Auditing User-provided Axioms in Software Verification Conditions

Paul Jackson[1], Florian Schanda[2] and Angela Wallenburg[2]

1. University of Edinburgh

2. Altran UK (Praxis)

Rich Model Toolkit COST Action Meeting
Malta
17th June 2013

Verification-Condition (VC) based software verification

The idea

Start with programs annotated with assertions

1. Generate FOL VCs sufficient to establish assertions
2. Prove VCs

Example tools

Boogie, Why3

- Support C, C#, Java and Ada
- Use provers Z3, Alt-Ergo, CVC4

Context of reported work

- Altran's SPARK-Ada verification tool-set
- Victor SMT solver interface
- Z3

Axiom uses

1. Giving properties of specification relations and functions
    - E.g. a permutation relation for a sorting program
2. Providing hints to automatic provers
    - VCs intractable or undecidable in general
        - Involve quantifiers and non-linear arithmetic
    - Addressing the 1-5% of VCs not automatically proved
        - Check by hand
        - Use interactive prover
        - Add axiom for proof step automatic prover is missing

Problems with using axioms

- Can introduce inconsistencies
  - Then have risk of prover claiming false VCs to be true
- Costly to create and maintain
  - Takes 15 mins – 1+ days to write an axiom
  - Axioms can need revisiting when programs change

Checking axiom properties

VCs of form $S \wedge U \wedge H \Rightarrow C$

with	S: system-provided axioms	H: hypotheses
	U: user-provided axioms $u_1, \ldots, u_n$	C: conclusions

Automatic proof attempted of goals of following kinds:

Kind	Goal shape	Description
S-incon	$S \Rightarrow \bot$	Are system axioms inconsistent?
U-incon	$S \wedge U \Rightarrow \bot$	Are user axioms inconsistent?
u-incon	$S \wedge u_i \Rightarrow \bot$	Is user axiom u_i inconsistent?
u-taut	$S \Rightarrow u_i$	Is user axiom u_i a tautology?
u-deriv	$S \wedge (U \setminus \{u_i\}) \Rightarrow u_i$	Is user axiom u_i derivable from other user axioms?

Unsat cores used to identify formulas involved in proofs

Finding minimal axiom sets

- Unused axioms common as provers get better
- Iteratively tried removing user-provided axioms while ensuring provability of VCs unchanged

Industrial Case Studies

Tokeneer ID Station

- Commissioned by NSA to evaluate SPARK
- 10k lines decls and executable code, 2k lines annotations
- 7k VCs, 107 user-provided axioms

Arithmetic on Integers and Floats

- Part of an industrial evaluation of SPARK
- Library of 30 functions and procedures
- 25 user-provided axioms concerning float-to-integer conversions

Inconsistent hint axiom 1

- Detected by *u-incon* check

```
 B1 and Op = Op_1 -> B2
 may_be_deduced_from
[ St = St_1 or (St = St_2 or St = St_3),
  St_1 <> St_2,
  St_1 <> St_3,
  St_2 <> St_3,
  St = St_1 or St = St_2 -> B1 and (B3 and Op = Op_2),
  Op_1 <> Op_2,
  St = S_3 -> not B1 ].
```

Inconsistent hint axiom 2

- Not detected by *u-incon* check
- Considered suspicious since it failed *u-taut* check

```
X - (Y - 1) * 100 <= 200 -> Y + 1 = (X - 1) div 100 + 1
                 may_be_deduced_from
               [ 100 < X - (Y - 1) * 100,
                 goal(checktype(X, integer)),
                 goal(checktype(Y, integer)) ] .
```

- Incorrect abstraction of VC subgoal unproved by Altran prover
- VC proved by Z3

Axiom inter-relationships

- Detected with *u-deriv* check and unsat core report

With

$$\begin{aligned} A_1 &: \; e(s) \Rightarrow \neg w(s) \\ A_2 &: \; (e(s) \vee p(s)) \Rightarrow \neg w(s) \\ A_7 &: \; p(s) \Rightarrow \neg w(s) \end{aligned}$$

found

$$\begin{gathered} A_2 \Rightarrow A_1 \\ A_1 \wedge A_7 \Rightarrow A_2 \\ A_2 \Rightarrow A_7 \end{gathered}$$

- For Tokeneer, 25 inter-relationships found among 107 axioms

Minimal axiom set discovery

50 of 107 Tokeneer user axioms found redundant

- 40 prover hints
- 3 unused property axioms
- 7 were property axioms subsumed by others

Mutually-inconsistent property axioms

$$\begin{aligned}
c0:&\quad \forall x:\mathbf{R}.\ x \le k-1 \ \Rightarrow\ \mathrm{ceil}(x) \le x+1\\
c1:&\quad \forall x:\mathbf{R}.\ x \le k-1 \ \Rightarrow\ \mathrm{ceil}(x) \le k\\
c2:&\quad \forall x:\mathbf{R}.\ x \le k-1 \ \Rightarrow\ x \le \mathrm{ceil}(x)\\
c3:&\quad \forall x:\mathbf{R}.\ x \le k-1 \ \Rightarrow\ -k \le \mathrm{ceil}(x)
\end{aligned}$$

Here k is the largest floating point number

- *U-incon* check identified that $c0$ and $c3$ were contradictory
- Z3 missed a similar *U-incon* check on axioms for floor function
  - Inconsistency picked up in *u-deriv* check where conclusion was not part of unsat core

Related work

- VCC - Boogie front-end for C
  - Can try to prove control points unreachable
  - Sometimes due to inconsistencies in axioms

- Why3
  - Can find minimal axiom sets

- K. Y. Ahn and E. Denney (2012)

  For axiom $\forall x.\ A(x) \Rightarrow B(x)$
  - Yices SMT solver finds satisfying assignments for $A(x)$
  - QuickCheck tries to find x such that $\neg B(x)$

  Used on aerospace flight code at NASA

Conclusions

- Automatic auditing of user-provided axioms can be useful
- Current/future work
  - Auditing real industrial examples
  - Persuading Altran & customers to audit *during* axiom development
  - Assisting switch from Altran's prover to SMT solvers