SMT-COMP

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

• first held in 2005
• 2013: evaluation instead of competition
• since 2014: hosted by StarExec

Goals

○ encourage scientific advances in SMT solvers
○ stimulate community to explore shared challenges
○ promote tools and their usage
○ engage and include new members of the community
○ support the SMT-LIB project to promote and develop the SMT-LIB format and collect relevant benchmarks
SMT solver: determine (un)satisfiability of benchmarks from SMT-LIB

- **SMT Solvers** in the ‘classical’ sense
- **Wrapper Tools**: call one or more other SMT solvers
- **Derived Tools**: based on and extends another SMT solver
- **Automated Theorem Provers** (e.g., Vampire)

→ **New** system description mandatory
→ **New** naming convention for derived tools
Tracks

- **Single Query Track** (previously: Main Track)
  - one **single** check-sat command, no push/pop commands
  - **New** remove benchmarks solved by all solvers in 2018 in $\leq 1s$
  - **New** selection of benchmarks
  - **New** time limit: 2400s (40 min)

- **Incremental Track** (previously: Application Track)
  - **multiple** check-sat and push/pop commands
  - solvers are executed on benchmarks via **trace executor**
  - **New** selection of benchmarks
  - **New** keep benchmarks with first check-sat status unknown
  - **New** execute solver **beyond** first status unknown check-sat call
  - **time limit**: 2400s (40 min)
• **Unsat Core Track**
  - one *single* check-sat command, *multiple* assert commands
  - benchmarks with *status* unsat
  - extract *unsat core* as set of top-level assertions
  - New *remove* benchmarks with a single assert command
  - New *selection* of benchmarks
  - *time limit*: 2400s (40 min)
Tracks

- **New: Challenge Track**
  - two subtracks: non-incremental and incremental
  - benchmarks that were *nominated* by their submitters for this track
  - *time limit*: 43200s (12 hours)

- **New: Model Validation Track** *(experimental)*
  - one *single* check-sat command,
  - *selection* of benchmarks with *status* sat
  - produce full, correct, well-formed *model* in SMT-LIB format
  - *only* for division QF_BV
  - *time limit*: 2400s (40 min)
→ **Tracks** are split into divisions

→ **Divisions** correspond to logics in SMT-LIB

- solvers are submitted to divisions in a track
- **winners** are declared
  - per division and track
  - with respect to different scoring schemes per track

- **New** do not run non-competitive divisions
Benchmark Selection

- **2015-2018**: all eligible benchmarks in a division
  - results more predictable
  - more of an evaluation than a competition
  - **Main Track (2018)**:
    - 78% solved by all participating solvers
    - 71% solved in $\leq 1$ s
    - in 7 out of 46 divisions $> 99\%$ solved by all solvers

- **New** alternative benchmark selection
  - **remove** easy/uninteresting benchmarks
    - SQ: all benchmarks solved by all solvers in $\leq 1$s in 2018
    - UC: all benchmarks with only a single assertion
  - **cap** number of instances in a division
    - $n \leq 300$: all instances
    - $300 < n \leq 600$: 300 instances
    - $n > 600$: 50% of the logic
  - guarantee inclusion of **new** benchmarks (at least one per family)
  - select benchmarks randomly using a uniform distribution
Single Query and Unsat Core Track Scoring

- **2016-2018**: weighted with respect to benchmark family size
  - → **goal**: de-emphasize large benchmark families
  - → fairly complicated, not necessarily intuitive
  - → complicates comparing paper and competition results

- **Competition report** for 2015-2018 (under review):
  - → families **no significant impact** on the (weighted) scores
    - ○ problems with scoring script (2016-2018)
    - ○ incorrect interpretation of benchmark family
    - ○ **after fix**: only one change (2017 AUFNIRA: CVC4 over Vampire)
  - → **unweighted**: only 7 out of 139 winners in 2016-2018 change

- **New** drop weighted scoring, use **unweighted** scheme from 2015
Scores

- **Single Query, Challenge (non-incremental):**
  number of correctly solved instances

- **Incremental, Challenge (incremental):**
  number of correctly solved check-sat calls

- **Unsat Core:**
  reduction in terms of top-level assertions

- **Model Validation:**
  number of correctly solved instances with validated models
Scores

- **sequential score** (SQ, CHSQ, UC, MV)
  time limit applied to CPU time

- **parallel score** (all)
  time limit applied to wall-clock time

- **New sat score** (SQ, CHSQ)
  parallel score for satisfiable instances

- **New unsat score** (SQ, CHSQ)
  parallel score for unsatisfiable instances

- **New 24s score** (SQ, CHSQ)
  parallel score for time limit of 24s
Competition-Wide Recognitions

- **2014-2018:**
  - competition-wide scores as weighted sum of division scores
  - emphasis on number of entered divisions

- **New** replace with **two new competition-wide rankings**
  - focus on measures that make sense to compare between divisions
  - for all scores in a track

- **biggest lead**
  - in terms of score over the solver in the second place
  - tie: ranked by biggest lead in CPU/wall-clock time

- **largest contribution**
  - ranked by contribution to virtual best solver in terms of score
  - tie: ranked by largest contribution in terms of CPU/wall-clock time
# Competition Overview

<table>
<thead>
<tr>
<th>Track</th>
<th>Solvers</th>
<th>Divisions</th>
<th>Benchmarks</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total</td>
<td>C/NC</td>
<td>Total</td>
</tr>
<tr>
<td>SQ</td>
<td>51 (+27)</td>
<td>37/14</td>
<td>57 (+7)</td>
</tr>
<tr>
<td>Inc</td>
<td>22 (+16)</td>
<td>14/8</td>
<td>29 (+8)</td>
</tr>
<tr>
<td>CHSQ</td>
<td>21 (+21)</td>
<td>15/6</td>
<td>3 (+3)</td>
</tr>
<tr>
<td>CHInc</td>
<td>12 (+12)</td>
<td>7/5</td>
<td>3 (+3)</td>
</tr>
<tr>
<td>UC</td>
<td>14 (+9)</td>
<td>8/6</td>
<td>38 (-6)</td>
</tr>
<tr>
<td>MV</td>
<td>10 (+10)</td>
<td>10/0</td>
<td>1 (+1)</td>
</tr>
</tbody>
</table>

C ... Competitive  NC ... Non-Competitive  Exp ... Experimental

**Teams:** 23 (+6)

**StarExec Stats:** 21.4 years CPU time; 1,022,802 job pairs
Non-Competitive Solvers

Total: 14 (SQ), 8 (Inc), 6 (CHSQ), 5 (CHINC), 6 (UC)

- submitted by organizers
  - Z3 4.8.4
  - best solvers 2018 (SQ: 9, Inc: 5, CHSQ: 3, CHINC: 3, UC: 5)

- submitted by participants
  - 2 derived tools (Boolector-ReasonLS, CVC4-SymBreak)
  - 3 fixed solver versions (1 x CVC4, 2 x STP)
Solver Presentations

Boolector, COLIBRI, CVC4, MathSAT, OpenSMT, SPASS-SATT, Vampire, VeriT Yices
Boolector at the SMT-COMP’19
Aina Niemetz, Mathias Preiner, Armin Biere

Tracks/Divisions

Single Query: BV, QF_ABV, QF_AUFBV, QF_BV, QF_UFBV
Incremental: QF_ABV, QF_AUFBV, QF_BV, QF_UFBV
Challenge: QF_ABV, QF_AUFBV, QF_BV
Model Validation: QF_BV

Improvements

- Incremental improvements to avoid redundant clauses in SAT solver
- SAT race 2019 version of CaDiCaL for all logics and tracks
  ▶ now default SAT engine for incremental and non-incremental
- GMP for faster BV implementation (improving LS engines)
- CryptoMiniSat support

Configurations

- Boolector: Combination of prop.-based local search + bit-blasting
  ▶ Local search for QF_BV and BV
- Poolector: Portfolio of four parallel (non-incremental) Boolector configurations:
  ▶ CaDiCaL, Lingeling, CryptoMiniSat, and SLS (for QF_BV)

https://boolector.github.io
QF_FP: Since last year small bug fix and improvements

- Forgot to participate to QF_FPLRA
- Focused on 25s
Divisions
This year’s configuration of CVC4 enters all divisions in all tracks.

New Features/Improvements

- Eager bit-blasting solver:
  - New version of CaDiCaL with support for incremental solving
  - Support for incremental eager bit-blasting with CaDiCaL as backend (QF_BV)
  - Not using ABC anymore
  - Fewer consistency lemmas in Ackermannization preprocessing pass

- String solver: better heuristics, more aggressive rewriting, more efficient reductions of extended operators

- Floating-point solver: new version of SymFPU (primarily bug fixes)

Configurations

- Industry Challenge Track and Model-Validation Track: Same configurations as Single Query Track

- Unsat-Core Track: Fixed last year’s configuration that had errors on QF_UFBV
OpenSMT

A relatively small DPLL(T)-based SMT Solver
Developed at University of Lugano, Switzerland
Supports QF_UF, QF_LRA, and to some extent QF_BV

Lookahead-Based SMT
Theory refinement
Interpolation (esp. in LRA)
Integration to model checkers HiFrog and Sally

2018-2019: Performance improvements, better defined development process

Available from http://verify.inf.usi.ch/opensmt
Developers:
Martin Bromberger, Mathias Fleury, Simon Schwarz, Christoph Weidenbach

Ground Linear Arithmetic Solver:
- newest tool in the SPASS Workbench
- combines our theory solver SPASS-IQ and our unnamed SAT solver
- supports QF_LIA, QF_LRA, (and QF_LIRA)
- complete but efficient theory solver [IJCAR2018]
- uses fast cube tests [IJCAR2016, FMSD2017]
- SAT decisions based on theory solver information
- uses many more well-known techniques for linear arithmetic
Developers:
Martin Bromberger, Mathias Fleury, Simon Schwarz, Christoph Weidenbach

Ground Linear Arithmetic Solver:
• newest tool in the SPASS Workbench
• combines our theory solver SPASS-IQ and our unnamed SAT solver
• supports QF_LIA, QF_LRA, (and QF_LIRA)
• complete but efficient theory solver [IJCAR2018]
• uses fast cube tests [IJCAR2016, FMSD2017]
• SAT decisions based on theory solver information
• uses many more well-known techniques for linear arithmetic

http://www.spass-prover.org/spass-satt
Vampire 4.4-SMT

Giles Reger\textsuperscript{1}, Martin Suda\textsuperscript{2}, Andrei Voronkov\textsuperscript{15}, Evgeny Kotelnikov\textsuperscript{3}, Simon Robillard\textsuperscript{3}, Laura Kovács\textsuperscript{4}, and Martin Riener\textsuperscript{1}

SMT Comp 2019
July 8, Lisbon, Portugal

\textsuperscript{1}University of Manchester, Manchester, UK

\textsuperscript{2}Czech Technical University in Prague, Czech Republic

\textsuperscript{3}Chalmers University of Technology, Gothenburg, Sweden

\textsuperscript{4}Institute for Information Systems, Vienna University of Technology, Austria

\textsuperscript{5}Easychair
Features

• Superposition based First Order Resolution Prover
• Finite Model Finding
• Inst-gen
• Redundancy elimination
• Splitting via AVATAR
• Sine axiom selection
• Induction
• CASC since 1999
SMT Related Features

- SMT Logics: A, DT, LIA, LRA, NIA, NRA, UF
- Single Queries
- SMT since 2016
- Theory axioms
- AVATAR modulo theories (ground splitting via Z3)
- Unification with abstraction
- Theory instantiation
Available online

https://vprover.github.io

https://github.com/vprover/vampire
What is new (not yet in the SMT-COMP version):

- cleaning, efficiency improvements
- $\lambda$-free Higher-order
- improved quantifier handling (ML, instantiation, superposition)
- better proofs

Goals:

- clean, small SMT for UF(N|L)IRA with quantifiers and proofs
- for verification platforms B, TLA+
Yices 2 in SMTCOMP 2019

Yices 2

- Supports linear and non-linear arithmetic, arrays, UF, bitvectors
- Supports incremental solving and unsat cores
- Includes two types of solvers: classic CDCL(T) + MC-SAT
- [https://github.com/SRI-CSL/yices2](https://github.com/SRI-CSL/yices2)
- [https://yices.csl.sri.com](https://yices.csl.sri.com)

New in 2019

- Models in SMT-LIB2 format
- Improved bitblasting-based solver
- MC-SAT for bitvectors
- Thread-safe
Bitblasting-Based Solver

Bitblasting in Yices 2
- implemented in 2009 + extended with many simplifications and rewriting rules
- uses a relatively simple CDCL solver (no preprocessing, simple heuristics)
- incremental

New developments
- support for third-party SAT-solvers (as long as provide the right API)
- currently supported:
  - CaDiCal (Armin Biere)
  - CryptoMiniSAT (Mate Soos)
- We also have developed a new, more performant CDCL-based SAT solver to replace the default
MC-SAT for Bitvectors

MC-SAT
- alternative to CDCL(T)
- in Yices: used primarily for non-linear arithmetic (+ UF)

New developments
- extended MC-SAT to QF_BV: our goal is to support word-level reasoning
  - BDDs for representing sets of values
  - specialized reasoning components for two QF_BV fragments:
    - concatenation + extraction + equalities
    - (simple) linear-arithmetic
  - unsat cores + bit-blasting outside these fragments
- still work in progress, very fast on some examples
MathSAT5 (Nonlinear) at the SMT Competition 2019

Ahmed Irfan\textsuperscript{1}, Alessandro Cimatti\textsuperscript{2}, Alberto Griggio\textsuperscript{2}, Roberto Sebastiani\textsuperscript{3}

\textsuperscript{1} Stanford University, USA
\textsuperscript{2} Fondazione Bruno Kessler, Italy
\textsuperscript{3} University of Trento, Italy

– SMT Competition 2019, Lisbon, Portugal –
MathSAT5 (Nonlinear)

MathSAT5, a DPLL(T) solver

- supports most SMT-LIB theories + functionalities (e.g. unsat cores, interpolation, ALLSMT)
- supports nonlinear arithmetic on reals & integers + transcendental functions (sin(), exp())
  - based on incremental linearization: abstraction/refinement to SMT(QF_UFLA)
  - multiplication, sin() and exp() modeled by uninterpreted functions
  - incrementally axiomatized on demand by linear constraints

Participation and Configurations

- Categories:
  - Single query track: QF_ANIA, QF_AUFNIA, QF_NIA, QF_NIRA, QF_NRA, QF_UFNIA, QF_UFNRA.
  - Incremental track: QF_ANIA, QF_AUFBVNIA, QF_NIA, QF_UFNIA.
  - Unsat Core track: QF_ANIA, QF_AUFNIA, QF_NIA, QF_NIRA, QF_NRA, QF_UFNIA, QF_UFNRA.

- Submitted versions:
  - MathSAT default: public release version 5.5.4 + minor fixes, ≈ as described in our SAT’18 paper
  - MathSAT-na-ext: MathSAT default
    - use of lazier strategy for the instantiation of linearization lemmas;
    - try to minimize the Boolean assignment that are given to theory solvers;
    - use bi-implication tangent lemmas:
    - linearization lemmas learnt only temporarily
SMT-COMP 2019 Results

Competition-Wide Recognitions

Largest Contribution Ranking
- Challenge Track (incremental)
- Challenge Track (non-incremental)
- Incremental Track
- Model Validation Track (experimental)
- Single Query Track
- Unsat Core Track

Biggest Lead Ranking
- Challenge Track (incremental)
- Challenge Track (non-incremental)
- Incremental Track
- Model Validation Track (experimental)
- Single Query Track
- Unsat Core Track

Tracks Summary
- Challenge Track (incremental)
- Challenge Track (non-Incremental)
- Incremental Track
- Model Validation Track (experimental)
- Single Query Track
- Unsat Core Track

Divisions
- ABVFP
  - Incremental Track
  - Single Query Track
Competition-Wide Recognitions

Trophies
## Trophies: Largest Contribution

<table>
<thead>
<tr>
<th>Single Query</th>
<th>1st Place</th>
<th>2nd Place</th>
</tr>
</thead>
<tbody>
<tr>
<td>seq</td>
<td>CVC4 (QF_NIA)</td>
<td>Vampire (UF)</td>
</tr>
<tr>
<td>par</td>
<td>CVC4 (QF_NIA)</td>
<td>Vampire (UF)</td>
</tr>
<tr>
<td>sat</td>
<td>Par4 (AUFLIRA)</td>
<td>SMTInterpol (UFLIA)</td>
</tr>
<tr>
<td>unsat</td>
<td>Par4 (UFNIA)</td>
<td>Vampire (UF)</td>
</tr>
<tr>
<td>24s</td>
<td>Vampire (UF)</td>
<td>Par4 (UFNIA)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Incremental</th>
<th>1st Place</th>
<th>2nd Place</th>
</tr>
</thead>
<tbody>
<tr>
<td>par</td>
<td>CVC4 (UFLRA)</td>
<td>Boolector (QF_BV)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Unsat Core</th>
<th>1st Place</th>
<th>2nd Place</th>
</tr>
</thead>
<tbody>
<tr>
<td>seq</td>
<td>CVC4 (AUFLIRA)</td>
<td>MathSAT (QF_NIA)</td>
</tr>
<tr>
<td>par</td>
<td>CVC4 (AUFLIRA)</td>
<td>MathSAT (QF_NIA)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Challenge</th>
<th>1st Place</th>
<th>2nd Place</th>
</tr>
</thead>
<tbody>
<tr>
<td>par</td>
<td>Yices (QF_AUFBV)</td>
<td>Boolector (QF_ABV)</td>
</tr>
</tbody>
</table>
## Trophies: Biggest Lead

<table>
<thead>
<tr>
<th>Single Query</th>
<th>1&lt;sup&gt;st&lt;/sup&gt; Place</th>
<th>2&lt;sup&gt;nd&lt;/sup&gt; Place</th>
</tr>
</thead>
<tbody>
<tr>
<td>seq</td>
<td>CVC4 (FP)</td>
<td>Par4 (UFBV)</td>
</tr>
<tr>
<td>par</td>
<td>CVC4 (FP)</td>
<td>Par4 (UFBV)</td>
</tr>
<tr>
<td>sat</td>
<td>CVC4 (AUFDTLIA)</td>
<td>Par4 (AUFLIRA)</td>
</tr>
<tr>
<td>unsat</td>
<td>CVC4 (BVFP)</td>
<td>SMT-RAT (QF_NIRA)</td>
</tr>
<tr>
<td>24s</td>
<td>CVC4 (BVFP)</td>
<td>Par4 (UFBV)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Incremental</th>
<th>1&lt;sup&gt;st&lt;/sup&gt; Place</th>
<th>2&lt;sup&gt;nd&lt;/sup&gt; Place</th>
</tr>
</thead>
<tbody>
<tr>
<td>par</td>
<td>CVC4 (ANIA)</td>
<td>Yices (QF_AUFBV)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Unsat Core</th>
<th>1&lt;sup&gt;st&lt;/sup&gt; Place</th>
<th>2&lt;sup&gt;nd&lt;/sup&gt; Place</th>
</tr>
</thead>
<tbody>
<tr>
<td>seq</td>
<td>CVC4 (UFLIA)</td>
<td>Yices (QF_AX)</td>
</tr>
<tr>
<td>par</td>
<td>CVC4 (UFLIA)</td>
<td>Yices (QF_AX)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Challenge</th>
<th>1&lt;sup&gt;st&lt;/sup&gt; Place</th>
<th>2&lt;sup&gt;nd&lt;/sup&gt; Place</th>
</tr>
</thead>
<tbody>
<tr>
<td>par</td>
<td>Yices (QF_AUFBV)</td>
<td>Boolector (QF_ABV)</td>
</tr>
</tbody>
</table>
Discussion

- **time limit**
  - increased back to 2400s (from 1200s 2017-2018) in SQ track
  - only $-3953$ instances if cut off at **1200s** (sequential score)
  - $\sim 50\%$ of the timeouts in **quantified** divisions

  $\rightarrow$ run selected challenging benchmarks in the **challenge track**

  $\rightarrow$ **decrease** time limit (maybe even further) for **other tracks**

  $\rightarrow$ shorter time limit for **quantified divisions**?
  (typically: solved within short time or “never”)
Discussion

- **divisions**
  - size of competitions is getting out of hand
  - this year we didn’t run non-competitive divisions
    
    → **don’t run** if less than 3? 4? competitive participants?

- **parallel score**
  - StarExec only offers 4 cores per job
  - not interesting for real parallelism
    
    → **future plans**: dedicated **parallel track**
    
    → would require to move away from StarExec
Discussion

• **portfolio wrapper tools**
  - wrapper tools allowed to participate without restrictions
  - problems with portfolio (not author of the wrapped solvers)
    - win with simple script and work of other teams
    - negative/unfair impact on competition-wide rankings
    - progress of non-portfolio tools harder to distinguish
  - disallowing wrapper tools entirely is problematic (example: Vampire)

  → **disallow** portfolio with wrapped solvers from other teams?
  → only allow non-competitive submission?
  → at least **exclude** them from competition-wide recognitions
  → **similar issues** with SATzilla-style systems
• Mathias Preiner (benchmark selection and scoring scripts)
• Aaron Stump (StarExec)
• Andres Nötzli (trace executor extension)
• Marco Gario and Andrea Micheli (PySMT)
• Martin Riener (certificates/trophies logistics)