The Reactive Synthesis Competition - SYNTCOMP 2016 and Beyond

Swen Jacobs  Saarland University
Roderick Bloem  TU Graz

18 July 2016 – SYNT Workshop, Toronto
SYNTCOMP: Goals

Make reactive synthesis tools comparable:
- establish **benchmark format**
- collect **benchmark library**
- provide platform for **fair and comprehensive evaluation**

Guide development of reactive synthesis tools:
- encourage implementation of mature, **push-button tools**
- improve state of the art through **challenging benchmarks**
SYNTCOMP: History

**First Call:** 2013 (discussion at SYNT 2013)

**Design Choices:**
- low entry-barrier
- only safety properties, low-level format (AIGER)
- correctness needs to be verified
- hardware model checkers
- output quality is major issue
- ranking based on solution size

**First Competition:** 2014 (FLoC, Vienna Summer of Logic)
- 5 participating groups, >500 benchmarks collected

**Second Competition:** 2015 (CAV, San Francisco)
- essentially same setup, >2000 benchmarks collected, comparison to 2014 tools
AIGER-based Safety Track of SYNTCOMP

- **synthesis problem** defined by AIGER circuit A, with controllable (C) and uncontrollable (U) inputs, and single output error

- **solution** of synthesis problem is an AIG that includes original AIG A, and adds control structure B for inputs C such that resulting system never raises error
Lessons Learned from Previous Competitions

• SYNTCOMP with verified results is (in principle) feasible
• SYNTCOMP is well-received
• tools improved significantly from 2014 to 2015

But:
• verification not always easy
• fair ranking based on solution size not obvious (comparability, fairness, effectiveness)
• pre-existing synthesis tools mostly target LTL or GR(1) synthesis and did not participate
Extensions for SYNTCOMP 2016

AIGER/Safety Track:
• allow witness information (winning region) to help verification of solution

New TLSF/LTL and TLSF/GR(1) Tracks:
• go beyond safety properties
• specifications in TLSF, based on LTL
• solutions in AIGER
• verification of solutions with AIGER model checkers

All tracks: no official ranking wrt. solution sizes
instead: observe effects of different possible rankings
AIGER/Safety Track

1. extension: winning region check
2. new benchmarks
3. participants
4. results
Winning Region Check for Safety Properties

• tools can produce winning region in addition to solution
• if winning region check fails, we fall back to model checking

Experience:
• only 3 cases where both winning region check and full model checking failed (solution size: >20MB, >1M AND-gates, used in 8 configurations)
• tools without winning region: model checking failed 14 times (3 configurations)
• but: we need to fall back to full model checking more often than expected
AIGER/Safety: New Benchmarks

No new benchmarks from community

Observation in 2015:
ltl2dba benchmarks are too easy, HWMCC benchmarks too difficult

New benchmarks generated:
• difficult ltl2dba benchmarks [LTLnfBA, Tian et al. 2016], translated by G. Perez
• more manageable HWMCC benchmarks
AIGER/Safety: Participants 2016

Re-entry/updated:

- **AbsSynthe** (Brenguier, Perez, Raskin, Sankur): BDD-based, compositional, abstraction
- **Demiurge** (Könighofer, Seidl): SAT-based, different cooperating strategies
- **Simple BDD Solver** (Walker, Ryzhyk): BDD-based, abstraction, CUDD 3.0.0

New:

- **SafetySynth** (Tentrup): BDD-based, implements all optimizations from [SYNTCOMP2014] except abstraction
- **SDF** (Khalimov): BDD-based, BDD caching and reordering heuristics
- **TermiteSAT** (Legg, Narodytska, Ryzhyk, (Walker)): SAT-based, portfolio with Simple BDD Solver, hybrid mode that shares information, see also [CAV2016]

Tools that only support realizability check (no synthesis): Simple BDD Solver, TermiteSAT
## AIGER/Safety Results 2016: Realizability

### Sequential execution mode:

<table>
<thead>
<tr>
<th>Rank</th>
<th>Tool (conf)</th>
<th>Solved</th>
<th>Unique</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Simple BDD Solver (w/ Abstraction)</td>
<td>175</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Simple BDD Solver (w/ Abstraction 2)</td>
<td>167</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>SafetySynth</td>
<td>164</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Simple BDD Solver</td>
<td>164</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>SafetySynth (Alt)</td>
<td>163</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>AbsSynthe (S3)</td>
<td>161</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td>AbsSynthe (S2)</td>
<td>151</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>SDF</td>
<td>149</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>AbsSynthe (S1)</td>
<td>147</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>Demiurge D1real</td>
<td>129</td>
<td>6</td>
</tr>
<tr>
<td>11</td>
<td>TermiteSAT</td>
<td>97</td>
<td>4</td>
</tr>
</tbody>
</table>

### Parallel execution mode:

<table>
<thead>
<tr>
<th>Rank</th>
<th>Tool (conf)</th>
<th>Solved</th>
<th>Unique</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AbsSynthe P1</td>
<td>181</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>TermiteSAT Hybrid</td>
<td>180</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>TermiteSAT Portfolio</td>
<td>179</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>Demiurge P3real</td>
<td>156</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>AbsSynthe P3</td>
<td>148</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>AbsSynthe P2</td>
<td>141</td>
<td>0</td>
</tr>
</tbody>
</table>

### Number of Benchmarks: 234

### Not solved: 12

### Not solved: 18
AIGER/Safety Results 2016: Realizability

The graph displays the time (in seconds) taken to solve problems by different tools. The y-axis represents time in seconds, ranging from 0.001 to 1000, while the x-axis represents the number of problems solved, ranging from 0 to 200.

The tools compared include:
- AbsSynthe S3
- Demiurge D1real
- SafetySynth
- Simple BDD Solver w/ Abstraction
- SDF
- TermiteSAT

The data shows the performance of these tools in terms of speed and efficiency as they solve more problems.
AIGER/Safety Results 2016: Realizability

Problems Solved

Time (s)

- AbsSynthe S3
- Demiurge D1real
- SafetySynth
- Simple BDD Solver w/ Abstraction
- SDF
- TermiteSAT
- AbsSynthe P1
- Demiurge P3real
- TermiteSAT Hybrid
A few benchmark classes (not necessarily representative):
### AIGER/Safety Results 2016: Synthesis

**Number of benchmarks:** 215

**Sequential execution mode:**

<table>
<thead>
<tr>
<th>Rank</th>
<th>Tool (conf)</th>
<th>Solved</th>
<th>Unique</th>
<th>MC timeout</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SafetySynth</td>
<td>153</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>SafetySynth Alt</td>
<td>152</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>AbsSynthe S3</td>
<td>151</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>AbsSynthe S2</td>
<td>140</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>AbsSynthe S1</td>
<td>137</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>SDF</td>
<td>134</td>
<td>0</td>
<td>11</td>
</tr>
<tr>
<td>7</td>
<td>Demiurge D1Synt</td>
<td>118</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

**Parallel execution mode:**

<table>
<thead>
<tr>
<th>Rank</th>
<th>Tool (conf)</th>
<th>Solved</th>
<th>Unique</th>
<th>MC timeout</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AbsSynthe PS1</td>
<td>165</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>Demiurge P3Synt</td>
<td>150</td>
<td>9</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>AbsSynthe PS3</td>
<td>140</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>AbsSynthe PS2</td>
<td>134</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

**not solved:** 23

**not solved:** 14
TLSF/LTL Track

1. synthesis problem for TLSF benchmarks
2. model checking for TLSF benchmarks
3. benchmark collection in TLSF
4. participants
5. results
TLSF-based LTL Track of SYNTCOMP

• synthesis problem defined by TLSF specification F, with inputs (U) and outputs (C) of the system to be synthesized

• solution of synthesis problem is an AIG with inputs U and outputs C such that the LTL formula represented by F is satisfied (on all possible runs)
Model Checking Solutions for TLSF/LTL

We still want solutions to be verified

Approach:
1. translate TLSF spec to AIGER monitor A (via SMV)
2. combine monitor with solution B
3. use existing AIGER model checkers

Experience:
• works reasonably well for LTL synthesis tools (20 timeouts for Acacia, no timeouts for bounded synthesis tools)
Benchmark Collection in TLSF

Translated existing benchmark suites to TLSF:

- Lily and Acacia benchmark suites

New, parameterized encodings of existing benchmarks:

- AMBA bus controller
- generalized buffer
- load balancer
- arbiters of different complexity
- different ltl2dba problems (translation of LTL formulas to Büchi automata)
TLSF/LTL: Participants 2016

- **Acacia4Aiger** (Brenguier, Perez, Raskin, Sankur): antichains, compositionality, adapted to SYNTCOMP requirements
- **BoSy** (Tentrup): bounded synthesis, SAT/QBF-solving
- **Party-Elli** (Khalimov): bounded synthesis, SMT-solving, adapted to SYNTCOMP requirements

All tools support realizability checking and synthesis

Additional legacy tool (hors concours):
**Unbeast** (Ehlers): bounded synthesis, BDD-based, not optimized for competition

For this new track, a significant amount of work went into making synthesis tools and verification/evaluation setup work nicely together
**TLSF/LTL Results 2016: Realizability**

Number of benchmarks: 195

**Sequential Execution Mode:**

<table>
<thead>
<tr>
<th>Rank</th>
<th>Tool (conf)</th>
<th>Solved</th>
<th>Unique</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Acacia4Aiger</td>
<td>153</td>
<td>32</td>
</tr>
<tr>
<td>2</td>
<td>Bosy (exp)</td>
<td>147</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>Bosy (lin)</td>
<td>138</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>Party (Elli-Rally)</td>
<td>118</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Unbeast</td>
<td>65</td>
<td>3</td>
</tr>
</tbody>
</table>

**Parallel Execution Mode:**

<table>
<thead>
<tr>
<th>Rank</th>
<th>Tool (conf)</th>
<th>Solved</th>
<th>Unique</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Bosy (exp par)</td>
<td>147</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>Bosy (lin par)</td>
<td>136</td>
<td>0</td>
</tr>
</tbody>
</table>

Not Solved: 7
What is the highest parameter value that can be solved?
TLSF/LTL Results 2016: Synthesis

Number of benchmarks: 185

Sequential and parallel execution modes:

<table>
<thead>
<tr>
<th>Rank</th>
<th>Tool (conf)</th>
<th>Solved</th>
<th>Unique</th>
<th>MC timeout</th>
<th>Wrong Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Bosy (exp parallel)</td>
<td>138</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>Bosy exp</td>
<td>136</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>Bosy lin</td>
<td>129</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>Bosy (lin parallel)</td>
<td>128</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>Party-Elli-Rally</td>
<td>119</td>
<td>6</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>Acacia4Aiger</td>
<td>133</td>
<td>16</td>
<td>20</td>
<td>4</td>
</tr>
</tbody>
</table>

not solved: 21
Only preliminary experiments this year:
• few benchmarks available
• 2 submitted solvers
• challenges in verifying results
**Benchmark Collection for GR(1)**

Only one dedicated GR(1) benchmark submitted from the community:
- simple pursuit evasion (Rüdiger Ehlers, non-parameterized)

Very few LTL benchmarks are directly in GR(1) (when interpreted in strict semantics):
- \(~2\) from Lily benchmark set
- parameterized simple and round robin arbiters

Some TLSF/LTL benchmarks can be translated to GR(1) with additional auxiliary variables:
- AMBA bus controller

Thus far, translation by hand is not enough benchmarks for real competition

More benchmarks are available in the literature and in other formats
Model Checking Solutions for TLSF/GR(1)

0. translate TLSF spec from strict to non-strict semantics
1. translate to AIGER monitor (via SMV)
2. combine monitor with solution
3. use existing AIGER model checkers

Experience:
verification may take much longer than synthesis
(AMBA w/ 4 masters: synthesized in 2.5m,
verification with v3 >100m on 4 cores, with tip >20h on 1 core)
TLSF/GR(1): Submissions 2016

- gr1x (Fillipidis, Murray): BDD-based, fine-grained separation of concerns on different levels
- slugs (Ehlers, Raman): BDD-based, some optimizations during fixpoint computation, focus on getting small implementations, see also [CAV2016]

Both tools support realizability checking and synthesis

Experience:
- for GR(1), a format between full and basic TLSF may be needed (pursuit evasion example: parsing basic TLSF needs much longer than solving)
- dedicated model checking support and/or additional witness information will be needed to verify solutions
- interest is there, but participation was low did we make the right choices? how can we attract more community participation?
A web frontend of our EDACC system is available online, with detailed data on all experiments for SYNTCOMP 2016:
http://syntcomp.cs.uni-saarland.de/syntcomp2016/experiments/

News and announcements for SYNTCOMP are available on
http://www.syntcomp.org
Summary for 2016

AIGER/Safety Track:
• 3 new solvers, and considerable improvements in AbsSynthe
• winning region check makes verification more tractable

TLSF/LTL Track:
• successfully run for the first time
• collected ~100 benchmarks, plus 16 parameterized benchmarks
• 3 participants, one legacy tool

TLSF/GR(1) Track:
• insufficient benchmark library for real competition
• identified problems to be solved:
  benchmark format, benchmark collection, verification
SYNTCOMP 2017 and Beyond: New Challenges?

(Tentative) Goals for 2017:

• Fix GR(1) track
• Re-introduce a quality ranking
• Witnesses:
  - better winning regions?
  - witnesses of unrealizability?
  - witnesses for liveness properties (ranking functions)?
• Keep everything else as it is, to stabilize
Quality ranking:

• 1 point for detecting unrealizability
• for realizable specifications, either:
  
  - $2 - \log_n\left(\frac{\text{solutionsize}}{\text{referencesize}}\right)$ points for a (verifiably correct) solution
  
  - $2 - \log_n\left(\frac{\text{strategysize}}{\text{strategy referencesize}}\right)$ points for a (verifiably correct) solution

where: $n \in \{2,10\}$,

solutionsize is size of returned circuit (including specification),
strategysize is number of additionally synthesized AND-gates,
respective referencesizes are for smallest known solutions of given benchmark
Different Options for Quality Measure with Results of 2016

AIGER/Safety, Best Configurations per Tool:

<table>
<thead>
<tr>
<th>Tool (conf)</th>
<th>Solved</th>
<th>Q2015</th>
<th>Q2015_log2</th>
<th>Strategy_log10</th>
<th>Strategy_log2</th>
</tr>
</thead>
<tbody>
<tr>
<td>AbsSynthe PS1</td>
<td>165</td>
<td>200,0255</td>
<td>186,8612</td>
<td>190,9418</td>
<td>173,0258</td>
</tr>
<tr>
<td>SafetySynth</td>
<td>153</td>
<td>194,8879</td>
<td>190,915</td>
<td>190,5528</td>
<td>185,2559</td>
</tr>
<tr>
<td>Demiurge P3Synt</td>
<td>150</td>
<td>202,0102</td>
<td>202,0337</td>
<td>202,0428</td>
<td>202,142</td>
</tr>
<tr>
<td>SDF</td>
<td>134</td>
<td>162,4012</td>
<td>151,7978</td>
<td>157,9091</td>
<td>147,3515</td>
</tr>
</tbody>
</table>

For LTL track, additional dimension of freedom:
how to assess size of state-space (number of latches) versus size of circuit (number of AND-gates)?
SYNTCOMP 2017 and Beyond: New Challenges?

(Tentative) Goals for 2017:
• Fix GR(1) track
• Re-introduce a quality ranking
• Witnesses: better winning regions? witnesses of unrealizability? witnesses for liveness props?
• Keep everything else to stabilize

Down the road:
• Compositional specifications and partial implementations
• Integration of TLSF and AIGER into one format?
• Encourage real progress, not implementation details:
  Special challenges (e.g., special quality metrics, “Hack Tracks”)?
  Specific classes of benchmarks?
• Extension of system class: imperfect information? timed systems?