CS 512, Spring 2018, Handout 16

Hoare Logic

Assaf Kfoury

13 March 2018

an imperative programming language

- integer expressions

$$E \;::=\; n \mid x \mid E_1 + E_2 \mid E_1 - E_2 \mid E_1 * E_2 \mid \cdots$$

where x ranges over the set of all variables and
n ranges over the set of all numerals $\{\ldots, -2, -1, 0, 1, 2, \ldots\}$

- boolean expressions

$$B \;::=\; \texttt{true} \mid \texttt{false} \mid \neg B \mid B_1 \wedge B_2 \mid B_1 \vee B_2 \mid E_1 = E_2 \mid E_1 < E_2 \mid \cdots$$

- program expressions (or commands)

$$C \;::=\; x := E \mid C_1; C_2 \mid \textbf{if } B \textbf{ then } C_1 \textbf{ else } C_2 \mid \textbf{while } B \textbf{ do } C \textbf{ od}$$

programming examples

- program Fac1(x) computes factorial of x and stores result in y (Example 4.2, in [LCS], page 262)

$y := 1;$
$z := 0;$
while $\neg(x = z)$ **do**
$z := z + 1;$
$y := y * z;$
od

- Are we sure Fac1(x) computes the factorial of x?

  What does it compute on input $x = -3$?

  Does it terminate on input $x = -3$?

programming examples

- what does program foo(z) compute?

$x := 1;$
$y := 0;$
while $(x < z)$ **do**
$x := x + x;$
$y := y + 1;$
od

- **claim:**
given an integer $z > 0$, foo(z) returns in y the value of $\lfloor \log_2 z \rfloor$

but why should you believe the claim?

or how can you confirm the claim?

programming examples

- what does program bar(m, n) compute?

while $(n \neq 0)$ **do**
  $r := m;$
  while $(r \geqslant n)$ **do**
    $r := r - n;$
    od;
  $m := n;$
  $n := r;$
od

- **claim:**
  given $m, n > 0$, bar(m, n) computes their gcd and stores it in m

  how can you confirm the claim?

Hoare triples

- $\{\,\varphi\,\}\ P\ \{\,\psi\,\}$ where
  - φ is called the *pre-condition*, typically a wff of first-order logic
  - P is a program
  - ψ is called the *post-condition*, typically a wff of first-order logic
  - quantifiers in φ and ψ, if any, cannot be used to bind variables in P

- **informally:** "if program P is run in a *state* satisfying φ,
  then the *state* resulting from P's execution will satisfy ψ,
  if and when P stops"
- an **input state** is defined relative to a model (or interpretation) $\mathcal{M}$ and an environment (or "lookup table") ℓ that assigns a value to every free variable in φ.
- and similarly for an **output state**

examples using Hoare triples

- write a program P satisfying the specification

$$\{\, x > 0 \quad \}\, P\, \{ \quad y \cdot y < x \,\}$$

- let P be the program "$y := 0$"

  (not very interesting!!)

- let P be the program

$$\begin{array}{l} y := 0; \\ \textbf{while} \quad (y * y < x)\ \textbf{do} \\ \qquad\quad y := y + 1; \\ \qquad\quad \textbf{od}; \\ y := y - 1 \end{array}$$

  (y is the largest integer whose square is less than x)

  what happens if we change the pre-condition to "$x \geqslant 0$"? or to "**`true`**"?

(THIS PAGE INTENTIONALLY LEFT BLANK)