IC3 Modulo Theories with Implicit Predicate Abstraction

ES EMBEDDED SYSTEMS

FONDAZIONE BRUNO KESSLER

Source Code

- Download (updated 24th May 2023).
- License: GPLv3.

Requirements

- The latest version of MathSAT.
- A modern C++11 compiler (e.g. a recent version of GCC or Clang).
- CMake (at least version 2.8).

Documentation

- TACAS'14 paper
- CAV'16 paper

  CAV'16 ARTIFACT, containing benchmarks and scripts for the experimental evaluation of the paper.

  The artifact was not evaluated positively by the CAV'16 Artifact Evaluation Committee (CAVAEC). We sustain that we carried out a high quality experimental comparison, that should be easy for anyone to reproduce, and that fully reflects the results reported in the paper. Do not hesitate to get in touch if you have questions and/or feedback.

  The Virtual Machine used for evaluating the artifact is available here.

- VTSA 2015 slides: Part 1, Part 2.
- Quick compilation instructions:

```
$ tar xzf ic3ia.tar.gz
$ cd ic3ia
$ mkdir build
$ cd build
$ cmake .. -DMATHSAT_DIR=/path/to/mathsat -DCMAKE_BUILD_TYPE=Release
$ make
```

- The input format is VMT, described here.
- Here is a script for converting from the nuXmv input language to VMT. Requires nuXmv.
- Some benchmarks.

- **Example (from the slides)**

- 2 state variables *c* and *d*
- Initial states: *(d = 1) & (c >= d)*
- Transition relation: *(c' = c + d) & (d' = d + 1)*
- Property: *(d > 2) => (c > d)*

VMT translation:

```
(declare-const c Int)
(declare-const d Int)
(declare-const c.next Int)
(declare-const d.next Int)

(define-fun sv1 () Int (! c :next c.next))
(define-fun sv2 () Int (! d :next d.next))

(define-fun init () Bool
(! (and (>= c d) (= d 1)) :init true))

(define-fun trans () Bool
(! (and (= c.next (+ c d)) (= d.next (+ d 1))) :trans true))

(define-fun prop () Bool
(! (=> (> d 2) (> c d)) :invar-property 0))
```

Back to Homepage