SMT-COMP 2023
18th International Satisfiability Modulo Theory Competition

Martin Bromberger Jochen Hoenicke François Bobot

CEA List, France

MPI für Informatik, Germany

Certora, Israel

July 5, 2023

SMT-COMP

Annual competition for SMT solvers
on (a selection of) benchmarks from SMT-LIB

History

2005	first competition
2013	evaluation instead of competition
2014	since then hosted by StarExec

Goals:

- spur development of SMT solver implementations
- promote SMT solvers and their usage
- support the SMT-LIB project
  - to promote and develop the SMT-LIB format
    - model validation
    - proof checking
  - to collect relevant benchmarks
- engage and include new members

SMT Solvers and SMT-LIB

SMT Solver

- checks formulas in SMT-LIB format for satisfiability modulo theories

SMT-LIB is

1. a language in which benchmarks are written
2. a community effort to collect benchmarks

Non-incremental

434 212 instances (+42849)
with 1 query each
in 83 logics (+2).

Incremental

43 287 instances (+2)
with 34 036 491 queries (+37 556)
in 38 logics.

SMT Solvers and SMT-LIB

SMT Solver

- checks formulas in SMT-LIB format for satisfiability modulo theories

SMT-LIB is

1. a language in which benchmarks are written
2. a community effort to collect benchmarks

Non-incremental

434 212 instances (+42849)
with 1 query each
in 83 logics (+2).

Incremental

43 287 instances (+2)
with 34 036 491 queries (+37 556)
in 38 logics.

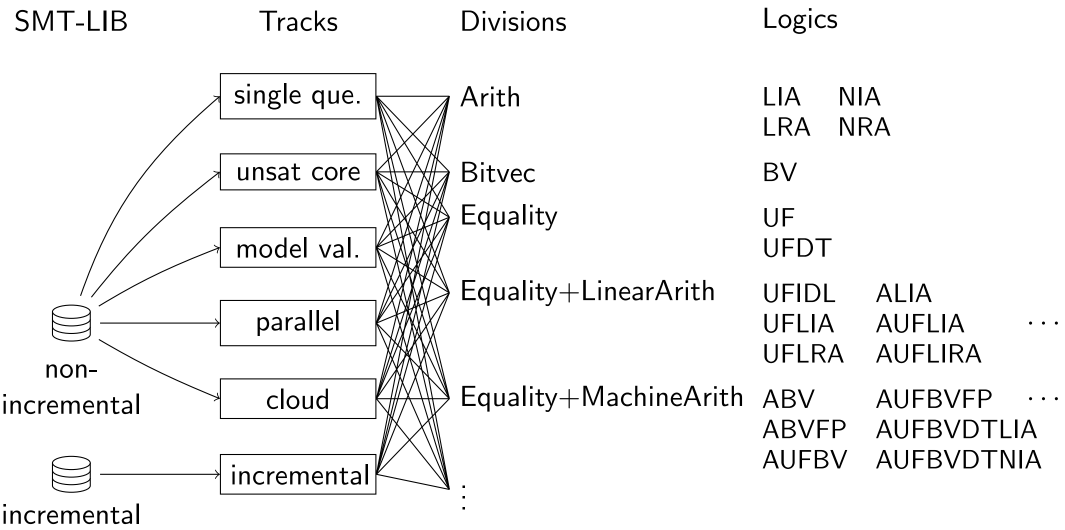
Selected Non-incremental

227 940 instances

Selected Incremental

22 302 instances

Competition Overview



SMT-COMP Tracks (traditional)

Single Query (SQ) Track

- Determine satisfiability of one problem
- Solver answers sat/unsat/unknown

Unsat Core Track

- Find small unsatisfiable subset of input.
- Solver answers unsat + list of formulas.

Model Validation Track

- Find a model for a satisfiable problem.
- Solver answers sat + value for each non-logical symbol.

Incremental Track

- Solve many small problems interactively.
- Solver acks commands and answers sat/unsat for each check.

SMT-COMP Tracks (experimental)

Model Validation

- Division with quantifier-free floating-point logics
- Model validation with Dolmen (thanks to Gillaume Bury and François Bobot)

Cloud and Parallel Track (sponsored by AWS, led by Mike Whalen)

- Solve a large problem over the cloud (or a big computer)
  - 100 machines, 1600 cores, 6400 GB of memory (cloud)
  - 64 cores, 256 GB of memory (parallel)
- Solver answers sat/unsat/unknown

SMT-COMP Tracks (experimental)

Model Validation

- Division with quantifier-free floating-point logics
- Model validation with Dolmen (thanks to Gillaume Bury and François Bobot)

Cloud and Parallel Track (sponsored by AWS, led by Mike Whalen)

- Solve a large problem over the cloud (or a big computer)
  - 100 machines, 1600 cores, 6400 GB of memory (cloud)
  - 64 cores, 256 GB of memory (parallel)
- Solver answers sat/unsat/unknown

Proof Exhibition Track

- Solver submitted together with a checker for unsatisfiability proofs
- No predefined format or checker
- No ranking
- Qualitative assessment

SMT-COMP Tracks (experimental)

Model Validation

- Division with quantifier-free floating-point logics
- Model validation with Dolmen (thanks to Gillaume Bury and François Bobot)

Cloud and Parallel Track (sponsored by AWS, led by Mike Whalen)

- Solve a large problem over the cloud (or a big computer)
  - 100 machines, 1600 cores, 6400 GB of memory (cloud)
  - 64 cores, 256 GB of memory (parallel)
- Solver answers sat/unsat/unknown

Proof Exhibition Track

- Solver submitted together with a checker for unsatisfiability proofs
- No predefined format or checker
- No ranking
- Qualitative assessment

As last year the sat/unsat results from sound solvers in SQ were used to include benchmarks on the MV, UC and PE tracks.

Tracks, Solvers, Divisions, and Benchmarks

Teams: 25 (+4)

Track	Solvers	Divisions	Benchmarks
Single Query	22(=)	19(=)	113 139
Incremental	7(-1)	17(=)	22 301
Unsat Core	6(=)	16(-1)	72 958
Model Validation	11(+3)	13(+ 6 exp.)	61 083
Proof Exhibition	4(=)	19 exp.	59 114
Parallel	3(-1)	4 exp.	400
Cloud	2(-2)	4 exp.	400

Number in parenthesis shows changes from 2022

Participants

SMT-COMP 2022 participants rely on multiple reasoning frameworks:

- CDCL(T), Saturation, MCSAT, CP
- automata
- finite domain
- local search
- besides wrappers extending the scope of existing solvers

Seven new solvers participated:

- iProver (Korovin et al.)
- SMT-RAT-MCSAT (Jasper Nalbach et al.)
- UltimateIntBlastingWrapper+SMTInterpol (Max Barth et al.)
- Yaga (Hanák et al.)
- Z3-alpha (Lu et al.)
- Z3-Noodler (Havlena et al.)
- Z3-Owl (Ma et al.)

Solver Presentation

Bitwuzla at the SMT-COMP'23

Aina Niemetz, Mathias Preiner

Tracks/Divisions: `^(QF_)?(A)?(UF)?(BV|FP|FPLRA)+$`
in tracks **Single Query**, **Incremental**, **Unsat Core**, and **Model Validation**

Hightlights

- **New SMT solver** for theories **A**, **BV**, **FP**, **UF** + quantifiers
- **Rewrite from scratch**
- previous versions were extended fork from Boolector
- system description at **CAV 2023**

https://bitwuzla.github.io

Stanford | Center for Automated Reasoning

COLIBRI(2023) Bruno Marre et al

- CP solver:
  - No SAT solver
  - Combination done by explicitely building a model
- main theories: **FP**, **LIA** $\leftrightarrow$ **BV**

- Some fixes
- Add some reasoning for transcendental functions

cvc5 at the SMT Competition 2023

L. Aniva H. Barbosa C. Barrett M. Brain V. Camillo G. Kremer H. Lachnitt A. Mohamed
M. Mohamed A. Niemetz A. Nötzli A. Ozdemir M. Preiner A. Reynolds Y. Sheng C. Tinelli A. Wilson Y. Zohar

cvc5 1.0.5

- Support for all standardized SMT-LIB theories
- User-friendly API

Features/Improvements

- New handwritten parser and lexer
- Bit-vector solver, integrating efficient SAT solvers, e.g., CaDiCaL, with CDCL($\mathcal{T}$)
- Syntax-guided and model-based quantifier instantiation
- Minor improvements to the lemma schemas used for the theory of strings
- Better performance for proof generation

Configurations

cvc5 entered all divisions in all tracks (non-experimental).

- Single query track: Sequential portfolio
- Unsat-core track: Based on new proof module and assumptions in the SAT solver

Follow the development: https://cvc5.github.io/

cvc5-NRA-LS

video: `slides-cvc5-nra-ls.mp4`

iProver

video: `slides-iProver.mp4`

ismt, Yices-ismt

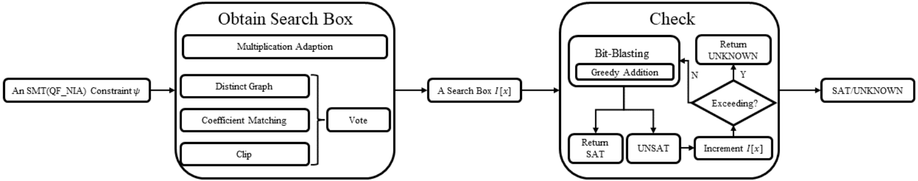
video: `slides-ismt.mp4`

ISMT

Fuqi Jia, Rui Han, Minghao Liu, Cunjing Ge, Pei Huang, Feifei Ma, Jian Zhang.

- ISMT is a pure bit-blasting based SMT solver, participating QF_NIA model validation track;
- YICES-ISMT combines YICES2 for unsat reasoning, participating in QF_NIA single query track.

- ISMT:



- YICES-ISMT: YICES(ϕ) → ISMT(ϕ) → YICES($\phi \wedge \psi$).
  - ψ rule out failed space.

Dependencies
- *Yices 2.6.2*
- *Libpoly v0.1,11*
- *CaDiCal 1.5.2*

- Link: https://github.com/MRVAPOR/BLAN

OpenSMT at SMT-COMP 2023

- Interpolating CDCL(T) SMT solver
  - Developed at University of Lugano, Switzerland
  - `https://github.com/usi-verification-and-security/opensmt`
- Support for *linear arithmetic*, uninterpreted functions, arrays
- Alternative lookahead core
- Used in Horn solver Golem
- New in 2023 edition ☺
  - Support for incrementality and interpolation in lookahead core
  - Theory combination with arrays
- On hold in 2023 edition ☹
  - Proof track
  - Parallel and cloud track

Ostrich

video: `slides-Ostrich.mkv`

SMTInterpol

Jochen Hoenicke, Tanja Schindler, . . .

UNI FREIBURG

Interpolating SMT solver

- based on CDCL(T)
- for Arrays, Uninterpreted Functions, Linear Integer and Real Arithmetic
    - plus `div` and `mod` with constants,

  and Data Types
- supports quantifiers
- produces models, proofs, and unsat cores
- computes sequence and tree interpolants

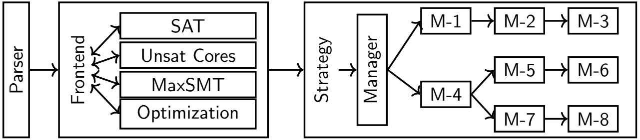
SMTInterpol at SMT-COMP 2023

- models for data type logics
- optimize size of proofs
- proof check in single query/unsat core

Try it in your browser:

`https://tinyurl.com/smtinterpol`

SMT-RAT 23.05



20

UltimateEliminator+MathSAT

video: `slides-UltimateEliminator.mkv`

Vampire 4.8

Reger, Suda, Voronkov, Kovács, Bhayat, Gleiss, Hajdu, Hozzová, Rath, Rawson, Schoisswohl

`https://vprover.github.io/`

- General Approach: proof search using the Superposition Calculus (and finite model finding in UF)
- SMT Logics: A, DT, LIA, LRA, NIA, NRA, UF (all with quantifiers)
- Uses a portfolio of strategies
  Parallel track: parallelize strategies
  Cloud track: randomize problem and strategies per node
- Use Z3 for ground reasoning (AVATAR)
- New for arithmetic: ALASCA Calculus and new simplification rules

Yaga—MCSat-based SMT solver

- Developed at Charles University, Prague, Czech Republic
  - Drahomír Hanák, Martin Blicha, Jan Kofroň
- Student project, from December 2022
- Support for QF_LRA
- Support for models
- Future work
  - More theories
  - Model-based interpolation
- `https://github.com/d3sformal/yaga`

Yices2 in SMT-COMP 2023 (SMT-comp 2023)

Bruno Dutertre, Aman Goel, Stéphane Graham-Lengrand, Ahmed Irfan, Dejan Jovanović, Ian A. Mason
https://yices.csl.sri.com/

Two solvers: CDCL(T) & MCSAT
Support:

- Quantifier-free: non-linear arithmetic (MCSAT only), linear arithmetic, bitvectors, uninterpreted functions, and arrays.
- With quantifiers: uninterpreted functions only, via E-graph matching and model-based instantiation.

Bitvectors: Yices 2/CDCL(T) uses bitblasting.
For QF_BV, it can optionally use third-party backend SAT solvers: CaDiCaL, CryptoMiniSat, and Kissat (the SMT-comp version uses Kissat for single-query and model validation tracks).
Functionalities: incremental and push/pop modes, unsat cores, model minimization and implicants, Model-Based Over-approximations, Model-Based Under-approximations, Craig Interpolants.

2023 NEW:

- arrays in MCSAT
- new variable decision and clause scoring heuristics in MCSAT.

YicesQS, an extension of Yices2 for quantifiers (SMT-comp 2023)

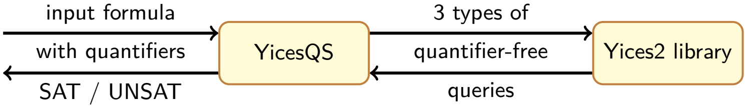
Stéphane Graham-Lengrand https://github.com/disteph/yicesQS

Same solver as in the 2022 SMT-comp.[1]

YicesQS implements a variant of the QSMA algorithm presented at CADE'2023:
https://www.csl.sri.com/users/sgl/Work/Reports/CADE2023.pdf
Lazy approach to quantifier elimination based on Model-Based Over-approximations (MBO) and Model-Based Under-approximations (MBU). YicesQS is written in OCaml, using Yices2 as a library via its OCaml bindings.

2023: YicesQS entered *NRA*, *NIA*, *LRA*, *LIA* and *BV* (single-track), & generally targets complete theories with procedures for answering 3 types of quantifier-free queries:

- *Satisfiability modulo assignment / modulo a model* (here relying on MCSAT)
- *MBU* (here using invertibility conditions for BV, CAD projections for arithmetic)
- *MBO* (here again relying on MCSAT, incl. CAD for arithmetic)



[1] YicesQS-2023 (Starexec solver 45053, too late for 2023 SMT-comp) is way better at BV, solves 805/970 instances out of the 2022 single-track BV selection.

Z3-Z3++

https://youtu.be/fBBOWxxf9vA

Other participants

- Q3B
- Q3B-pBDD
- STP
- UltimateIntBlastingWrapper+SMTInterpol
- z3-alpha
- Z3-Noodler
- Z3-Owl

Non-Competitive Solvers

Submitted by organisers

- Best solvers, per division, from previous years (27 Solvers)

Submitted by participants

- Fixed solvers (OSTRICH, Z3-Owl, Bitwuzla, Yices2, Z3-Noodler, iProver)

Scoring

Computing scores:

- Single Query/Parallel/Cloud: number of solved instances
- Incremental: number of solved queries
- Unsat Core: number of top-level assertions removed
- Model Validation: number of solved instances with correct models

Error scores:

- All Tracks: given for sat reply for unsat instance, or vice versa
- Unsat Core: given if returned core is satisfiable.
- Model Validation: given if given model evaluates formula to false

Error scores are **draconian**.

Score and Ranking

In each track we collect different scores:

- Sequential score (SQ, UC, MV): all time limits apply to cpu time
- Parallel score (all): all time limits apply to wallclock time
- SAT score (SQ): parallel score for satisfiable instances
- UNSAT score (SQ): parallel score for unsatisfiable instances
- 24s (SQ): parallel score with time limit of 24s

Division ranking (for each score)

- For each division, one winner is declared

Two competition-wide rankings (for each score)

- Biggest lead: division winner with most score difference to second place
- Largest contribution: improvement each solver provided to a virtual best solver

Results

$\top$	satisfiable	$\bot$	unsatisfiable
;	sequential	$\parallel$	parallel
24	less than 24s	inc	incremental
uc	unsat core	mv	model validation

Experimental track are added in the slides but are not present in the certificates

Bitwuzla

Overall Winner
BIGGEST LEAD(inc)

Winner of the Divisions
BITVEC($\top$,24), EQUALITY+MACHINEARITH(inc), FPARITH(;,$\|$,$\top$,$\bot$,24,inc,uc), QF_ADT+BITVEC(mv), QF_BITVEC(inc), QF_EQUALITY+BITVEC(;,$\|$,$\top$,$\bot$,24,inc,mv), QF_FPARITH(;,$\|$,$\top$,$\bot$,24,inc,mv)

Winner of the Logics (where it did not win the corresponding division)
ABVFP($\top$,24), ABVFPLRA(24), AUFBV(;,$\|$,$\top$,$\bot$,24), AUFBVFP(;,$\|$,$\top$,$\bot$,24), QF_ABVFP(uc), QF_ABVFPLRA(uc), QF_BVFP(uc), QF_BVFPLRA(uc), UFBV(;,$\|$,$\top$,$\bot$,24), UFBVFP(;,$\|$,$\bot$)

COLIBRI

Winner of the Logics

QF_ABVFPLRA$_{(;,\|,\perp,24)}$, QF_FP$_{(\perp,24)}$, QF_FPLRA$_{(\perp,24)}$

cvc5

Overall Winner

Biggest Lead(;,∥,⊤,⊥,uc), Largest Contribution(;,∥,⊤,⊥,24,inc,uc)

Winner of the Divisions

Arith(;,∥,⊥,inc,uc), Bitvec(;,∥,⊥,inc,uc), Equality(;,∥,⊤,⊥,inc,uc,cloud), Equality+LinearArith(;,∥,⊤,⊥,24,inc,uc,cloud), Equality+MachineArith(;,∥,⊤,⊥,24,uc), Equality+NonLinearArith(;,∥,⊤,⊥,24,inc,uc), QF_Datatypes(;,∥,⊤,⊥,uc$^{;}$), QF_Equality(inc), QF_Equality+Bitvec+Arith(inc), QF_Equality+NonLinearArith(;,∥,⊥,24,inc,uc,mv), QF_FPArith(uc), QF_LinearIntArith(⊥), QF_LinearRealArith(⊥), QF_NonLinearRealArith(⊥,24), QF_Strings(;,∥,⊤,⊥)

Winner of the Logics (where it did not win the corresponding division)

LIA(⊤,24), NIA(⊤,24), QF_ALIA(inc), QF_AUFLIA(uc,mv), QF_AX(mv), QF_BVFPLRA(inc), QF_FP(⊤,mv), QF_IDL(uc), QF_NIRA(;,∥,⊥), QF_UFBVDT(;,∥,⊤,⊥,24), QF_UFDTLIRA(;,∥,⊤,⊥,24,uc), QF_UFLRA(uc),

iProver

Winner of the Divisions
EQUALITY(parallel), EQUALITY+LINEARARITH(parallel)

Winner of the Logic (where it did not win the corresponding division)
ANIA($\bot$)

OpenSMT

Winner of the Divisions

QF_EQUALITY+LINEARARITH($\perp$), QF_LINEARREALARITH($\top$,inc,mv)

Winner of the Logics (where it did not win the corresponding division)

QF_LIA(;,$\parallel$), QF_LRA(;,$\parallel$,$\perp$,24), QF_UFIDL(;,$\parallel$,$\top$,mv)

OSTRICH

Winner of the Logic

$\mathrm{QF\_S}(;,\|,\perp)$

SMTInterpol

Overall Winner
BIGGEST LEAD(24)

Winner of the Divisions
QF_ADT+LINARITH(mv), QF_DATATYPES(24,uc$^{\|}$,mv),
QF_EQUALITY+LINEARARITH(;,‖,⊤,inc,mv), QF_NONLINEARINTARITH(inc)

Winner of the Logics (where it did not win the corresponding division)
ALIA(⊤,uc), AUFDTLIA(uc), QF_ALIA(uc), QF_ANIA(;,‖,⊤,⊥,24,inc,uc),
QF_AUFNIA(;,‖,⊤,⊥,24,uc), QF_LIA(⊥), QF_UF(uc$^{\|}$), QF_UFDT(uc$^{;}$),
QF_UFDTLIA(⊥,24,uc), UFIDL(⊤), UFLIA(⊤)

STP

Winner of the Division

QF_BITVEC(;,$\|$,$\top$,$\bot$,24,mv)

Vampire

Overall Winner
BIGGEST LEAD(cloud,parallel)

Winner of the Divisions
ARITH(cloud,parallel), EQUALITY(24),
EQUALITY+NONLINEARARITH(cloud,parallel)

Winner of the Logics (where it did not win the corresponding division)
ALIA(;,$\parallel$,$\perp$,24), AUFLIA($\perp$,24,cloud,parallel), AUFLIRA(24),
AUFNIRA($\parallel$,$\perp$,24), UF(;,$\parallel$,$\top$), UFDTLIA(24,uc), UFDTNIA(;,$\parallel$,$\perp$,24,uc),
UFLIA(cloud,parallel)

Yices2

Winner of the Divisions

QF_BITVEC(uc), QF_EQUALITY(;,‖,⊤,⊥,24,uc,mv), QF_EQUALITY+BITVEC(uc), QF_EQUALITY+LINEARARITH(24,uc), QF_EQUALITY+NONLINEARARITH(⊤), QF_LINEARINTARITH(24,inc,uc), QF_LINEARREALARITH(;,‖,24,uc)

Winner of the Logics (where it did not win the corresponding division)

QF_ALIA(⊥), QF_AUFBV(24), QF_AUFBVLIA(inc), QF_AUFLIA(;,‖,⊤,⊥), QF_LIRA(;,‖,⊤,⊥,mv), QF_RDL(⊤,⊥,mv), QF_UFBVLIA(inc), QF_UFLIA(⊥,inc), QF_UFLRA(;,‖,⊤,⊥,inc,mv), QF_UFNRA(;,‖,⊥,24,inc,mv)

YicesQS

Winner of the Division

$\textsc{Arith}(\top,24)$

Winner of the Logics (where it did not win the corresponding division)

$\mathrm{LRA}(;,\|,\bot)$, $\mathrm{NRA}(;,\|,\bot)$

Z3-Z3++

Overall Winner
BIGGEST LEAD(mv), LARGEST CONTRIBUTION(mv)

Winner of the Divisions
QF_LINEARINTARITH(;,$\|$,$\top$,mv), QF_NONLINEARINTARITH(;,$\|$,$\top$,$\bot$,24,mv),
QF_NONLINEARREALARITH(;,$\|$,$\top$,mv)

Winner of the Logic (where it did not win the corresponding division)
QF_IDL($\bot$)

z3-alpha

Winner of the Division
QF_STRINGS(24)

Winner of the Logics (where it did not win the corresponding division)
QF_S($\top$), QF_SNIA(;,$\|$,$\top$)

Checking Disagreements

- 50411 benchmarks of 227 938 have no status
- 300 benchmarks with disagreements (ABV, BV, LIA, NIA, QF_ABV, QF_AUFBV, QF_BVFP, QF_FP, QF_NRA, QF_SLIA, UF, UFDT, UFDTLIRA)
- We manually resolved the disagreements: authors confirmed solver unsoundness
- We had 10 solvers with soundness issues:

- Bitwuzla (uc printing error)
- iProver (& Fixed)
- OSTRICH
- Q3B
- UltimateEliminator+MathSAT
- UltimateIntBlastingWrapper+SMTInterpol
- Vampire
- Yices2 (& Fixed)
- Z3-Noodler (& Fixed)
- Z3-Owl (& Fixed)

Plans for SMT-COMP 2023

- "Revelations are found in clouds" — Serge King
- MV: Algebraic number $\implies$ non experimental

  ```
  (root-of-with-ordering (coeffs p_0 p_1 ... p_n) i)

  (root-of-with-interval  (coeffs p_0 p_1 ... p_n) min max)
  (root-of-with-enclosure (coeffs p_0 p_1 ... p_n) min max)
  ```

  Thanks for the solvers that implemented one of them!
- MV: partial function more discussion needed
- MV: array theory more discussion needed

Plans for SMT-COMP 2023?

- Proof *validation* track
- Hopefully proof exhibition this year will help
- We have to analyze the data still
  - Job only finished last week (took 18 days to run)
  - 830gb

SMT-COMP organizing committee

Three people organize the SMT-COMP. In 2023:

- Martin Bromberger
- Jochen Hoenicke
- François Bobot

Jochen have been organizer for four-years! He can rest happy.

We need a successor for next year's competition. Contact us if you would like to volunteer!

Acknowledgements

- Andrea Micheli: pysmt
- Guillaume Bury, FB.: Model Validator
- Clark Barrett, Pascal Fontaine, Aina Niemetz, Mathias Preiner, Hans-Jörg Schurr: SMT-LIB benchmarks
- Aaron Stump: StarExec support
- Mike Whalen and team: Cloud/Parallel Track

starexec

powered by aws

Benchmark contributors

In 2023 new benchmarks were contributed by:

- Alex Coffin
- Alex Ozdemir
- Ali Uncu, James Davenport and Matthew England
- Bohan Li
- Elizabeth Polgreen
- Fuqi Jia
- Johann-Tobias Aaron and Raphael Schäg
- Matthew England and Miguel Del Rio Almajano
- Nicolas Amat
- Yannick Moy
- Yoni Zohar

Thanks

to all participants

Thanks

to all participants

and to you for listening