- DRAT vs. LRAT
- trade solving and engineering effort for checking efficiency.
- good trade in SAT solvers [SAT'23].
- necessary for PB [CP'25].

Implication Graph



Implementation — LRAT — practical challenges

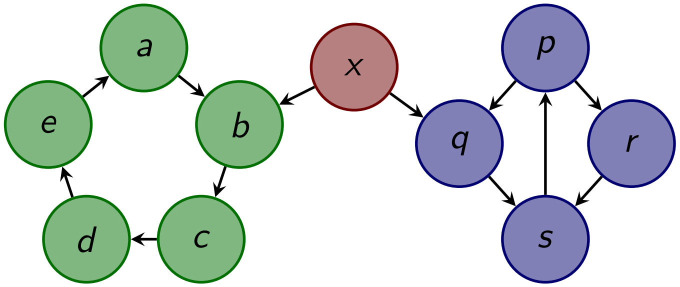
- RUP vs. resolution semantics.
- only RUP in cadical $\rightarrow$ resolution.
- many different algorithms.
- implicit resolution.
- track and reconstruct.
- on the fly computation.
- simplify if possible.
- design your algorithms differently?

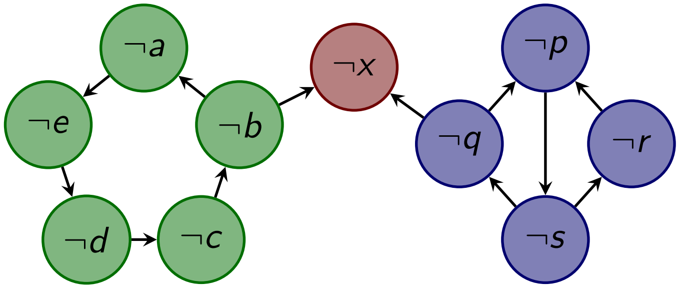
Example — BIG decomposition

$\neg a \wedge b$
$\neg b \wedge c$
$\neg c \wedge d$
$\neg d \wedge e$
$\neg e \wedge a$

$\neg x \wedge b$
$\neg x \wedge q$

$\neg p \wedge q$
$\neg p \wedge r$
$\neg q \wedge s$
$\neg r \wedge s$
$\neg s \wedge p$





Example — BIG decomposition

- substitution depends on representative.
- positive and negative cycles.
- lowest absolute value for consistency.
- recompute paths after cycles are fixed.
- binary resolution chains for both directions.
- not easily combinable.
- learn all binary pairs first.

And Then?

- do this to all the entire solver.
- or use mix-and-match approach (FRAT).
- What about other uses of CaDiCaL.

Applications — Incremental solving

- non-incremental checking feasible but slow.
- problem changes dependent on answers.
- (L)IDRUP: certifying interactions as well as reasoning.
- not all reasoning is equal (equivalence preserving vs. equisatisfiable).
- for now: equisatisfiable reasoning only for SAT.
- model reconstruction in SAT solvers.
- complete model of original formula in proofs.

Build your own application — the tracer interface [CAV'24]

- API for proof steps.
- captures a range of applications.
- proof formats (DRAT, FRAT, LRAT, VeriPB, IDRUP, LIDRUP)
- beyond SAT proof checking:
- proof extraction (e.g. for MaxSAT [CP'24], Constraint Programming),
- interpolation (model checker CaDiCraig),
- persistent certificate (bit-vector solver in Lean),
- explainability,
- other applications?

What about VeriPB?

- pseudo-Boolean proof version 2.0
- DRAT and LRAT variants
- checked deletions? I always assumed you could not do variable elimination...

What I learned yesterday:

---

① $e \lor x$ $\quad\quad$ $\bar{x} \lor c \lor d$ ④

② $a \lor b \lor x$ $\quad\quad$ $\bar{x} \lor \bar{a} \lor b$ ⑤

③ $\bar{a} \lor \bar{b} \lor x$

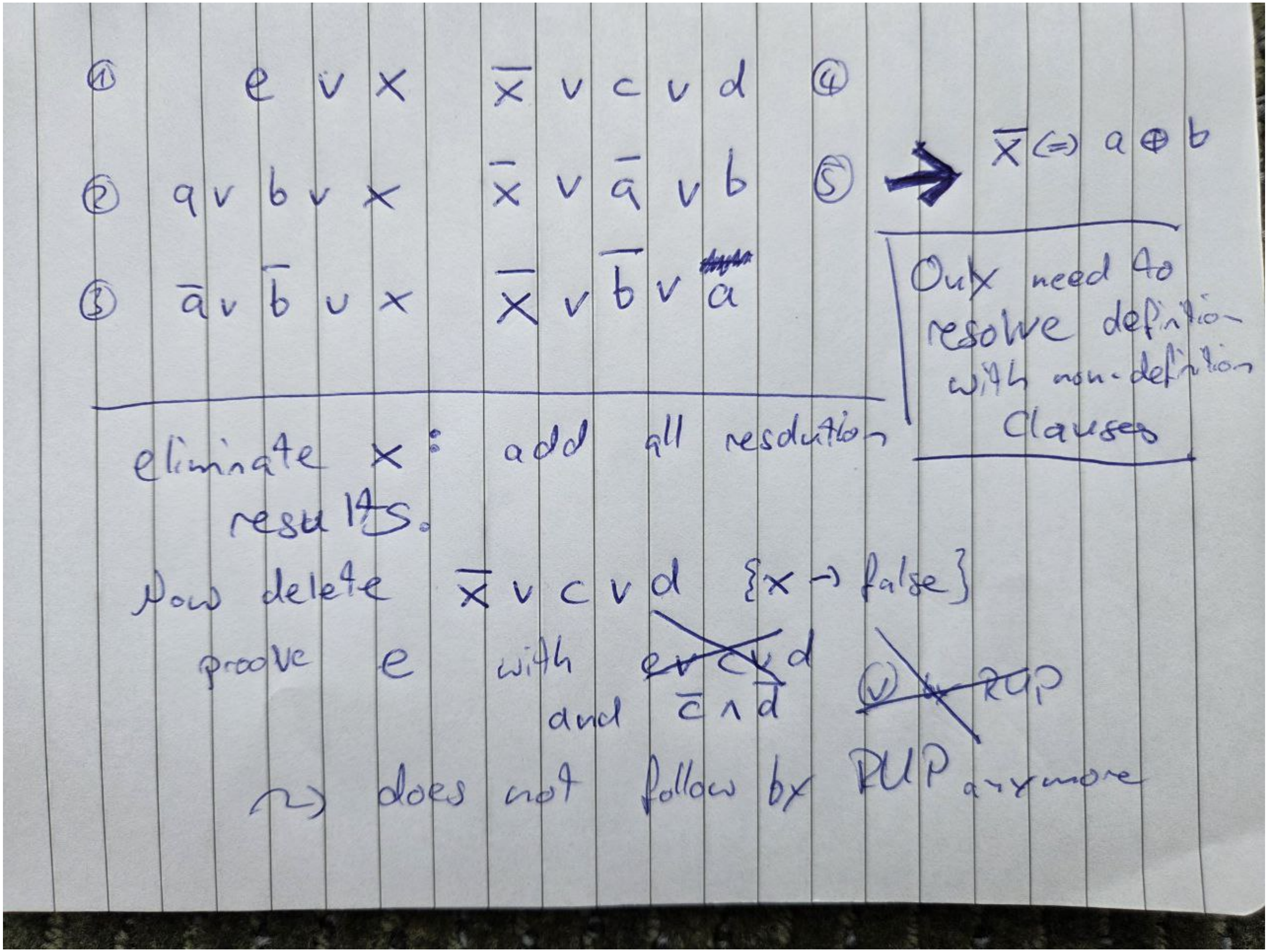
---

eliminate x: add all resolution results.

Now delete $\bar{x} \lor c \lor d$ $\{x \mapsto false\}$

proove e with $e \lor c \lor d$ and $\bar{c} \land \bar{d}$ ✓ by RUP

Not always though...



proof statistics

- sudoku-N30-10.cnf: p cnf 842089 2262758, 44MB
- DRAT — LRAT — VeriPB (RUP) — VeriPB (RES)
- solving time: 8117s — 8933s — 8141s — 9011s
- proof sizes: 6/15GB — 29/62GB — 13GB — 93GB
- checking: 2060s — 93s/254s — N/A — N/A

Bibliography

[CP'25] *Wietze Koops, Daniel Le Berre, Magnus O. Myreen, Jakob Nordström, Andy Oertel, Yong Kiam Tan and Marc Vinyals*. "Practically Feasible Proof Logging for Pseudo-Boolean Optimization"

[CP'24] *Jeremias Berg, Bart Bogaerts, Jakob Nordström, Andy Oertel, Tobias Paxian and Dieter Vandesande*. "Certifying Without Loss of Generality Reasoning in Solution-Improving Maximum Satisfiability"

[CAV'24] *Armin Biere, Tobias Faller, Katalin Fazekas, Mathias Fleury, Nils Froleyks and Florian Pollitt* . "CaDiCaL 2.0"

[SAT'23] *Florian Pollitt, Mathias Fleury and Armin Biere*. "Faster LRAT Checking Than Solving with CaDiCaL"