

Generalising

Rather than just working with \mathbf{Int} , consider an arbitrary underlying data type given by:

- Σ : an algebraic signature with sort *Bool* and boolean constants and connectives
- \mathcal{A} : a Σ -structure with the boolean part interpreted in the standard way

$\mathrm{TINY}_\mathcal{A}$

Syntax: As in TINY, except that:

- $\Sigma\text{-terms}$ used instead of integer expressions
- variables classified by the sorts of Σ , assignments allowed only when the sorts of the variable and the term coincide
- $-\Sigma$ -terms of sort *Bool* used instead of boolean expressions

Semantic domains: As in $T{\sc INY},$ except with a modified notion of state:

$$\mathbf{State}_{\mathcal{A}} = \mathbf{Var} \to |\mathcal{A}|$$

(with variables and their values classified by the sorts of Σ)

Semantic functions: As in TINY, except that referring to \mathcal{A} for interpretation of the operations on $|\mathcal{A}|$.

Hoare's logic



— — — as before — — —

For instance

• add the following to the original signature Σ for TINY:

• and expand the original algebra \mathcal{A} for TINY as follows:

carriers	$\mathcal{A}_{Array} = \mathbf{Int} o \mathbf{Int}$
operations	$newarr_{\mathcal{A}}(j) = 0$
	$put_{\mathcal{A}}(a, i, n) = a[i \mapsto n]$
	$get_{\mathcal{A}}(a,i) = a(i)$

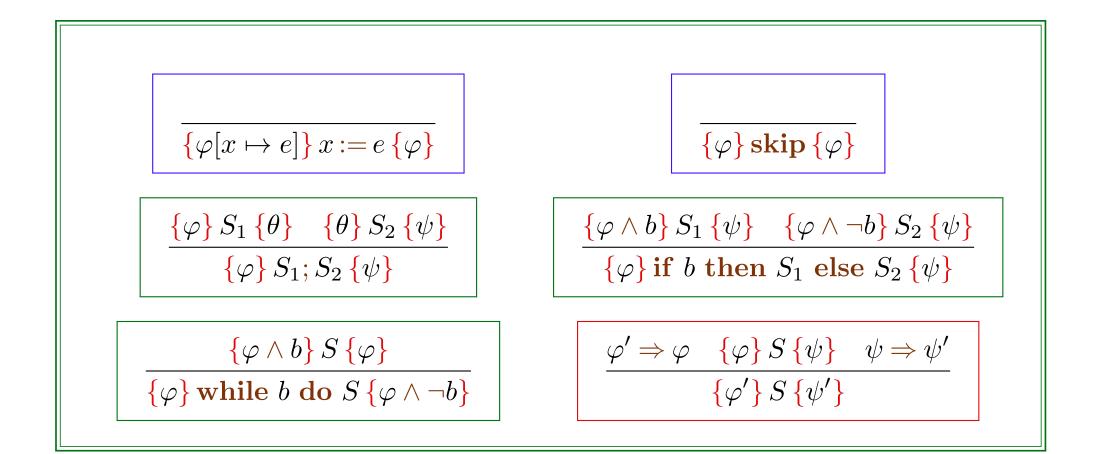
Example

$$\begin{aligned} &\{a: Array \land 0 \leq n\} \\ &m := 0; \\ & \textbf{while } \{0 \leq m \leq n \land is\text{-sorted}(a, 0, m)\} \ m + 1 \leq n \ \textbf{do} \\ &m := m + 1; k := m; \\ & \textbf{while } \{0 \leq k \leq m \leq n \land is\text{-nearly-sorted}(a, 0, k, m)\} \ 1 \leq k \ \textbf{do} \\ &k := k - 1; \\ & \textbf{if } get(a, k) \leq get(a, k + 1) \ \textbf{then } k := 0 \\ & \textbf{else } x := get(a, k + 1); a := put(a, k + 1, get(a, k)); a := put(a, k, x) \\ &\{is\text{-sorted}(a, 0, n)\} \end{aligned}$$

where:

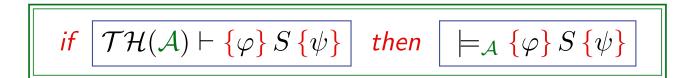
$$\begin{split} is-sorted(a,i,j) &\equiv a : Array \land \forall i', j' : Int. i \leq i' \leq j' \leq j \Rightarrow get(a,i') \leq get(a,j') \\ is-nearly-sorted(a,i,k,j) &\equiv is-sorted(a,i,k-1) \land is-sorted(a,k,j) \land \\ \forall i', j' : Int. (i \leq i' \leq k-1 \land k+1 \leq j' \leq j) \Rightarrow get(a,i') \leq get(a,j') \end{split}$$

Hoare's logic: proof rules



Soundness

Fact: Hoare's proof calculus is sound, that is:







Toward completeness

We have to ensure that all the assertions necessary in the proofs may be formulated in the assertion logic.

Given $S \in \mathbf{Stmt}_{\Sigma}$ and $\psi \in \mathbf{Form}_{\Sigma}$, define:

 $wpre_{\mathcal{A}}(S,\psi) = \{s \in \mathbf{State}_{\mathcal{A}} \mid \text{ if } \mathcal{S}_{\mathcal{A}}\llbracket S \rrbracket s = s' \in \mathbf{State}_{\mathcal{A}} \text{ then } \mathcal{F}_{\mathcal{A}}\llbracket \psi \rrbracket s' = \mathbf{tt} \}$

Definition: First-order logic is expressive over \mathcal{A} for $\operatorname{TINY}_{\mathcal{A}}$ (\mathcal{A} is expressive) if for all $S \in \operatorname{Stmt}_{\Sigma}$ and $\psi \in \operatorname{Form}_{\Sigma}$, there exists the weakest liberal precondition for S and ψ , that is, a formula $\varphi_0 \in \operatorname{Form}_{\Sigma}$ such that

$$\{\varphi_0\}_{\mathcal{A}} = wpre_{\mathcal{A}}(S,\psi)$$

Relative completeness of Hoare's logic

(completeness in the sense of Cook)

Fact: If A is expressive then Hoare's proof calculus is sound and relatively complete, that is:

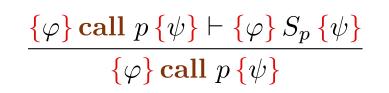
$$\mathcal{TH}(\mathcal{A}) \vdash \{\varphi\} S \{\psi\} \quad iff \quad \models_{\mathcal{A}} \{\varphi\} S \{\psi\}$$

Proof: By structural induction on S. In fact: given expressivity and arbitrary use of facts from $\mathcal{TH}(\mathcal{A})$, all the cases go through easily!

Fact: \mathcal{A} is expressive if and only if either the standard model of Peano arithmetic is definable in \mathcal{A} , or for each $S \in \mathbf{Stmt}_{\Sigma}$, there is a finite bound on the number of states reached in any computation of S.

Beyond TINY

Procedures: Given proc p is (S_p) :



Not quite good enough; requires additional rules to manipulate auxiliary variables to ensure relative completeness

Variables: Given a fresh variable *y*:

$$\frac{\{\varphi \land y = ??\} S[x \mapsto y] \{\psi\}}{\{\varphi\} \text{ begin var } x \text{ } S \text{ end } \{\psi\}}$$

etc. . .

But there are limits...

Fact: There exists no Hoare's proof system which is sound and relatively complete in the sense of Cook for a programming language which admits recursive procedures with procedure parameters, local procedures and global variables with static binding. Key to the proof:

Fact: The halting problem is undecidable for programs of such a language even for finite data types \mathcal{A} (with at least two elements).

Total correctness revisited

What about $TINY_{\mathcal{A}}$?

GOOD NEWS:

Proving termination using well-founded relations works as before!

Still, recall the basic rule:

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

Problem?

Given a signature Σ , let Σ^+ be its extension by the language of (Peano) arithmetic: predicates $nat(_)$ and $_ \leq _$, constants 0, 1, operations $_ + _$, $_ - _$, $_ * _$.

Let \mathcal{A} be a Σ^+ -structure; assume that the interpretation of $nat(_)$ in \mathcal{A} is closed under the arithmetical constants and operations as expected.

Even then:

the loop rule need not be sound for $\mathrm{TINY}_\mathcal{A}$

For instance, we will typically get:

Serious trouble?

 $\mathcal{TH}(\mathcal{A}) \vdash [nat(x)]$ while x > 0 do x := x - 1 [true]

BUT: This is not valid for instance if A is a non-standard model of arithmetic.

Soundness and completeness

A Σ^+ -structure \mathcal{A} is arithmetical if the interpretations in \mathcal{A} of the arithmetical operations and predicates restricted to those elements $n \in |\mathcal{A}|$ for which nat(n) holds in \mathcal{A} form the standard model of arithmetic.

Fact: If A is arithmetical then

if
$$\mathcal{TH}(\mathcal{A}) \vdash [\varphi] S[\psi]$$
 then $\models_{\mathcal{A}} [\varphi] S[\psi]$ *Soundness*

If moreover, finite sequences of elements in $|\mathcal{A}|$ can be encoded using a formula as a single element in $|\mathcal{A}|$, then

$$\mathcal{TH}(\mathcal{A}) \vdash [\varphi] S [\psi] \quad \textit{iff} \quad \models_{\mathcal{A}} [\varphi] S [\psi]$$

