A Brief Introduction to `Smtlink`

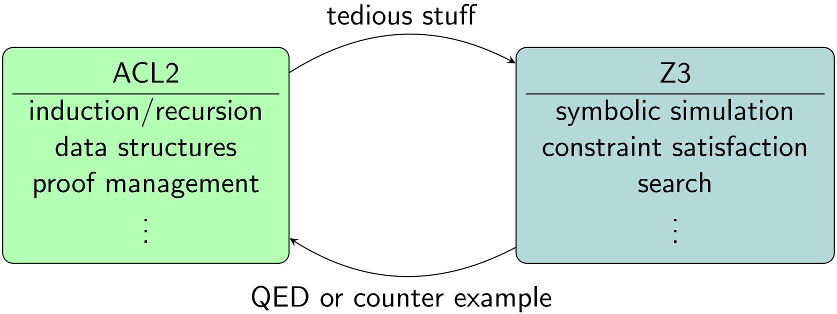
Carl Kwan

The University of Texas at Austin

2020 November 09

Outline

- SMT solvers and Z3
- `Smtlink`
- Basics & Counterexamples (polynomials)
- Uninterpreted functions (`expt`)
- Uninterpreted types (Cauchy-Schwarz)



SMT Solvers

SMT solvers: solves decision problems for logical formulas with respect to theories expressible in first-order logic

E.g., Are there integers $a, b \geq 0$ satisfying

$$\begin{aligned} & (b \geq 2) \\ \wedge\ & (-a + b \leq 1) \\ \wedge\ & (3a + 2b \leq 12) \\ \wedge\ & (2a + 3b \leq 12) \text{ ?} \end{aligned}$$

(decision integer program)

Z3

Z3 supports:

Theory	example
mixed integer linear programming	$a + b \leq x + 1$
non-linear arithmetic	$x^2 \leq 2$
bitvectors	$1001 \oplus x = 0110$
arrays	$a[0] = 10$
datatypes, quantifiers, and more ...	

ACL2 can already do all this ...
What about the non-experts? What about new, ad-hoc, or abstract data structures?

Example that's hard in ACL2, even with `arithmetic` or `non-linear-arithmetic`:

$$x, y \in \mathbb{R},\ x^2 - 2xy + y^2 \geq 0$$

Smtlink

Yan Peng's ACL2 book that calls Z3 at the back end, supports

- basic types: `booleanp`, `integerp`, `real`, `rationalp`, `real/rationap`, `symbolp`.
- FTY types using: `defprod`, `deflist`, `defalist`, `defoption`
- basic functions: `binary-+`, `binary-*`, `unary-/`, `unary--`, `equal`, $<$, `implies`, `if`, `not`, and `lambda`

Polynomials & Counterexamples

Basic example: $x, y \in \mathbb{R}$,

$$x^2 - 2xy + y^2 \geq 0$$

Can be proven in ACL2:

- but would require proving lemmas, instantiating particular theorems, etc.
- Arithmetic books don't help with automation here.

Much easier with `Smtlink`... Let's go take a look!

Polynomials & Counterexamples

e.g., Are there integers $a, b \geq 0$ satisfying

$$(b \geq 2) \wedge (-a + b \leq 1) \wedge (3a + 2b \leq 12) \wedge (2a + 3b \leq 12)$$

In ACL2, try to prove:

```
(not (and (>= b 2)
          (<= (+ (- a) b) 1)
          (<= (+ (* 3 a) (* 2 b)) 12)
          (<= (+ (* 2 a) (* 3 b)) 12)))
```

ACL2 returns:

```
Possible counter-example found:  ((B 2) (A 1))
```

Polynomials & Counterexamples

What if your counterexample is algebraic but not rational? E.g., for $x \in \mathbb{R}$,

$$x(x^2 - 2) = 0 \implies x = 0$$

An S-expression representing the polynomial for which x is a root is returned: `(+ (^ x 2) (- 2))`

Uninterpreted functions

We can prove theorems involving functions that Z3 doesn't know.
E.g., for $x, y, z \in \mathbb{R}$, $z \in (0,1)$, $n, m \in \mathbb{N}_{>0}$,

$$m < n \implies 2z^n xy \leq z^m(x^2 + y^2)$$

Proof:

$$0 \leq z^n(x-y)^2 = z^n(x^2 + y^2 - 2xy) \leq z^m(x^2 + y^2) - 2z^n xy$$

Used: $z^n \geq 0$, $z^m \geq 0$, $z^m \geq z^n$.

Idea: If you understand the human proof, you can give the `Smtlink` proof.

Smtlink hints

From expt example:

```
:smtlink-custom (
  :functions (
    (expt :formals ((r real/rationalp)
                    (i real/rationalp))
          :returns ((ex real/rationalp))
          :level 0))
  :hypotheses (((< (expt z n) (expt z m)))
               ((< 0 (expt z m)))
               ((< 0 (expt z n))))
  :int-to-rat t)
```

- `:functions` – list of functions, with formal arguments, and expansion level (`:level 0` is uninterpreted)
- `:hypotheses` – theorems that are true in ACL2 which `Smtlink` can use to help with the proof
- `:int-to-rat` – coerce all integers to reals (!)

Why real?

Why might we want to use real numbers?

- Real arithmetic is easier than mixed integer/real in Z3.

What's the worst that can happen?

- If the theorem is true for the reals, then it's true for the integers.
- If the theorem isn't true for the reals but true for the integers, then the Z3 proof fails.
- If the theorem isn't true for the reals nor the integers, then the Z3 proof fails.
- ACL2 logical world still OK.

Caution:

- `(equal (/ x 0) 0)`, see XDOC[1]
- Other logical issues (e.g., are the models of Z3 and ACL2 compatible?)

[1] or `[books]/projects/smtlink/examples/examples.lisp`

Uninterpreted types

Reasoning about objects that aren't natively supported by Z3.

```
(encapsulate
  (((abstract-p *) => *))
  (local
   (defun abstract-p (x)
       (acl2::any-p x))))

(defthm abstract-example
  (implies (abstract-p x)
           (equal x x))
  :hints(("Goal"
          :smtlink (:abstract (abstract-p))))
:rule-classes nil)
```

`Smtlink` requires a type-recogniser for each free variable in the hypotheses.

Uninterpreted types

For scalar $a \in \mathbb{R}$, vectors $u, v \in V$, inner product $\langle -, - \rangle : V \times V \to \mathbb{R}$, we have

$$
\begin{aligned}
& \langle u - av, u - av \rangle = \cdots & \\
&= \langle u, u \rangle + (-a)\langle u, v \rangle + (-a)\langle v, u \rangle + (-a)(-a)\langle v, v \rangle & \text{bilinearity} \\
&= \langle u, u \rangle + (-a)\langle u, v \rangle + (-a)\langle u, v \rangle + (-a)(-a)\langle v, v \rangle & \text{commutativity} \\
&= \langle u, u \rangle - 2a\langle u, v \rangle + a^2\langle v, v \rangle & \text{field operations}
\end{aligned}
$$

A key step in the proof of Cauchy-Schwarz.

Conclusion

- We saw some existing applications of `Smtlink`.
- Future applications: matrices, lattice-based encryption.
- What do you want to do?
- Caution: logic

References

- XDOC: `cs.utexas.edu/users/moore/acl2/manuals/current/manual/index.html?topic=SMT____SMTLINK`
- Yan Peng, Mark R. Greenstreet. *Smtlink 2.0*. 15th International Workshop on the ACL2 Theorem Prover and Its Applications (ACL2-2018).
- `expt` example: `/books/projects/smtlink/examples/examples.lisp`
- Carl Kwan, Yan Peng, Mark R. Greenstreet. *Cauchy-Schwarz in ACL2(r) Abstract Vector Spaces*. 16th International Workshop on the ACL2 Theorem Prover and Its Applications (ACL2-2020).
- Leonardo de Moura, Nikolaj Bjørner. *Z3: An Efficient SMT Solver*. Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2008).

Thank you!