universität freiburg

SAT Solvers: Verify, Improve, And Use Them In Interactive Theorem Provers

Mathias Fleury

7th of December

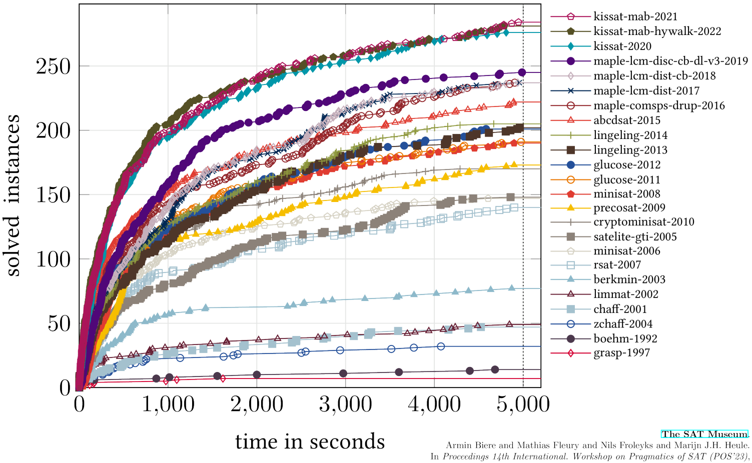
Why SAT?

formal verification	security	bioinformatics	train safety
planning	automated theorem proving	exploit generation	termination rewriting

Encode your problem and then ask a SAT solver (and possibly decode)

Introduction

SAT Competition All Time Winners on SAT Competition 2022 Benchmarks



https://cca.informatik.uni-freiburg.de/satmuseum

The SAT Museum.
Armin Biere and Mathias Fleury and Nils Froleyks and Marijn J.H. Heule.
In *Proceedings 14th International. Workshop on Pragmatics of SAT (POS'23)*,
vol. 3545, CEUR Workshop Proceedings, pages 72-87, CEUR-WS.org 2023.
[paper - bibtex - data - zenodo - ceur - workshop - proceedings]

Contributions

Theorem (Contribution 1 [IJCAR'16, NFM'19, CADE 2023])

IsaSAT is correct (answer ≠ unknown) and terminates.

where unknown = array size larger than 64-bit in

Theorem (Contribution 2, [Wagner's Msc])

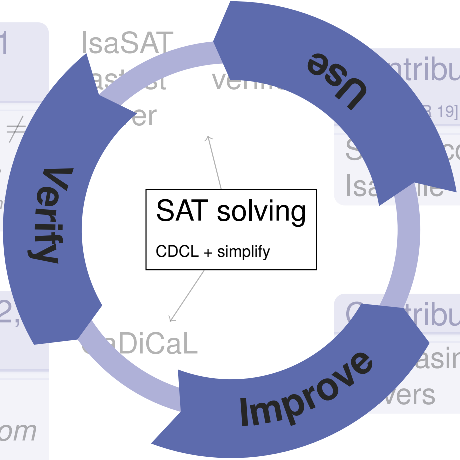
Fixing model is correctly implemented but differs from the paper

ntribution 4 [CADE 2021, PXTP'19 R 19]

S construction in Isa lle

C tribution 3 [JAIR'22]

asing techniques in SAT vers



SAT Solver Verification

Contributions

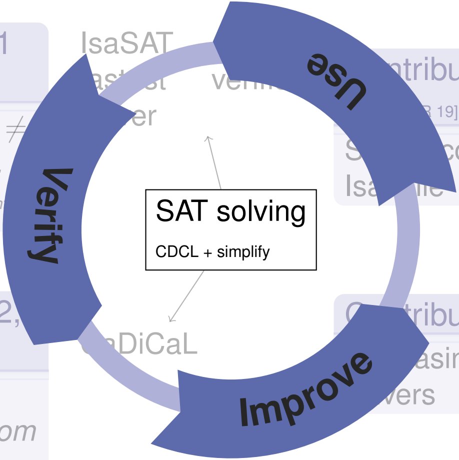
Theorem (Contribution 1 [IJCAR'16, NFM'19, CADE 2023])

IsaSAT is correct (answer ≠ unknown) and terminates.

where unknown = array size larger than 64-bit in

Theorem (Contribution 2, [Wagner's Msc])

Fixing model is correctly implemented but differs from the paper



ntribution 4 [CADE 2021, PXTP'19

R 19]

S construction in

Isa lle

C tribution 3 [JAIR'22]

asing techniques in SAT

vers

Contributions

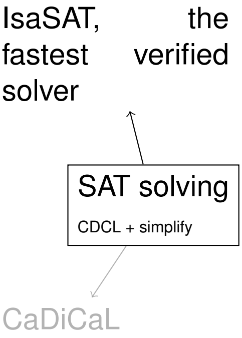
Theorem (Contribution 1 [IJCAR'16, NFM'19, CADE 2023])

IsaSAT is correct (answer $\neq$ unknown) and terminates.

where unknown = array size larger than 64-bit integer

Theorem (Contribution 2, [Wagner's Msc])

Fixing model is correctly implemented but differs from the paper



IsaSAT, the fastest verified solver

SAT solving

CDCL + simplify

CaDiCaL

Contribution 4 [CADE 2021, PXTP'19 and 21, JAR 19]

SMT reconstruction in Isabelle

Contribution 3 [JAIR'22]

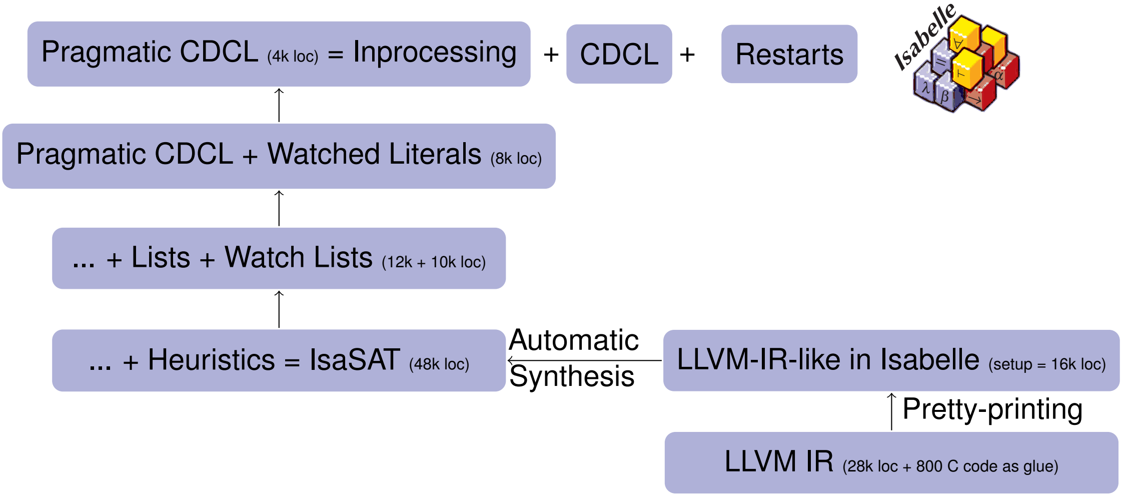
Rephasing techniques in SAT solvers

Refinement in IsaSAT

Pragmatic CDCL (4k loc) = Inprocessing + CDCL + Restarts

Isabelle

Refinement in IsaSAT



Pragmatic CDCL (4k loc) = Inprocessing + CDCL + Restarts

Pragmatic CDCL + Watched Literals (8k loc)

... + Lists + Watch Lists (12k + 10k loc)

... + Heuristics = IsaSAT (48k loc)

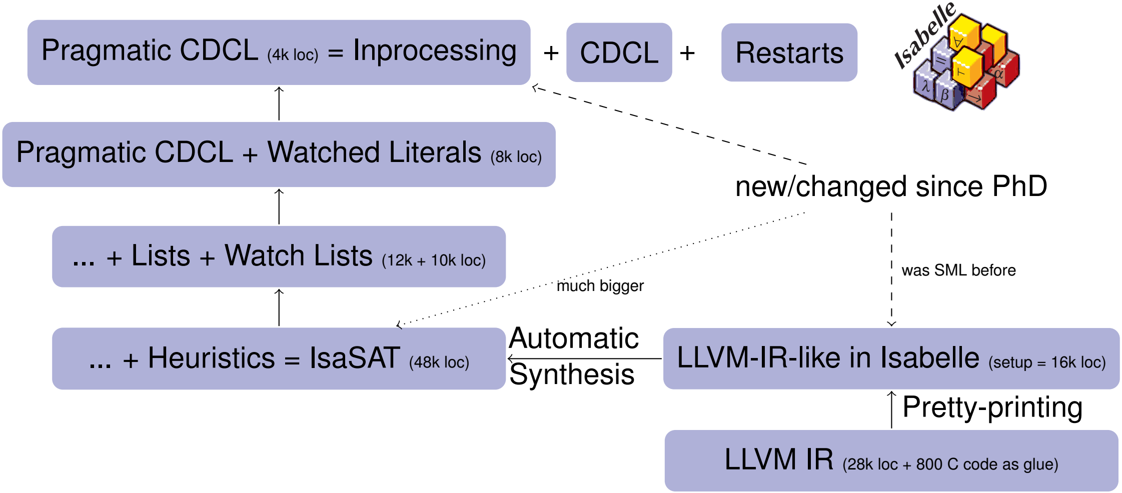
Automatic Synthesis

LLVM-IR-like in Isabelle (setup = 16k loc)

Pretty-printing

LLVM IR (28k loc + 800 C code as glue)

Refinement in IsaSAT



Pragmatic CDCL (4k loc) = Inprocessing + CDCL + Restarts

Isabelle

Pragmatic CDCL + Watched Literals (8k loc)

new/changed since PhD

... + Lists + Watch Lists (12k + 10k loc)

much bigger

was SML before

... + Heuristics = IsaSAT (48k loc)

Automatic Synthesis

LLVM-IR-like in Isabelle (setup = 16k loc)

Pretty-printing

LLVM IR (28k loc + 800 C code as glue)

How Do They Perform?

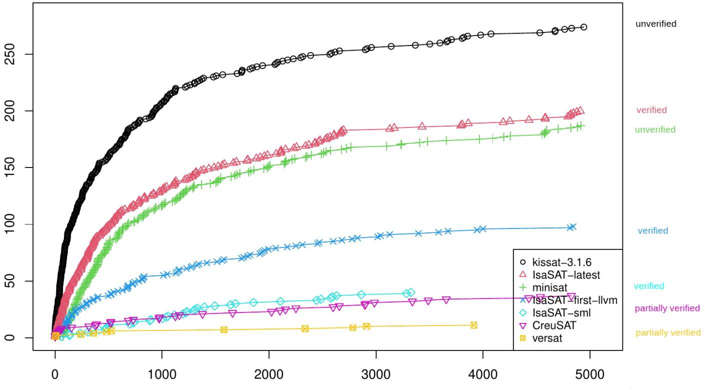


Figure 1: CDF of various solvers on the SC2022 (7 GB, 5000 s)

Contributions

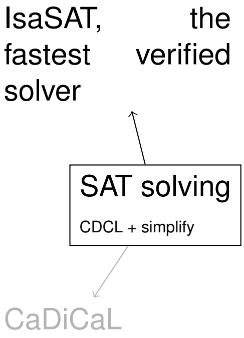
Theorem (Contribution 1 [IJCAR'16, NFM'19, CADE 2023])

IsaSAT is correct (answer $\neq$ unknown) and terminates.

where unknown = array size larger than 64-bit integer

Theorem (Contribution 2, [Wagner's Msc])

Fixing model is correctly implemented but differs from the paper



IsaSAT, the fastest verified solver

SAT solving

CDCL + simplify

CaDiCaL

Contribution 4 [CADE 2021, PXTP'19 and 21, JAR 19]

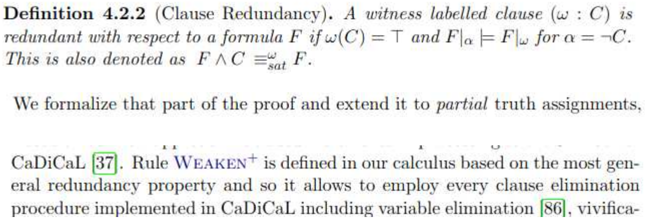
SMT reconstruction in Isabelle

Contribution 3 [JAIR'22]

Rephasing techniques in SAT solvers

Model Reconstruction for Incremental Solving [Msc Thesis, Wagner]

How to simplify clauses when further are coming? [Fazekas, Scholl and Biere, SAT'19]



Definition 4.2.2 (Clause Redundancy). *A witness labelled clause* $(\omega : C)$ *is redundant with respect to a formula* F *if* $\omega(C) = \top$ *and* $F|_\alpha \models F|_\omega$ *for* $\alpha = \neg C$. *This is also denoted as* $F \wedge C \equiv^{\omega}_{sat} F$.

We formalize that part of the proof and extend it to *partial* truth assignments,

CaDiCaL [37]. Rule WEAKEN$^+$ is defined in our calculus based on the most general redundancy property and so it allows to employ every clause elimination procedure implemented in CaDiCaL including variable elimination [86], vivifica-

CaDiCaL does not implement Def 4.2.2.

I did not realize that either before Isabelle refused a proof

Implementation heavily tested... on total modals

Setting phases

Contributions

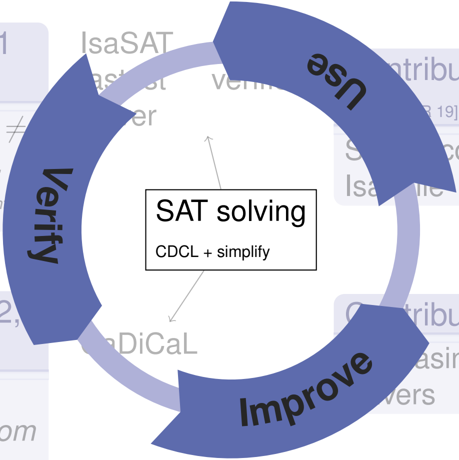
Theorem (Contribution 1 [IJCAR'16, NFM'19, CADE 2023])

IsaSAT is correct (answer ≠ unknown) and terminates.

where unknown = array size larger than 64-bit in

Theorem (Contribution 2, [Wagner's Msc])

Fixing model is correctly implemented but differs from the paper



IsaSAT

as st ver

veri

aDiCaL

ntribution 4 [CADE 2021, PXTP'19

R 19]

S construction in

Isa lle

C tribution 3 [JAIR'22]

asing techniques in SAT

vers

Contributions

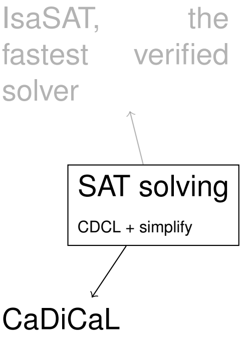
Theorem (Contribution 1 [IJCAR'16, NFM'19, CADE 2023])

IsaSAT is correct (answer $\neq$ unknown) and terminates.

where unknown = array size larger than 64-bit integer

Theorem (Contribution 2, [Wagner's Msc])

Fixing model is correctly implemented but differs from the paper



IsaSAT, the fastest verified solver

SAT solving

CDCL + simplify

CaDiCaL

Contribution 4 [CADE 2021, PXTP'19 and 21, JAR 19]

SMT reconstruction in Isabelle

Contribution 3 [JAIR'22]

Rephasing techniques in SAT solvers

Guessing values

CDCL solvers work by (i) guessing a value, (ii) propagating, and (iii) fixing the assignment.

How do we guess? Old wisdom:

- set to last set value — SAT subproblems remain SAT
- otherwise default to false — closed world assumption

Local search solvers work by randomly flipping one literal as long as no model is found

Guessing values

CDCL solvers work by (i) guessing a value, (ii) propagating, and (iii) fixing the assignment.

How do we guess? Old wisdom:

- set to last set value — SAT subproblems remain SAT
- otherwise default to false — closed world assumption

Local search solvers work by randomly flipping one literal as long as no model is found

SAT as Optimization

New view for CDCL: maximize the partial assignment

- Objective is to maximize the size of the trail without conflict
- Save *maximum consistent trail* as target phases

- Intensification: use target phases

and best phases

- Diversification: rephasing

Autarky detection does not seem important

Include also Local-Search

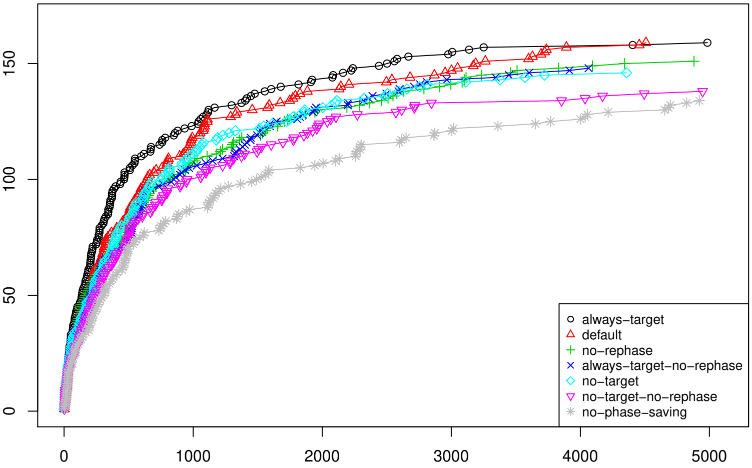
CDCL very good at propagating

Local-Search very bad at propagation chains

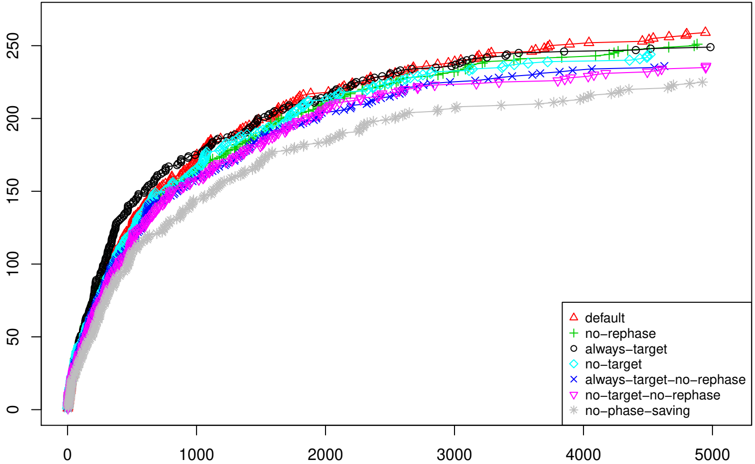
Import phase from CDCL after propagating

use CDCL ignoring conflicts as start point

Kissat, SAT Race 2019, satisfiable only



Kissat, SAT Race 2019, all



SMT Tactic

Contributions

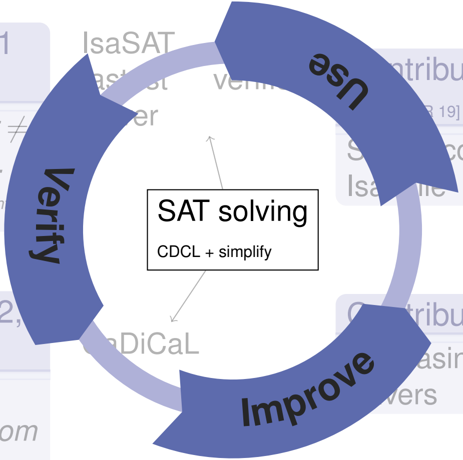
Theorem (Contribution 1 [IJCAR'16, NFM'19, CADE 2023])

IsaSAT is correct (answer ≠ unknown) and terminates.

where unknown = array size larger than 64-bit in

Theorem (Contribution 2, [Wagner's Msc])

Fixing model is correctly implemented but differs from the paper



ntribution 4 [CADE 2021, PXTP'19

R 19]

S construction in

Isa lle

C tribution 3 [JAIR'22]

asing techniques in SAT

vers

Contributions

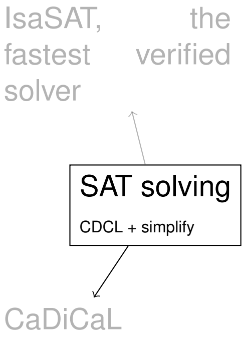
Theorem (Contribution 1 [IJCAR'16, NFM'19, CADE 2023])

IsaSAT is correct (answer $\neq$ unknown) and terminates.

where unknown = array size larger than 64-bit integer

Theorem (Contribution 2, [Wagner's Msc])

Fixing model is correctly implemented but differs from the paper



IsaSAT, the fastest verified solver

SAT solving

CDCL + simplify

CaDiCaL

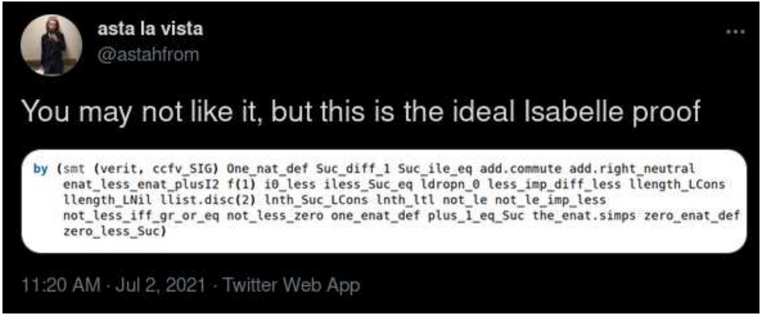
Contribution 4 [CADE 2021, PXTP'19 and 21, JAR 19]

SMT reconstruction in Isabelle

Contribution 3 [JAIR'22]

Rephasing techniques in SAT solvers

Idea: Click on a Button



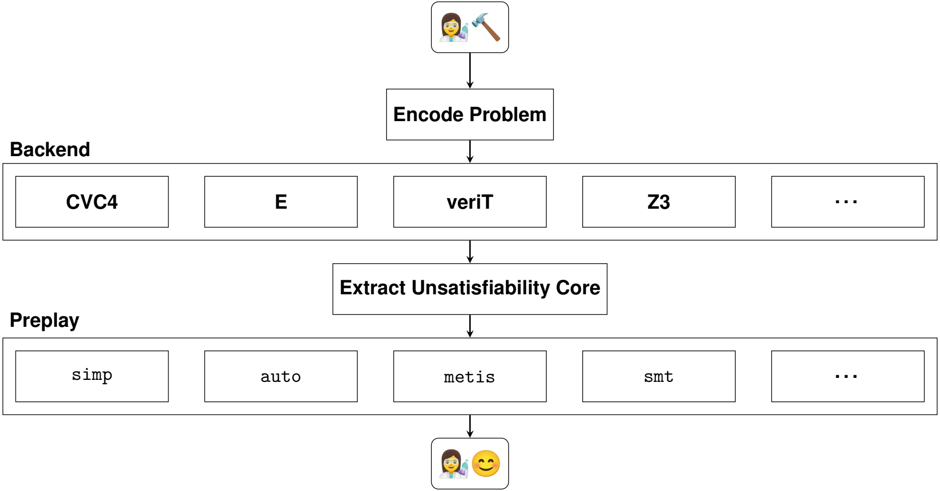
asta la vista
@astahfrom

You may not like it, but this is the ideal Isabelle proof

```
by (smt (verit, ccfv_SIG) One_nat_def Suc_diff_1 Suc_ile_eq add.commute add.right_neutral
    enat_less_enat_plusI2 f(1) i0_less iless_Suc_eq ldropn_0 less_imp_diff_less llength_LCons
    llength_LNil llist.disc(2) lnth_Suc_LCons lnth_ltl not_le not_le_imp_less
    not_less_iff_gr_or_eq not_less_zero one_enat_def plus_1_eq_Suc the_enat.simps zero_enat_def
    zero_less_Suc)
```

11:20 AM · Jul 2, 2021 · Twitter Web App

Interactive Theorem Proving with Sledgehammer



👩‍🔬🔨

Encode Problem

Backend

CVC4

E

veriT

Z3

···

Extract Unsatisfiability Core

Preplay

`simp`

`auto`

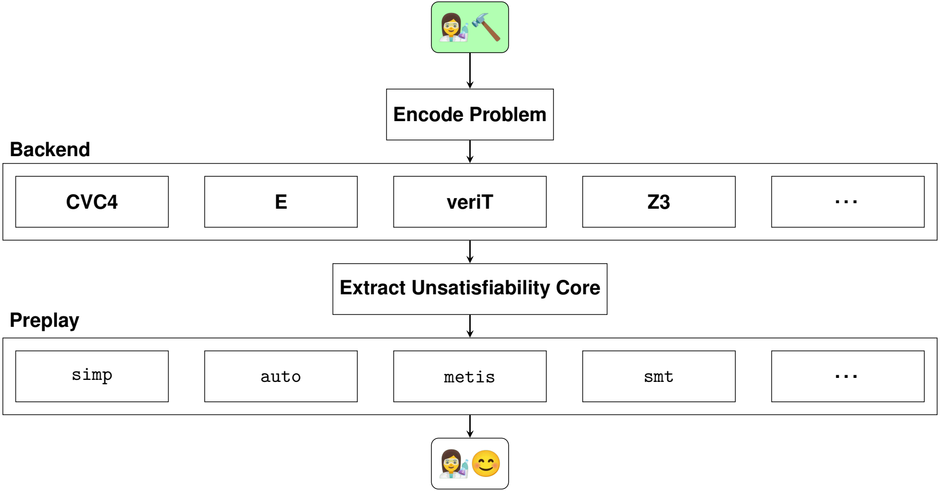
`metis`

`smt`

···

👩‍🔬😊

Interactive Theorem Proving with Sledgehammer



👩‍🔬🔨

Encode Problem

Backend

CVC4

E

veriT

Z3

· · ·

Extract Unsatisfiability Core

Preplay

`simp`

`auto`

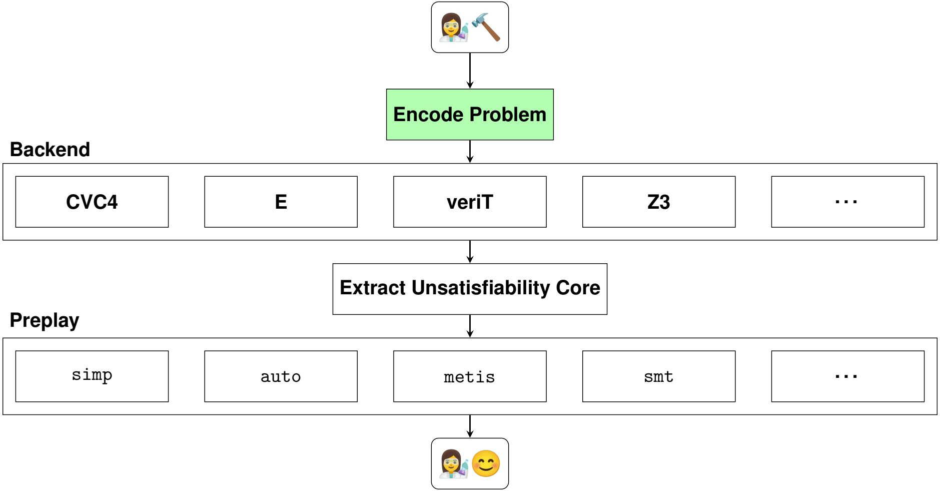
`metis`

`smt`

· · ·

👩‍🔬😊

Interactive Theorem Proving with Sledgehammer



👩‍🔬🔨

Encode Problem

Backend

CVC4

E

veriT

Z3

···

Extract Unsatisfiability Core

Preplay

`simp`

`auto`

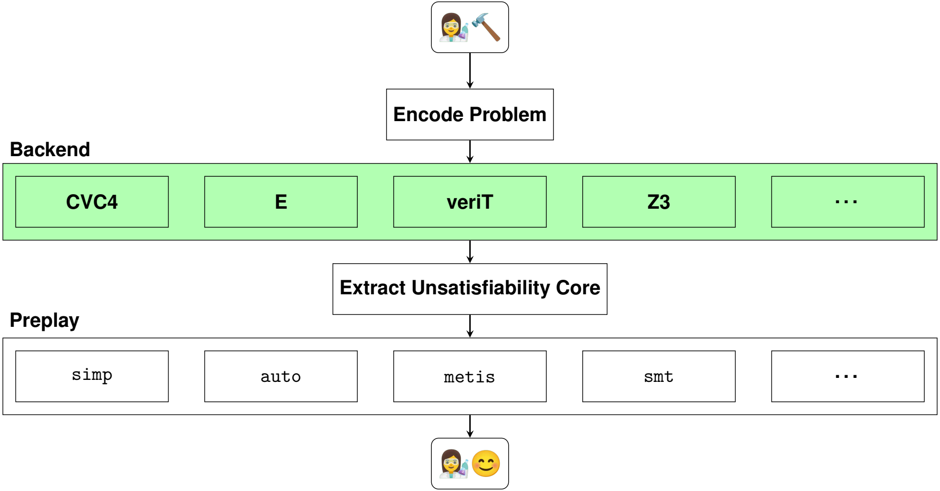
`metis`

`smt`

···

👩‍🔬😊

Interactive Theorem Proving with Sledgehammer



👩‍🔬🔨

Encode Problem

Backend

CVC4

E

veriT

Z3

···

Extract Unsatisfiability Core

Preplay

`simp`

`auto`

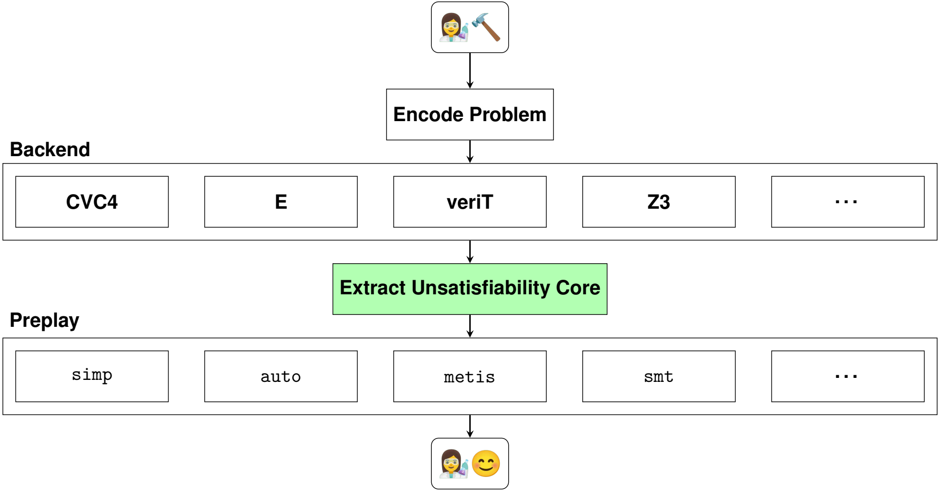
`metis`

`smt`

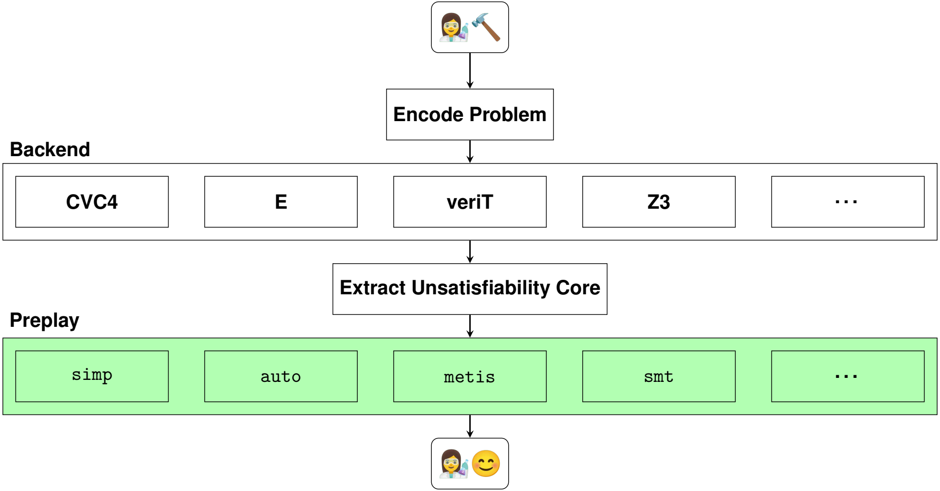
···

👩‍🔬😊

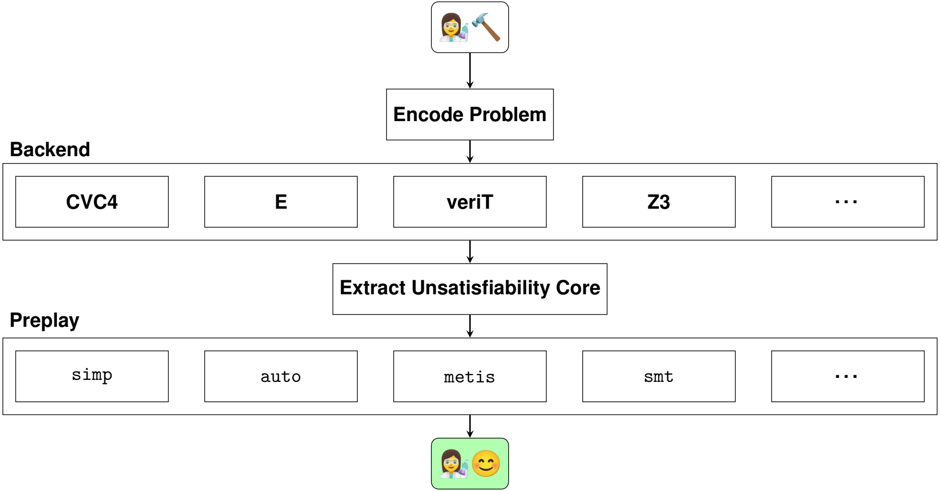
Interactive Theorem Proving with Sledgehammer



Interactive Theorem Proving with Sledgehammer



Interactive Theorem Proving with Sledgehammer



👩‍🔬🔨

Encode Problem

Backend

CVC4

E

veriT

Z3

···

Extract Unsatisfiability Core

Preplay

`simp`

`auto`

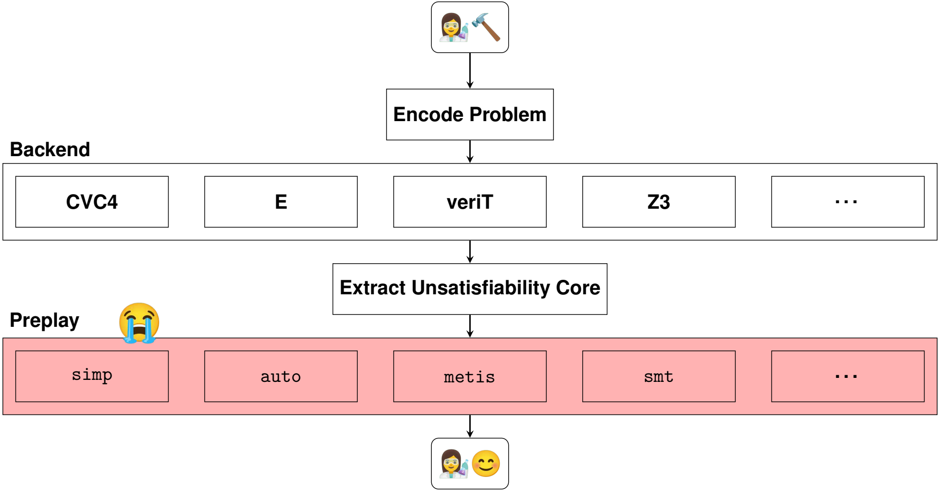
`metis`

`smt`

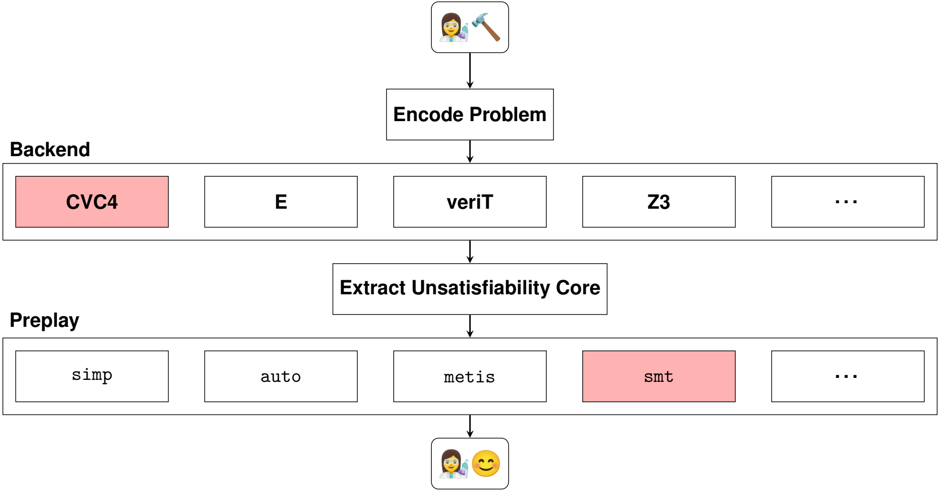
···

👩‍🔬😊

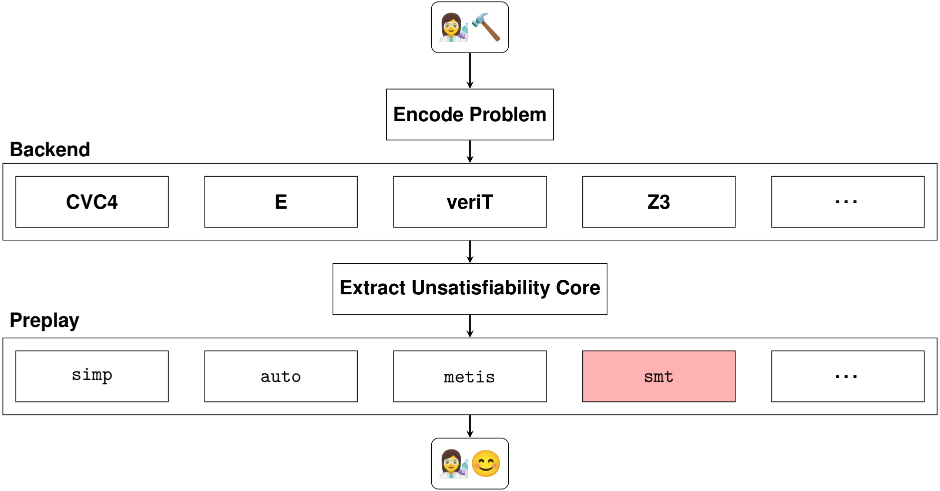
Interactive Theorem Proving with Sledgehammer



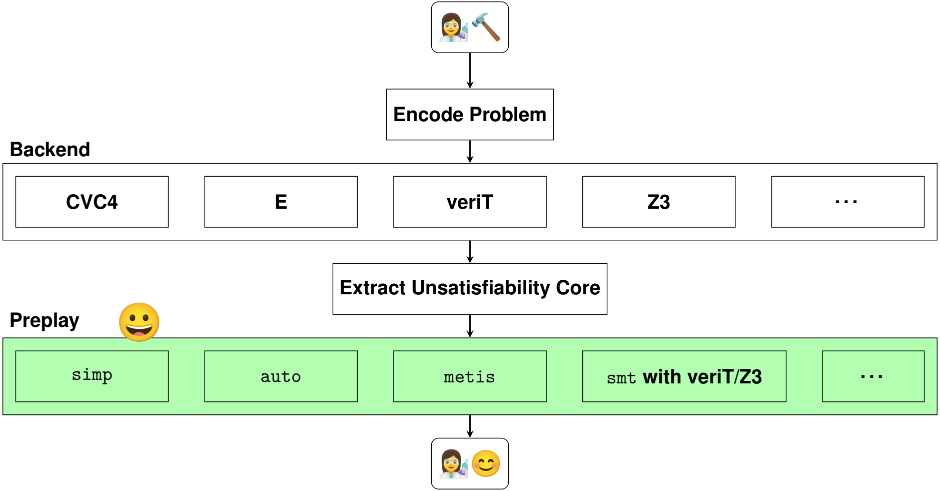
Interactive Theorem Proving with Sledgehammer



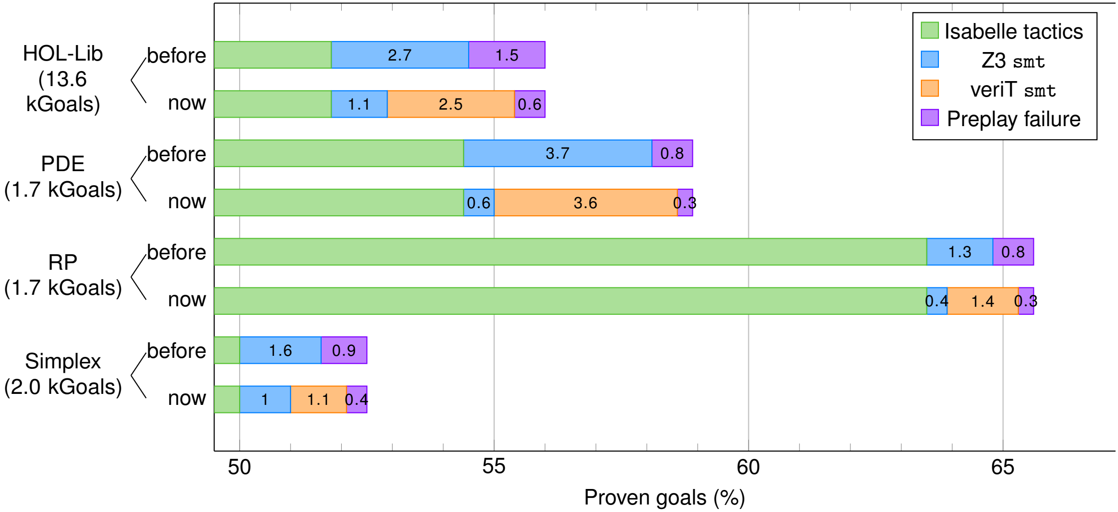
Interactive Theorem Proving with Sledgehammer



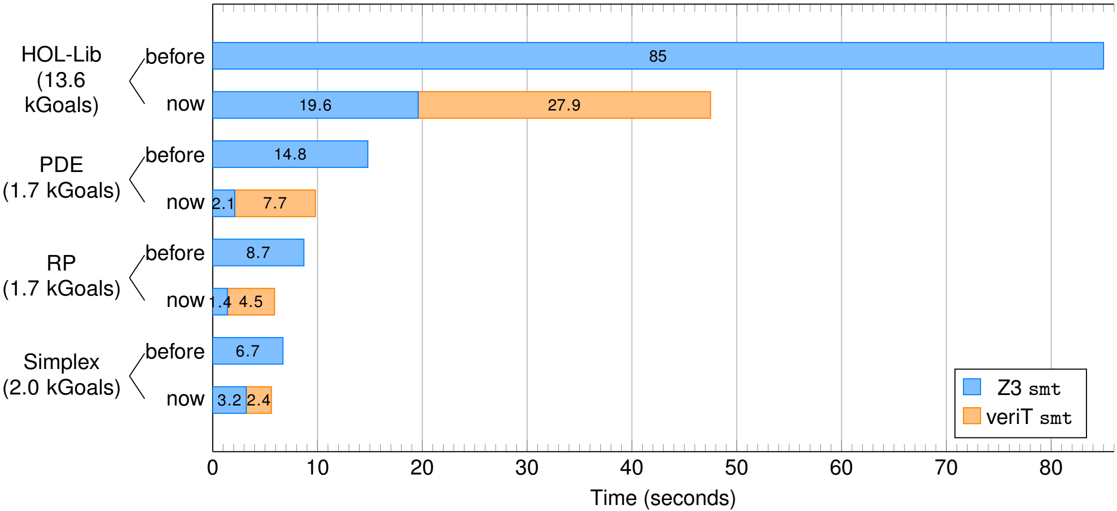
Interactive Theorem Proving with Sledgehammer



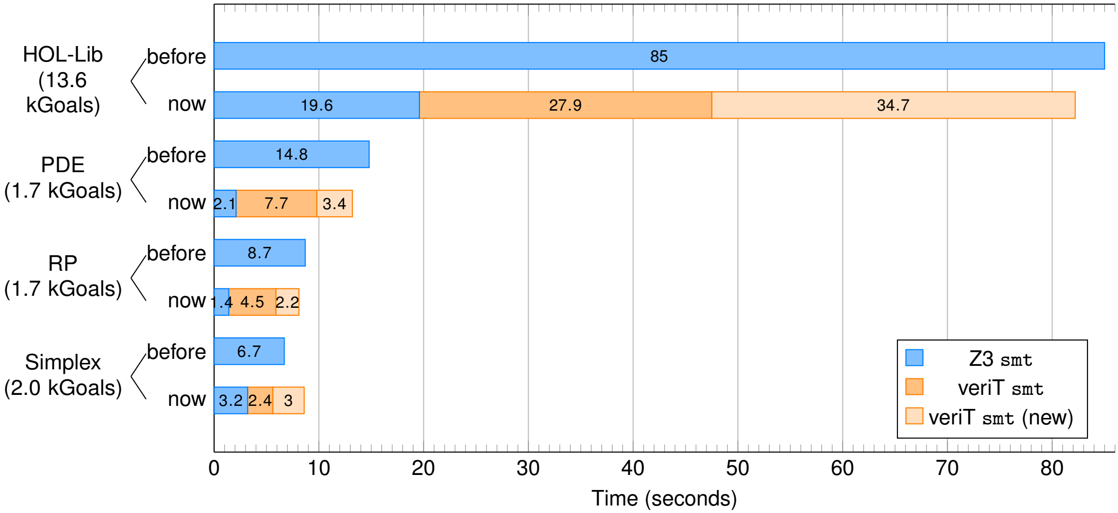
CVC4: Preplay Success Rate



CVC4: Preplay Time (smt only)



CVC4: Preplay Time (smt only)



cvc5

With Hanna Lachnitt, and the cvc5[1] team [SMT'2023 workshop, submitted]

- support for Alethe proof format is ongoing with more details
- work for RARE rules: solver rules can be extended

ongoing work

- detailed bitvector reconstruction

- ongoing work on the cvc5 side, not only on the Isabelle side

[1] yes it is CVC4 and cvc5 with this capitalization

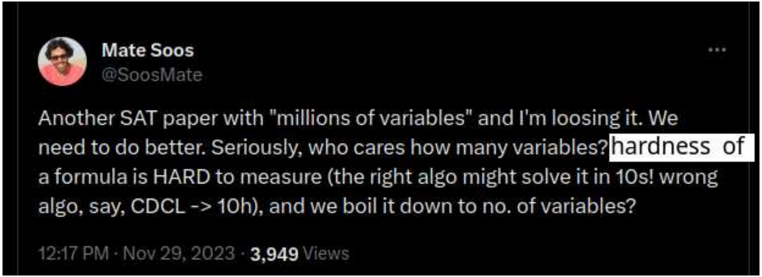
Conclusion

Conclusion

Ongoing work:

- implement reconstruction in IsaSAT incompatible with current inprocessing
- model-checking proof format and beyond and incremental with LRAT from [SAT'23]
- understanding performance of SAT solvers minimization is complete [SAT'21], options [POS'23]

Why do Techniques Work?



Mate Soos
@SoosMate

Another SAT paper with "millions of variables" and I'm loosing it. We need to do better. Seriously, who cares how many variables? hardness of a formula is HARD to measure (the right algo might solve it in 10s! wrong algo, say, CDCL -> 10h), and we boil it down to no. of variables?

12:17 PM · Nov 29, 2023 · **3,949** Views

with a fixed typo

Theorem (Contribution 1 [IJCAR'16, NFM'19, CADE 2023])

IsaSAT is correct (answer $\neq$ unknown) and terminates.

where unknown = array size larger than 64-bit integer

Theorem (Contribution 2, [Wagner's Msc])

Fixing model is correctly implemented but differs from the paper

Contribution 4 [CADE 2021, PXTP'19 and 21, JAR 19]

SMT reconstruction in Isabelle

Contribution 3 [JAIR'22]

Rephasing techniques in SAT solvers

Appendix start