

Decision Procedures

Jochen Hoenicke



Software Engineering
Albert-Ludwigs-University Freiburg

Summer 2013

Organisation

Dates

- Lecture is Tuesday 10–12 (c.t) and Thursday 10–11 (c.t).
- Tutorials will be given on Thursday 11–12.
Starting next week (this week is a two hour lecture).
- Exercise sheets are uploaded on Tuesday.
They are due on Tuesday the week after.

To successfully participate, you must

- prepare the exercises (at least 50 %)
- actively participate in the tutorial
- pass an oral examination

THE CALCULUS OF COMPUTATION:
Decision Procedures with
Applications to Verification

by
Aaron Bradley
Zohar Manna

Springer 2007

Motivation

Decision Procedures are algorithms to decide formulae.

These formulae can arise

- in Hoare-style software verification.
- in hardware verification

Consider the following program:

```
for
  @  $l \leq i \leq u \wedge (rv \leftrightarrow \exists j. l \leq j < i \wedge a[j] = e)$ 
  (int  $i := l; i \leq u; i := i + 1$ ) {
  if (( $a[i] = e$ )) {
     $rv := \text{true};$ 
  }
}
```

How can we prove that the **formula** is a loop invariant?

Prove the Hoare triples (one for if case, one for else case)

assume $l \leq i \leq u \wedge (rv \leftrightarrow \exists j. l \leq j < i \wedge a[j] = e)$

assume $i \leq u$

assume $a[i] = e$

$rv := \text{true};$

$i := i + 1$

@ $l \leq i \leq u \wedge (rv \leftrightarrow \exists j. l \leq j < i \wedge a[j] = e)$

assume $l \leq i \leq u \wedge (rv \leftrightarrow \exists j. l \leq j < i \wedge a[j] = e)$

assume $i \leq u$

assume $a[i] \neq e$

$i := i + 1$

@ $l \leq i \leq u \wedge (rv \leftrightarrow \exists j. l \leq j < i \wedge a[j] = e)$

A Hoare triple $\{P\} S \{Q\}$ holds, iff

$$P \rightarrow wp(S, Q)$$

(wp denotes is weakest precondition)

For assignments wp is computed by substitution:

assume $l \leq i \leq u \wedge (rv \leftrightarrow \exists j. l \leq j < i \wedge a[j] = e)$

assume $i \leq u$

assume $a[i] = e$

$rv := \text{true};$

$i := i + 1$

@ $l \leq i \leq u \wedge (rv \leftrightarrow \exists j. l \leq j < i \wedge a[j] = e)$

holds if and only if:

$$l \leq i \leq u \wedge (rv \leftrightarrow \exists j. l \leq j < i \wedge a[j] = e) \wedge i \leq u \wedge a[i] = e \\ \rightarrow l \leq i + 1 \leq u \wedge (\text{true} \leftrightarrow \exists j. l \leq j < i + 1 \wedge a[j] = e)$$

We need an algorithm that decides whether a formula holds.

$$\begin{aligned} & \ell \leq i \leq u \wedge (rv \leftrightarrow \exists j. \ell \leq j < i \wedge a[j] = e) \wedge i \leq u \wedge a[i] = e \\ \rightarrow & \ell \leq i + 1 \leq u \wedge (\text{true} \leftrightarrow \exists j. \ell \leq j < i + 1 \wedge a[j] = e) \end{aligned}$$

If the formula does not hold it should give a counterexample, e.g.:

$$\ell = 0, i = 1, u = 1, rv = \text{false}, a[0] = 0, a[1] = 1, e = 1,$$

This counterexample shows that $i + 1 \leq u$ can be violated.

This lecture is about algorithms checking for validity and producing these counterexamples.

Contents of Lecture

- Propositional Logic
- First-Order Logic
- First-Order Theories
- Quantifier Elimination
- Decision Procedures for Linear Arithmetic
- Decision Procedures for Uninterpreted Functions
- Decision Procedures for Arrays
- Combination of Decision Procedures
- DPLL(T)
- Craig Interpolants