12th International Satisfiability Modulo Theories Competition
SMT-COMP 2017

Matthias Heizmann (co-organizer)
Giles Reger (co-organizer)
Tjark Weber (chair)
Outline

▶ Main changes over last competition
  ▶ Benchmarks with 'unknown' status
  ▶ Logics with algebraic data-types
    AUFBVDTLIA, AUFDTLIA, QF_DT, UFDT, UFDTLIA
  ▶ Unsat-core Track

▶ Statistics and selected results of competition

▶ Short presentation of solvers
  Boolector, COLIBRI, CVC4, SMTInterpol, veriT, Yices
SMT-COMP – Procedure

SMT-LIB users → submit benchmarks → SMT-LIB benchmarks curated by Clark Barrett, Pascal Fontaine, Cesare Tinelli, Christoph Wintersteiger

SMT solver developers → upload solvers → StarExec maintained by Aaron Stump

StarExec → upload benchmarks → Competition results
Solvers, Logics, and Benchmarks

- 15 teams participated

**Solvers:**

- Main track: 19
- Application track: 4
- Unsat-core track: 2

**Logics:**

- Main track: 40
- Application track: 14
- Unsat-core track: 39

**Benchmarks:**

- Main track: 256973
- Application track: 5971
- Unsat-core track: 114233
Cluster of machines at the University of Iowa.

Hardware:
- Intel Xeon CPU E5-2609 @ 2.4 GHz, 10 MB cache
- 2 processors per node, 4 cores per processor
- Main memory capped at 60 GB per job pair

Software:
- Red Hat Enterprise Linux Server release 7.2
- Kernel 3.10.0-514, gcc 4.8.5, glibc 2.17
Main Track

Main Track benchmark

(set-logic ...)
(set-info ...)
.
.
(declare-sort ...)
(define-sort ...)
(declare-fun ...)
(define-fun ...)
(assert term0)
(assert term1)
(assert term2)
.
.
(check-sat)
(exit)

any number of
set-info, declare-sort, define-sort,
declare-fun, define-fun, assert
in any order

← one single check-sat command
Some benchmarks in SMT-LIB repository do not have a sat/unsat status.

Benchmarks with ‘unknown’ status in SMT-COMP

- **2015**: not used in competition
- **2016**: separate experimental track
- **2017**: included in Main Track
New logics

Algebraic data-types

- defined in SMT-LIB 2.6 draft
- “experimental” this year (i.e., no winner determined)

<table>
<thead>
<tr>
<th>benchmarks</th>
<th>solvers</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUFBVDDTLIA</td>
<td>1709, CVC4</td>
</tr>
<tr>
<td>AUFDTLIA</td>
<td>728, CVC4, vampire</td>
</tr>
<tr>
<td>QF_DT</td>
<td>8000, CVC4</td>
</tr>
<tr>
<td>UFDT</td>
<td>4535, CVC4, vampire</td>
</tr>
<tr>
<td>UFDTTLIA</td>
<td>303, vampire, CVC4</td>
</tr>
</tbody>
</table>
Benchmarks with 'unknown' status

Rules

- we trust the results of the solver(s)
- in case of disagreement we trust solvers that are sound on benchmarks with known status
- if there is disagreement between otherwise sound solvers, we exclude the benchmark

Outcome

- There were 29 benchmarks with unknown status on which solvers disagreed on the result.
- On one benchmark (in BV) the corresponding solvers were sound on all benchmarks with known status.
- On 28 benchmarks (all in QF FP) the presumably wrong answers were given by unsound solvers.
Benchmarks with 'unknown' status

Rules

- we trust the results of the solver(s)
- in case of disagreement we trust solvers that are sound on benchmarks with known status
- if there is disagreement between otherwise sound solvers, we exclude the benchmark

Outcome

- There were 29 benchmarks with unknown status on which solvers disagreed on the result.
- On one benchmark (in BV) the corresponding solvers were sound on all benchmarks with known status.
- On 28 benchmarks (all in QF_FP) the presumably wrong answers were given by unsound solvers.
Competition run of Main Track

- run all job pairs with 10 min timeout
- made preliminary results available
- rerun all job pairs that timed out with 20 min timeout
- made final results available on Friday (21st June)
Main Track – Selected results – QF_ABV

Benchmarks in this division: 15061

Winners:

<table>
<thead>
<tr>
<th>Sequential Performances</th>
<th>Parallel Performances</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolector</td>
<td>Boolector</td>
</tr>
</tbody>
</table>

Result table

<table>
<thead>
<tr>
<th>Solver</th>
<th>Sequential performance</th>
<th></th>
<th>Parallel performance</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Error Score</td>
<td>Correctly Solved Score</td>
<td>CPU time Score</td>
<td>Errors</td>
</tr>
<tr>
<td>Boolector</td>
<td>0.000</td>
<td>14627.624</td>
<td>56.054</td>
<td>0.000</td>
</tr>
<tr>
<td>CVC4</td>
<td>0.000</td>
<td>13136.950</td>
<td>155.681</td>
<td>0.000</td>
</tr>
<tr>
<td>Yices2</td>
<td>0.000</td>
<td>14515.846</td>
<td>65.395</td>
<td>0.000</td>
</tr>
<tr>
<td>mathsat-5.4.1n</td>
<td>0.000</td>
<td>13098.157</td>
<td>155.650</td>
<td>0.000</td>
</tr>
<tr>
<td>z3-4.5.0n</td>
<td>0.000</td>
<td>13115.900</td>
<td>171.596</td>
<td>0.000</td>
</tr>
</tbody>
</table>

n. Non-competing.

http://smtcomp.sourceforge.net/2017/results-QF_ABV.shtml
Main Track: Competition-Wide Scoring

<table>
<thead>
<tr>
<th>Rank</th>
<th>Solver</th>
<th>Score (sequential)</th>
<th>Score (parallel)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CVC4</td>
<td>161.38</td>
<td>161.76</td>
</tr>
<tr>
<td>2</td>
<td>Yices2</td>
<td>110.63</td>
<td>110.63</td>
</tr>
<tr>
<td>3</td>
<td>SMTInterpol</td>
<td>65.96</td>
<td>66.00</td>
</tr>
<tr>
<td>Logic</td>
<td>Solvers</td>
<td>Benchmarks</td>
<td>Order (parallel performance)</td>
</tr>
<tr>
<td>-----------</td>
<td>---------</td>
<td>------------------------------</td>
<td>---------------------------------------</td>
</tr>
<tr>
<td>ANIA</td>
<td>2</td>
<td>3</td>
<td>CVC4; z3-4.5.0^n</td>
</tr>
<tr>
<td>QF ANIA</td>
<td>2</td>
<td>5</td>
<td>z3-4.5.0^n; CVC4</td>
</tr>
<tr>
<td>QF ALIA</td>
<td>5</td>
<td>44</td>
<td>z3-4.5.0^n; SMTInterpol; Yices2; mathsat-5.4.1^n; CVC4</td>
</tr>
<tr>
<td>QF UFNIA</td>
<td>2</td>
<td>1</td>
<td>z3-4.5.0^n; CVC4</td>
</tr>
<tr>
<td>QF BVFP</td>
<td>1</td>
<td>2</td>
<td>z3-4.5.0^n</td>
</tr>
<tr>
<td>LIA</td>
<td>2</td>
<td>6</td>
<td>z3-4.5.0^n; CVC4</td>
</tr>
<tr>
<td>ALIA</td>
<td>2</td>
<td>24</td>
<td>z3-4.5.0^n; CVC4</td>
</tr>
<tr>
<td>QF UFLRA</td>
<td>5</td>
<td>3056</td>
<td>Yices2; z3-4.5.0^n; SMTInterpol; CVC4; mathsat-5.4.1^n</td>
</tr>
<tr>
<td>UFLRA</td>
<td>2</td>
<td>1870</td>
<td>z3-4.5.0^n; CVC4</td>
</tr>
<tr>
<td>QF UFLIA</td>
<td>5</td>
<td>780</td>
<td>z3-4.5.0^n; CVC4; Yices2; SMTInterpol; mathsat-5.4.1^n</td>
</tr>
<tr>
<td>QF_NIA</td>
<td>2</td>
<td>10</td>
<td>CVC4; z3-4.5.0^n</td>
</tr>
<tr>
<td>QF_FP</td>
<td>1</td>
<td>2</td>
<td>z3-4.5.0^n</td>
</tr>
<tr>
<td>QF BV</td>
<td>4</td>
<td>18</td>
<td>mathsat-5.4.1^n; Yices2; CVC4; z3-4.5.0^n</td>
</tr>
<tr>
<td>QF LRA</td>
<td>6</td>
<td>10</td>
<td>mathsat-5.4.1^n; SMTInterpol; Yices2; z3-4.5.0^n; CVC4; opensmt2</td>
</tr>
<tr>
<td>QF LIA</td>
<td>5</td>
<td>68</td>
<td>Yices2; z3-4.5.0^n; SMTInterpol; mathsat-5.4.1^n; CVC4</td>
</tr>
<tr>
<td>QF AUFLIA</td>
<td>5</td>
<td>72</td>
<td>Yices2; z3-4.5.0^n; SMTInterpol; CVC4; mathsatsat-5.4.1^n</td>
</tr>
</tbody>
</table>
Unsat-core Track

Motivation

▶ Important application of SMT-LIB
▶ One step towards verifiable proofs

History

2012 introduced

2016 reinstated as experimental track

2017 “regular” track
Unsat-core Track

Main Track benchmark

(set-logic ...)
(set-info ...)
.
.
(declare-sort ...)
(define-sort ...)
(declare-fun ...)
(define-fun ...)
(assert term0)
(assert term1)
(assert term2)
.
.
(check-sat)
(exit)

Solver input

(set-option :produce-unsat-cores true)
(set-logic ...)
(set-info ...)
.
.
(declare-sort ...)
(define-sort ...)
(declare-fun ...)
(define-fun ...)
(assert (! term0 :named y0))
(assert (! term1 :named y1))
(assert (! term2 :named y2))
.
.
(check-sat)
(get-unsat-core)
(exit)
**Unsat-core Track**

**Main Track benchmark**

```lisp
(set-logic ...)
(set-info ...)
.
.
(declare-sort ...)
(define-sort ...)
(declare-fun ...)
(define-fun ...)
(assert term0)
(assert term1)
(assert term2)
.
.
(check-sat)
(exit)
```

**Solver input**

```lisp
(set-option :produce-unsat-cores true)
(set-logic ...)
(set-info ...)
.
.
(declare-sort ...)
(define-sort ...)
(declare-fun ...)
(define-fun ...)
(assert (! term0 :named y0))
(assert (! term1 :named y1))
(assert (! term2 :named y2))
.
.
(check-sat)
(get-unsat-core)
(exit)
```

**Solver output**

```
unsat
(y0 y2)
```

*timeout: 40min*
Unsat-core Track

Main Track benchmark

(set-logic ...)
(set-info ...)
.
.
(declare-sort ...)
(define-sort ...)
(define-fun ...
(declare-fun ...
(assert term0)
(assert term1)
(assert term2)
.
.
(check-sat)
(exit)

Solver input

(set-option :produce-unsat-cores true)
(set-logic ...)
(set-info ...)
.
.
(declare-sort ...)
(define-sort ...)
(define-fun ...
(declare-fun ...
(assert (! term0 :named y0))
(assert (! term1 :named y1))
(assert (! term2 :named y2))
.
.
(check-sat)
(get-unsat-core)
(exit)

Validation script

(set-logic ...)
(set-info ...)
.
.
(declare-sort ...)
(define-sort ...)
(define-fun ...
(declare-fun ...
(assert term1)
(declare-fun ...
(assert term2)
(assert term3)
.
.
(check-sat)
(exit)

Solver output

unsat
(y0 y2)

timeout: 40min

Validation
solver 1
sat/

Validation
solver 2
unsat

Validation
solver 3
unsat

Validation
solver 4
unsat

timeout: 5min each

Scoring scheme

n = # assert commands – # size of unsatisfiable core

e = 1 result erroneous
0 otherwise

result erroneous if

⊿ wrong check-sat result or
⊿ unsat-core rejected by validating solvers
Unsat-core Track

Main Track benchmark

(set-logic ...)
(set-info ...)
.
.
(declare-sort ...)
(define-sort ...)
(define-fun ...)
(define-fun ...)
(assert term0)
(assert term1)
(assert term2)
.
.
(check-sat)
(exit)

Solver input

(set-option :produce-unsat-cores true)
(set-logic ...)
(set-info ...)
.
.
(declare-sort ...)
(define-sort ...)
(define-fun ...)
(define-fun ...)
(assert (! term0 :named y0))
(assert (! term1 :named y1))
(assert (! term2 :named y2))
.
.
(check-sat)
(get-unsat-core)
(exit)

Solver output

timeout: 40min
unsat
(y0 y2)

Validation script

(set-logic ...)
(set-info ...)
.
.
(declare-sort ...)
(define-sort ...)
(define-fun ...)
(define-fun ...)
(assert term1)
(assert term2)
.
.
(check-sat)
(exit)

timeout: 5min each

Validation solver 1
sat/
unknown/unsat

Validation solver 2
sat/
unknown/unsat

Validation solver 3
sat/
unknown/unsat

Validation solver 4
sat/
unknown/unsat

Scoring scheme

n = # assert commands – # size of unsatisfiable core
e = \{ 1 result erroneous
0 otherwise \}
result erroneous if
⊿
wrong check-sat result or
⊿
unsat-core rejected by validating solvers
Unsat-core Track

Main Track benchmark

(set-logic ...)
(set-info ...)

(check-sat)

Solver input

(setq-option :produce-unsat-cores true)
(set-logic ...)
(set-info ...)

(check-sat)

Validation script

(set-logic ...)
(set-info ...)

(check-sat)

Scoring scheme

\[ n = \# \text{ assert commands} - \# \text{ size of unsatisfiable core} \]

\[ e = \begin{cases} 
1 & \text{result erroneous} \\
0 & \text{otherwise}
\end{cases} \]

result erroneous if

\( \triangleright \) wrong check-sat result or

\( \triangleright \) unsat-core rejected by validating solvers

Validation

Solver 1

sat/

unknown/unsat

Validator 2

sat/

unknown/unsat

Validator 3

sat/

unknown/unsat

Validator 4

sat/

unknown/unsat
Unsat-core Track – Statistics

245483 job pairs

226501 correct check-sat responses

29 incorrect check-sat responses

18982 timeout/crash/unknown

92% correct check-sat responses

1% incorrect check-sat responses

30 timeout/crash

99.99% correct check-sat responses

0.01% incorrect check-sat responses

226471 get-unsat-core responses

226463 unsatisfiable core validated

99.99% unsatisfiable core validated

0.01% unsatisfiable core rejected by validating solvers

19 times there was no consensus among the validating solvers (→ majority decision)

27525 (∼12%) times no independent validating solver approved the correctness of the unsatisfiable core
Unsat-core Track – Statistics

- 245483 job pairs
  - 92% correct check-sat responses
  - 8% timeout/crash/unknown
  - ~0.01% incorrect check-sat responses

- 226501 correct check-sat responses
  - ~99.99% get-unsat-core responses
  - ~0.01% timeout/crash

- 226471 get-unsat-core responses
  - ~99.99% unsatisfiable core validated
  - ~0.01% unsatisfiable core rejected by validating solvers

- 226463 unsatisfiable core validated
- 8 unsatisfiable core rejected by validating solvers

- 18982 timeout/crash/unknown

- 30 timeout/crash

- 29 incorrect check-sat responses

- 19 times there was no consensus among the validating solvers (⇝ majority decision)

- 27525 (~12%) times no independent validating solver approved the correctness of the unsatisfiable core
(Very) short presentations of

Solvers

that sent us slides.

Boolector, COLIBRI, CVC4, SMTInterpol, veriT, Yices
Boolector at the SMTCOMP’17
Aina Niemetz, Mathias Preiner, Armin Biere

Divisions

BV  QF_BV  QF_UFBV  QF_ABV  QF_AUFBV

Changes since 2016 (QF_BV)

- combination of prop.-based local search + bit-blasting now default
- experimental configuration with new SAT solver CaDiCaL

Major Improvements

- BV engine at SMT-COMP 2016 was a prototype implementation
  \[\rightarrow\text{ rewrite}\] of BV engine with major improvements
COLIBRI \((\text{QF\_FP, QF\_BVFP})\)

- 776 errors due to a wrong backward propagation for \(\text{fp\_fma}\)
- 1 error due to a mix between \(-0\). and \(+0\).

On the 308, non Wintersteiger benchmark of \(\text{QF\_FP}\) in sec.
CVC4 1.5

Clark Barrett (Stanford), Martin Brain (Oxford), Guy Katz (Stanford), Tim King (Google), Paul Meng (U Iowa), Aina Niemetz (Stanford), Mathias Preiner (Stanford), Andres Nötzli (Stanford), Andrew Reynolds (U Iowa), Cesare Tinelli (U Iowa)

SMT 2017, July 22, 2017
CVC4 1.5: Recent Developments

- A new theory of sets with cardinality and relations.
- A new theory of strings.
- A new theory of separation logic constraints.
- Support for many new heuristics for reasoning with quantifiers, including finite model finding.
- Improved heuristics for reasoning about non-linear arithmetic.
- Support for proofs for uninterpreted functions, arrays, bitvectors, and their combinations.
- Support for unsat cores.
- Native support for syntax-guided synthesis (sygus).
We aim for CVC4 to be a versatile research platform for SMT and are open to collaborators and contributors.

For more information:

- Contact one of the project leaders:
  - Clark Barrett `barrett@cs.stanford.edu`
  - Cesare Tinelli `cesare-tinelli@uiowa.edu`
- Visit the website: `cvc4.stanford.edu`
Quantifier Free Linear Arithmetic

\[ y \leq i + 1 \]
\[ i \leq y \]

Quantifier Free Arrays

\[ b = a\langle i \triangleleft v \rangle \]
\[ a[v] = v \]

Quantifier Free Uninterpreted Functions

\[ f(b) = v \]
\[ f(a) \neq v \]

SMTInterpol

decides Satisfiability Modulo Theory
computes Craig Interpolants

http://ultimate.informatik.uni-freiburg.de/smtinterpol
Quantifier Free Linear Arithmetic

\[ y \leq i + 1 \]
\[ i \leq y \]

Quantifier Free Arrays

\[ b = a \langle i \triangleleft v \rangle \]
\[ a[v] = v \]

Quantifier Free Uninterpreted Functions

\[ f(b) = v \]
\[ f(a) \neq v \]

SMTInterpol

decides Satisfiability Modulo Theory
computes Craig Interpolants

http://ultimate.informatik.uni-freiburg.de/smtinterpol
Quantifier Free Linear Arithmetic

\[ y \leq i + 1 \]
\[ i \leq y \]
\[ y - \text{to}_\text{int}(y) < 3 \]

Quantifier Free Arrays

\[ b = a\langle i \bowtie v \rangle \]
\[ a[v] = v \]

Quantifier Free Uninterpreted Functions

\[ f(b) = v \]
\[ f(a) \neq v \]

SMTInterpol decides Satisfiability Modulo Theory
computes Craig Interpolants

http://ultimate.informatik.uni-freiburg.de/smtinterpol
Quantifier Free
Linear Arithmetic
\[ y \leq i + 1 \]
\[ i \leq y \]
\[ y - \text{to}_{-}\text{int}(y) < 0.3 \]

Uninterpreted Functions
\[ f(b) = v \]
\[ f(a) \neq v \]

Arrays
\[ b = a \langle i \triangleleft v \rangle \]
\[ a[v] = v \]
\[ b[i] \geq i \]
\[ f(i + y) = 2v \]

SMTInterpol

decides Satisfiability Modulo Theory
computes Craig Interpolants

http://ultimate.informatik.uni-freiburg.de/smtinterpol
SMTInterpol

decides Satisfiability Modulo Theory
computes Craig Interpolants

http://ultimate.informatik.uni-freiburg.de/smtinterpol
Quantifier Free
Linear Arithmetic

\[ y \leq i + 1 \]

\[ i \leq y \]

\[ y - \text{to\_int}(y) < 0.3 \]

Uninterpreted Functions

\[ f(b) = v \]

\[ f(a) \neq v \]

Arrays

\[ b = a \langle i \triangleleft v \rangle \]

\[ a[v] = v \]

\[ b[i] \geq i \]

\[ f(i + y) = 2v \]

SMTInterpol

decides Satisfiability Modulo Theory
computes Craig Interpolants

http://ultimate.informatik.uni-freiburg.de/smtinterpol
Quantifier Free Linear Arithmetic

$y \leq i + 1$

$i \leq y$

$y - \text{to\_int}(y) < .3$

Uninterpreted Functions

$f(b) = v$

$f(a) \neq v$

Linear Arithmetic

Arrays

$b = a\langle i \triangleleft v \rangle$

$a[v] = v$

$[i] \geq i$

$f(i + y) = 2v$

$f(b) \leq i$

Uninterpreted Functions

$unsat$

decides Satisfiability Modulo Theory

computes Craig Interpolants

http://ultimate.informatik.uni-freiburg.de/smtinterpol
Quantifier Free Linear Arithmetic

\[ y \leq i + 1 \]
\[ i \leq y \]
\[ y - \text{to_int}(y) < 0.3 \]

Uninterpreted Functions

\[ f(b) = v \]
\[ f(a) \neq v \]

Arrays

\[ b = a\langle i \triangleleft v \rangle \]
\[ a[v] = v \]
\[ b[i] \geq i \]
\[ f(i + y) = 2v \]
\[ f(b) \leq i \]

\textbf{SMTInterpol}

decides \textbf{Satisfiability Modulo Theory}
computes Craig \textbf{Interpolants}

http://ultimate.informatik.uni-freiburg.de/smtinterpol
Quantifier Free Linear Arithmetic

\[ y \leq i + 1 \]
\[ i \leq y \]
\[ y - \text{to\_int}(y) < 0.3 \]

Uninterpreted Functions

\[ f(b) = v \]
\[ f(a) \neq v \]

Arrays

\[ b = a \langle i \triangleleft v \rangle \]
\[ a[v] = v \]
\[ b[i] \geq i \]
\[ f(i + y) = 2v \]
\[ f(b) \leq i \]

SMTInterpol

decides Satisfiability Modulo Theory
computes Craig Interpolants

http://ultimate.informatik.uni-freiburg.de/smtinterpol
Quantifier Free Linear Arithmetic

\[ y \leq i + 1 \]

\[ i \leq y \]

\[ y - \text{to\_int}(y) < .3 \]

Uninterpreted Functions

\[ f(b) = v \]

\[ f(a) \neq v \]

Arrays

\[ b = a(i \triangleleft v) \]

\[ a[v] = v \]

\[ b[i] \geq i \]

\[ f(i + y) = 2v \]

\[ f(b) \leq i \]

\[ f(b) = v \]

\[ i \neq v \]

\[ b[i] \geq i \]

\[ f(b) \neq v \]

\[ f(b) \leq i \]

SMTInterpol

interpolants

unsat

proof

unsat core
Quantifier Free Linear Arithmetic

\[ y \leq i + 1 \]

\[ i \leq y \]

\[ y - \text{to\_int}(y) < .3 \]

Quantifier Free Uninterpreted Functions

\[ f(b) = v \]

\[ f(a) \neq v \]

Quantifier Free Arrays

\[ b = a(i \triangleleft v) \]

\[ a[v] = v \]

\[ b[i] \geq i \]

\[ f(i + y) = 2v \]

\[ f(b) \leq i \]

\[ b[i] \geq i \]

\[ f(i + y) = 2v \]

\[ f(b) \leq i \]

SMTInterpol

http://ultimate.informatik.uni-freiburg.de/smtinterpol
Haniel Barbosa, Pascal Fontaine, Maximilian Jaroschek, Marek Kosta, Thomas Sturm, and Vu Xuan Tung

University of Lorraine, CNRS, Inria, and LORIA (France), MPI Informatics and Saarland University (Germany), and JAIST (Japan)

What is new:

- Few improvements for quantifier handling
- Fine-grained proofs for formula processing
- veriT+raSAT+Redlog for supporting QF\_[UF]NRA
- veriT+Redlog for better handling (N|L)RA

Goals:

- Clean, small SMT for UF(N|L)IRA with quantifiers and proofs
- For verification platforms (e.g. B, TLA+) and proof assistants (e.g. Isabelle, Coq)
Yices 2.6

Bruno Dutertre and Dejan Jovanović
SRI International

SMTCOMP 2017
Heidelberg, Germany
Yices 2.6 in SMTCOMP 2017

Status
- No major change since last year
- Supports linear and non-linear arithmetic, arrays, UF, bitvectors
- Limited quantifier reasoning: $\exists\forall$ fragments for bitvector, LRA
- Includes two types of solvers: classic DPPL($T$) + MC-SAT

Entered in all the divisions that Yices supports
- Main track: Quantifier-free logics including linear and nonlinear arithmetic, bitvectors, and combination with UF and Arrays.
- Application track: Same logics, except that the MC-SAT solver is not incremental yet.
What’s New

New Licence
- Yices 2 is now GPL

Distributions
- Prebuilt binaries + source tarfile at http://yices.csl.sri.com
- Git repository on Github https://github.com/SRI-CSL/yices2
- Ubuntu/Debian package
- Homebrew package for MacOS X
What’s Next

MC-SAT Extensions
- Add support for incremental solving
- Extends MC-SAT to bitvector problems

CDCL Solver
- New sat solver in progress

Miscellaneous
- Complete support for SMT-LIB 2.6
- Fix some API issues
Teams:

- Congratulations on your accomplishments!
- Thanks for your participation!