

Specification as a development task

Given precondition φ and postcondition ψ
develop a program S such that

$$\{\varphi\} S \{\psi\}$$

For instance

Find S such that

$$\{n \geq 0\} S \{rt^2 \leq n \wedge n < (rt + 1)^2\}$$

One correct solution:

$$\{n \geq 0\}$$

$rt := 0; sqr := 1;$

while $sqr \leq n$ **do** $rt := rt + 1; sqr := sqr + 2 * rt + 1$

$$\{rt^2 \leq n \wedge n < (rt + 1)^2\}$$

Hoare's logic: trouble #1

Another correct solution:

$$\{n \geq 0\}$$

while true do skip

$$\{rt^2 \leq n \wedge n < (rt + 1)^2\}$$

since $\vdash$

$$\{n \geq 0\}$$

while {true} true do skip

$$\{rt^2 \leq n \wedge n < (rt + 1)^2\}$$

Partial correctness:

termination not guaranteed,
and hence **not requested!**

Total correctness

Total correctness = partial correctness + successful termination

Total correctness judgements:

$$[\varphi] S [\psi]$$

Intended meaning:

*Whenever the program S starts in a state satisfying the precondition φ
then it terminates successfully in a final state that satisfies the postcondition ψ*

Total correctness: semantics

$$\begin{array}{c} \models [\varphi] S [\psi] \\ \text{iff} \\ \{\varphi\} \subseteq \llbracket S \rrbracket \{\psi\} \end{array}$$

where for $S \in \mathbf{Stmt}$, $A \subseteq \mathbf{State}$:

$$\llbracket S \rrbracket A = \{s \in \mathbf{State} \mid \mathcal{S}[\llbracket S \rrbracket] s = a, \text{ for some } a \in A\}$$

Spelling this out:

The total correctness judgement $[\varphi] S [\psi]$ holds, written $\models [\varphi] S [\psi]$,
if for all states $s \in \mathbf{State}$

$$\text{if } \mathcal{F}[\varphi] s = \mathbf{tt} \text{ then } \mathcal{S}[\llbracket S \rrbracket] s \in \mathbf{State} \text{ and } \mathcal{F}[\psi] (\mathcal{S}[\llbracket S \rrbracket] s) = \mathbf{tt}$$

Total correctness: proof rules

$$\frac{}{[\varphi[x \mapsto e]] \ x := e \ [\varphi]}$$

$$\frac{}{[\varphi] \ \mathbf{skip} \ [\varphi]}$$

$$\frac{[\varphi] \ S_1 \ [\theta] \quad [\theta] \ S_2 \ [\psi]}{[\varphi] \ S_1; S_2 \ [\psi]}$$

$$\frac{[\varphi \wedge b] \ S_1 \ [\psi] \quad [\varphi \wedge \neg b] \ S_2 \ [\psi]}{[\varphi] \ \mathbf{if} \ b \ \mathbf{then} \ S_1 \ \mathbf{else} \ S_2 \ [\psi]}$$

$$\frac{???}{[???] \ \mathbf{while} \ b \ \mathbf{do} \ S \ [???)}$$

$$\frac{\varphi' \Rightarrow \varphi \quad [\varphi] \ S \ [\psi] \quad \psi \Rightarrow \psi'}{[\varphi'] \ S \ [\psi']}$$

Adjustments are necessary if expressions may generate errors!

Total-correctness rule for loops

$$\frac{(\text{nat}(l) \wedge \varphi(l+1)) \Rightarrow b \quad [\text{nat}(l) \wedge \varphi(l+1)] S [\varphi(l)] \quad \varphi(0) \Rightarrow \neg b}{[\exists l. \text{nat}(l) \wedge \varphi(l)] \text{ while } b \text{ do } S [\varphi(0)]}$$

where

- $\varphi(l)$ is a formula with a free variable l that does not occur in **while** b **do** S ,
- $\text{nat}(l)$ stands for $0 \leq l$, and
- $\varphi(l+1)$ and $\varphi(0)$ result by substituting, respectively, $l+1$ and 0 for l in $\varphi(l)$.

Informally:

l is a *counter*

that indicates the number of iterations of the loop body

For example

To prove:

$$\begin{aligned} &[n \geq 0 \wedge rt = 0 \wedge sqr = 1] \\ &\quad \textbf{while } sqr \leq n \textbf{ do} \\ &\quad \quad rt := rt + 1; \text{ } sqr := sqr + 2 * rt + 1 \\ &[rt^2 \leq n \wedge n < (rt + 1)^2] \end{aligned}$$

use the following invariant with the iteration counter l :

$$sqr = (rt + 1)^2 \wedge rt^2 \leq n \wedge l = \lfloor \sqrt{n} \rfloor - rt$$

Cheating here, of course:

“ $l = \lfloor \sqrt{n} \rfloor - rt$ ” has to be captured by
a first-order formula in the language of TINY

Luckily: this can be done!

Here, this is quite easy:
 $(rt + l)^2 \leq n < (rt + l + 1)^2$

Well-founded relations

A relation $\succ \subseteq W \times W$ is *well-founded* if there is no infinite chain

$$a_0 \succ a_1 \succ \dots \succ a_i \succ a_{i+1} \succ \dots$$

Typical example:

$$\langle \mathbf{Nat}, > \rangle$$

BTW: For well-founded $\succ \subseteq W \times W$, its transitive and reflexive closure $\succ^* \subseteq W \times W$ is a partial order on W .

BUT: subtracting identity from an arbitrary partial order on W need not in general yield a well-founded relation.

Few other examples:

- $\mathbf{Nat}^n$ with component-wise (strict) ordering;
- A^* with proper prefix ordering;
- $\mathbf{Nat}^n$ with lexicographic (strict) ordering generated by the usual ordering on $\mathbf{Nat}$;
- any ordinal with the natural (strict) ordering; etc.

Total correctness = partial correctness + successful termination

Proof method

To prove

$[\varphi] \textbf{while } b \textbf{ do } S [\varphi \wedge \neg b]$

- show “partial correctness”: $[\varphi \wedge b] S [\varphi]$
- show “termination”: find a set W with a well-founded relation $\succ \subseteq W \times W$ and a function $w: \textbf{State} \rightarrow W$ such that for all states $s \in \{\varphi \wedge b\}$,

$w(s) \succ w(\mathcal{S}[[S]] s)$

BTW: $w: \textbf{State} \rightarrow W$ may be partial as long as it is defined on $\{\varphi \wedge b\}$.

Example

Prove:

```
[ $x \geq 0 \wedge y \geq 0$ ]  
  while  $x > 0$  do  
    if  $y > 0$  then  $y := y - 1$  else  $(x := x - 1; y := f(x))$   
  [true]
```

where f yields a natural number for any natural argument.

- If one knows nothing more about f , then the previous proof rule for the total correctness of loops is useless here.
- **BUT:** termination can be proved easily using the function $w: \mathbf{State} \rightarrow \mathbf{Nat} \times \mathbf{Nat}$, where $w(s) = \langle s\ x, s\ y \rangle$:
after each iteration of the loop body the value of w decreases w.r.t. the (well-founded) lexicographic order on pairs of natural numbers.

A fully specified program

```
 $[x \geq 0 \wedge y \geq 0]$   
while  $[x \geq 0 \wedge y \geq 0]$   $x > 0$  do decr  $\langle x, y \rangle$  in  $\mathbf{Nat} \times \mathbf{Nat}$  wrt  $\succ$   
    if  $y > 0$  then  $y := y - 1$  else  $(x := x - 1; y := f(x))$   
 $[\mathbf{true}]$ 
```

... with various notational variants
assuming some external definitions for
the well-founded set and function into it

Hoare's logic: trouble #2

Find S such that

$$\{n \geq 0\} S \{rt^2 \leq n \wedge n < (rt + 1)^2\}$$

Another correct solution:

$$\begin{array}{l} \{n \geq 0\} \\ \quad rt := 0; n := 0 \\ \{rt^2 \leq n \wedge n < (rt + 1)^2\} \end{array}$$

OOOOOPS?!

A number of techniques to avoid this:

- variables that are required not to be used in the program;
- binary postconditions;
- various forms of algorithmic/dynamic logic, with program modalities.

Binary postconditions

Sketch

- New syntactic category **BForm** of *binary formulae*, which are like the usual formulae, except they can use both the usual variables $x \in \mathbf{Var}$ and their “past” copies $\hat{x} \in \widehat{\mathbf{Var}}$.

For any syntactic item ω , we write $\hat{\omega}$ for ω with each variable x replaced by $\hat{x}$.

- Semantic function: $\mathcal{BF}: \mathbf{BForm} \rightarrow \mathbf{State} \times \mathbf{State} \rightarrow \mathbf{Bool}$

$\mathcal{BF}[\psi] \langle s_0, s \rangle$ is defined as usual, except that the state s_0 is used to evaluate “past” variables $\hat{x} \in \widehat{\mathbf{Var}}$ and s is used to evaluate the usual variables $x \in \mathbf{Var}$.

Correctness judgements

$$pre\ \varphi; S\ post\ \psi$$

where $\varphi \in \mathbf{Form}$ is a (unary) precondition; $S \in \mathbf{Stmt}$ is a statement (as usual); and $\psi \in \mathbf{BForm}$ is a binary postcondition.

Semantics:

The judgement $pre\ \varphi; S\ post\ \psi$ holds, written $\models pre\ \varphi; S\ post\ \psi$, if for all states $s \in \mathbf{State}$

if $\mathcal{F}[\varphi]\ s = \mathbf{tt}$ then $\mathcal{S}[S]\ s \in \mathbf{State}$ and $\mathcal{BF}[\psi]\ \langle s, \mathcal{S}[S]\ s \rangle = \mathbf{tt}$

Proof rules

$$\frac{}{pre\ \varphi; x := e\ post\ (\widehat{\varphi} \wedge x = \widehat{e} \wedge \vec{y} = \widehat{\vec{y}})}$$

where $\vec{y}$ are variables other than x .

$$\frac{}{pre\ \varphi; \text{skip}\ post\ (\varphi \wedge \vec{y} = \widehat{\vec{y}})}$$

$$\frac{pre\ \varphi_1; S_1\ post\ (\psi_1 \wedge \varphi_2) \quad pre\ \varphi_2; S_2\ post\ \psi_2}{pre\ \varphi_1; S_1; S_2\ post\ \psi_1 * \psi_2}$$

where $\psi_1 * \psi_2$ is $\exists \vec{z}. (\psi_1[\vec{x} \mapsto \vec{z}] \wedge \psi_2[\widehat{\vec{x}} \mapsto \vec{z}])$, with all the variables free in ψ_1 or ψ_2 are among $\vec{x}$ or $\widehat{\vec{x}}$, and $\vec{z}$ are new variables.

Further rules

$$\frac{\text{pre } \varphi \wedge b; S_1 \text{ post } \psi \quad \text{pre } \varphi \wedge \neg b; S_2 \text{ post } \psi}{\text{pre } \varphi; \text{ if } b \text{ then } S_1 \text{ else } S_2 \text{ post } \psi}$$

$$\frac{\text{pre } \varphi \wedge b; S \text{ post } (\psi \wedge \widehat{e} \succ e) \quad \psi \Rightarrow \varphi \quad (\psi * \psi) \Rightarrow \psi}{\text{pre } \varphi; \text{ while } b \text{ do } S \text{ post } ((\psi \vee (\varphi \wedge \vec{y} = \widehat{\vec{y}})) \wedge \neg b)}$$

where $\succ$ is well-founded, and all the free variables are among $\vec{y}$ or $\widehat{\vec{y}}$.

$$\frac{\varphi' \Rightarrow \varphi \quad \text{pre } \varphi; S \text{ post } \psi \quad \psi \Rightarrow \psi'}{\text{pre } \varphi'; S \text{ post } \psi'}$$

$$\frac{\text{pre } \varphi; S \text{ post } \psi}{\text{pre } \varphi; S \text{ post } (\widehat{\varphi} \wedge \psi)}$$

The rules can (have to?) be polished...

Example

We have now:

$$\models$$

```
pre  $n \geq 0$ ;  
   $rt := 0; sqr := 1$ ;  
  while  $sqr \leq n$  do  $rt := rt + 1; sqr := sqr + 2 * rt + 1$   
post  $rt^2 \leq \hat{n} \wedge \hat{n} < (rt + 1)^2$ 
```

BUT : $\not\models$

$$\begin{array}{l} \{n \geq 0\} \\ rt := 0; n := 0 \\ \{rt^2 \leq \hat{n} \wedge \hat{n} < (rt + 1)^2\} \end{array}$$

Algorithmic/dynamic logic

Sketch

- Salwicki 1970
- Pratt 1974, Harel 1976
- many others to follow (see Harel, Kozen & Tiuryn 2000)

Overall idea:

*Extend the logical formulae so that they are closed under the usual logical connectives and quantification, as well as under **program modalities***

Syntax: For any formula φ and a statement $S \in \mathbf{Stmt}$, build a new formula:

$\langle S \rangle \varphi$

Semantics: $\mathcal{F}[\langle S \rangle \varphi] s = \begin{cases} \mathcal{F}[\varphi] s' & \text{if } \mathcal{S}[S] s = s' \in \mathbf{State} \\ \mathbf{ff} & \text{if } \mathcal{S}[S] s \notin \mathbf{State} \end{cases}$

Proof system

... axioms and rules to handle the standard connectives and quantification ...

Plus axioms and rules to deal with program modalities — interaction between modalities and propositional connectives; (de)composition of modalities — for instance:

$$\langle S \rangle (\varphi \wedge \psi) \iff (\langle S \rangle \varphi \wedge \langle S \rangle \psi)$$

$$\langle S \rangle \neg \varphi \implies \neg \langle S \rangle \varphi$$

$$\langle S \rangle \mathbf{true} \implies (\neg \langle S \rangle \varphi \implies \langle S \rangle \neg \varphi)$$

$$\langle S_1; S_2 \rangle \varphi \iff \langle S_1 \rangle (\langle S_2 \rangle \varphi)$$

etc.

Key to the completeness here:

infinitary rules for loops