Deductive Verification of Programs

6.0 VU Formal Methods in Computer Science

Gernot Salzer

AB Theoretische Informatik und Logik
Institut für Computersprachen

25 November 2013

Organisational issues

Lecture

- Today is the last day . . . of block 3.
- Helmut Veith (block 4) starts on Wednesday, Nov 27!

Exercises block 3

- Discussion of exercises: Tuesday, Nov 26, 17:00–19:00, EI 9
- Submission of solutions: Sunday, Dec 1
  On a voluntary basis, no extra points
- Discussion of solutions: Monday, Dec 9, 13:00–15:00, EI 10

Questions regarding block 3

- Email: gernot.salzer@tuwien.ac.at
- Office hours: www.logic.at/staff/salzer

Topics last time

1. Why formal methods?
2. Syntax of TPL(toy programming language)
3. Operational Semantics of TPL
4. Correctness assertions
5. How to prove correctness assertions
6. Assignments
7. Sequential Composition
8. Annotation calculus
9. Conditionals
10. Abort statements

Three ways to prove correctness assertions

Task: Show that $\{ F \}\, p \,\{ G \}$ is partially/totally correct.

Method 1: Hoare calculus.
Decompose correctness assertion into simpler ones (guided by rules) until we obtain true assertions (instances of axioms) and valid formulas.

Method 2: Weakest (liberal) precondition.
Compute the weakest formula F' such that $\{ F' \}\, p \,\{ G \}$ is true, and show that F implies F'.

Method 3: Strongest postcondition.
Compute the strongest formula G' such that $\{ F \}\, p \,\{ G' \}$ is true, and show that G' implies G.

Annotation calculus.
No new method, just combines the above methods for practical usage.

Hoare Calculus

- finite collection of axioms and rules for deriving correctness assertions
- two calculi: one for partial correctness, one for total correctness
- is sound and complete (with restrictions)

"Sound" means: Every derivable correctness assertion is true.
"Complete" means: Every true correctness assertion is derivable.

Admissible (sound) axiom

Let F and G be formula schemas and p be a program schema.
$\{ F \}\, p \,\{ G \}$ is an admissible axiom if all assertions of this form are true.

Admissible (sound) rule

Let $X_1, \ldots, X_n$ be schemas for formulas or correctness assertions.

$\frac{X_1 \cdots X_n}{\{ F \}\, p \,\{ G \}}$ is an admissible rule if for all valid/true instances
$\hat{X}_1, \ldots, \hat{X}_n, \hat{F}, \hat{p}, \hat{G}$ it holds that:

If $\hat{X}_1, \ldots, \hat{X}_n$ are valid/true then $\{ \hat{F} \}\, \hat{p} \,\{ \hat{G} \}$ is true.

"Whenever the premises hold, the conclusion has to hold."

Weakest Precondition

$\mathrm{wp}(p, \mathcal{S}_{\mathrm{out}})$ ... maximal set of states such that
$\mathrm{wp}(p, \mathcal{S}_{\mathrm{out}})\ p\ \mathcal{S}_{\mathrm{out}}$ is totally correct

$$\begin{aligned}\mathrm{wp}(p, \mathcal{S}_{\mathrm{out}}) &= \{\, \sigma \in \mathcal{S} \mid [p]\,\sigma \text{ defined and } [p]\,\sigma \in \mathcal{S}_{\mathrm{out}} \,\} \\ &= [p]^{-1}(\mathcal{S}_{\mathrm{out}})\end{aligned}$$

$\mathrm{wp}(p,\ G\ )$ ... weakest formula such that
$\{\, \mathrm{wp}(p, G) \,\}\ p\ \{\, G \,\}$ is totally correct

$\{\, F \,\}\, p\, \{\, G \,\}$ is totally correct if and only if $F \Rightarrow \mathrm{wp}(p, G)$ is valid.

Weakest Liberal Precondition

$\mathsf{wlp}(p, \mathcal{S}_{\text{out}})$ ... maximal set of states such that
$\mathsf{wlp}(p, \mathcal{S}_{\text{out}})\ p\ \mathcal{S}_{\text{out}}$ is partially correct

$$\begin{aligned}
\mathsf{wlp}(p, \mathcal{S}_{\text{out}}) &= \{\, \sigma \in \mathcal{S} \mid [p]\,\sigma \text{ undefined or } [p]\,\sigma \in \mathcal{S}_{\text{out}} \,\} \\
&= \{\, \sigma \in \mathcal{S} \mid [p]\,\sigma \text{ undefined} \,\} \cup [p]^{-1}(\mathcal{S}_{\text{out}}) \\
&= \{\, \sigma \in \mathcal{S} \mid [p]\,\sigma \text{ undefined} \,\} \cup \mathsf{wp}(p, \mathcal{S}_{\text{out}})
\end{aligned}$$

$\mathsf{wlp}(p,\ G\ )$ ... weakest formula such that
$\{\, \mathsf{wlp}(p, G) \,\}\ p\ \{\, G \,\}$ is partially correct

$\{\, F \,\}\, p\, \{\, G \,\}$ is partially correct if and only if $F \Rightarrow \mathsf{wlp}(p, G)$ is valid.

Strongest Postcondition

$\mathrm{sp}(\mathcal{S}_{\mathsf{in}}, p)$ ... minimal set of states such that
$\mathcal{S}_{\mathsf{in}}\ p\ \mathrm{sp}(\mathcal{S}_{\mathsf{in}}, p)$ is partially correct

$$\begin{aligned}\mathrm{sp}(\mathcal{S}_{\mathsf{in}}, p) &= \{\, [p]\,\sigma \mid \sigma \in \mathcal{S}_{\mathsf{in}} \text{ and } [p]\,\sigma \text{ defined} \,\} \\ &= [p](\mathcal{S}_{\mathsf{in}})\end{aligned}$$

$\mathrm{sp}(\ F\ , p)$ ... strongest formula such that
$\{\ F\ \}\ p\ \{\ \mathrm{sp}(F, p)\ \}$ is partially correct

$\{\ F\ \}\ p\ \{\ G\ \}$ is partially correct if and only if $\mathrm{sp}(F, p) \Rightarrow G$ is valid.

Weakest precondition

$\mathrm{wp}(\textbf{skip}, G) = G$

$\mathrm{wp}(v := e, G) = G[v/e]$

$\mathrm{wp}(p; q, G) = \mathrm{wp}(p, \mathrm{wp}(q, G))$

$\mathrm{wp}(\textbf{if } e \textbf{ then } p \textbf{ else } q \textbf{ fi}, G)$

$$= (\ \ e \Rightarrow \mathrm{wp}(p, G)) \wedge (\neg e \Rightarrow \mathrm{wp}(q, G))$$

$$= (\ \ e \wedge \mathrm{wp}(p, G)) \vee (\neg e \wedge \mathrm{wp}(q, G))$$

$\mathrm{wp}(\textbf{abort}, G) = \mathrm{false}$

Weakest liberal precondition

$\mathrm{wlp}(\textbf{skip}, G) = G$

$\mathrm{wlp}(v := e, G) = G[v/e]$

$\mathrm{wlp}(p; q, G) = \mathrm{wlp}(p, \mathrm{wlp}(q, G))$

$\mathrm{wlp}(\textbf{if } e \textbf{ then } p \textbf{ else } q \textbf{ fi}, G)$

$$= (\ \ e \Rightarrow \mathrm{wlp}(p, G)) \wedge (\neg e \Rightarrow \mathrm{wlp}(q, G))$$

$$= (\ \ e \wedge \mathrm{wlp}(p, G)) \vee (\neg e \wedge \mathrm{wlp}(q, G))$$

$\mathrm{wlp}(\textbf{abort}, G) = \mathrm{true}$

Strongest postcondition

$\mathrm{sp}(F, \textbf{skip}) = F$

$\mathrm{sp}(F, v := e) = \exists v' \, (F[v/v'] \wedge v = e[v/v'])$

where v' does not occur in F and e

$\mathrm{sp}(F, p; q) = \mathrm{sp}(\mathrm{sp}(F, p), q)$

$\mathrm{sp}(F, \textbf{if } e \textbf{ then } p \textbf{ else } q \textbf{ fi}) = \mathrm{sp}(F \wedge e, \, p) \vee \mathrm{sp}(F \wedge \neg e, \, q)$

$\mathrm{sp}(F, \textbf{abort}) = \mathrm{false}$

Hoare calculus

$$\{\, F \,\}\ \mathbf{skip}\ \{\, F \,\} \quad \text{(sk)}$$

$$\{\, F[v/e] \,\}\ v := e\ \{\, F \,\} \quad \text{(as)}$$

$$\frac{\{\, F \,\}\ p\ \{\, G \,\} \quad \{\, G \,\}\ q\ \{\, H \,\}}{\{\, F \,\}\ p; q\ \{\, H \,\}} \quad \text{(sc)}$$

$$\frac{\{\, F \wedge e \,\}\ p\ \{\, G \,\} \quad \{\, F \wedge \neg e \,\}\ q\ \{\, G \,\}}{\{\, F \,\}\ \mathbf{if}\ e\ \mathbf{then}\ p\ \mathbf{else}\ q\ \mathbf{fi}\ \{\, G \,\}} \quad \text{(if)}$$

$$\{\, F \,\}\ \mathbf{abort}\ \{\, G \,\} \quad \text{(ab)} \qquad \text{(partial correctness)}$$

$$\{\, \text{false} \,\}\ \mathbf{abort}\ \{\, G \,\} \quad \text{(abt)} \qquad \text{(total correctness)}$$

$$\frac{F \Rightarrow F' \quad \{\, F' \,\}\ p\ \{\, G' \,\} \quad G' \Rightarrow G}{\{\, F \,\}\ p\ \{\, G \,\}} \quad \text{(lc)}$$

Annotation rules

$$\begin{array}{lll} & \{F\} & \text{sk}\uparrow \\ \textbf{skip} & \mapsto & \textbf{skip} \\ \{F\} & & \{F\} \end{array} \qquad \begin{array}{lll} \{F\} & & \{F\} \\ \textbf{skip} & \mapsto & \textbf{skip} \\ & & \{F\} \quad \text{sk}\downarrow \end{array}$$

$$\begin{array}{lll} & \{F[v/e]\} & \text{as}\uparrow \\ v := e & \mapsto & v := e \\ \{F\} & & \{F\} \end{array} \qquad \begin{array}{lll} \{F\} & & \{F\} \\ v := e & \mapsto & v := e \\ & & \{\exists v'(F[v/v'] \wedge v = e[v/v'])\} \quad \text{as}\downarrow \end{array}$$

if v not free in F and e: $\{ \quad F \wedge v = e \quad \}$ $\text{as}'\downarrow$

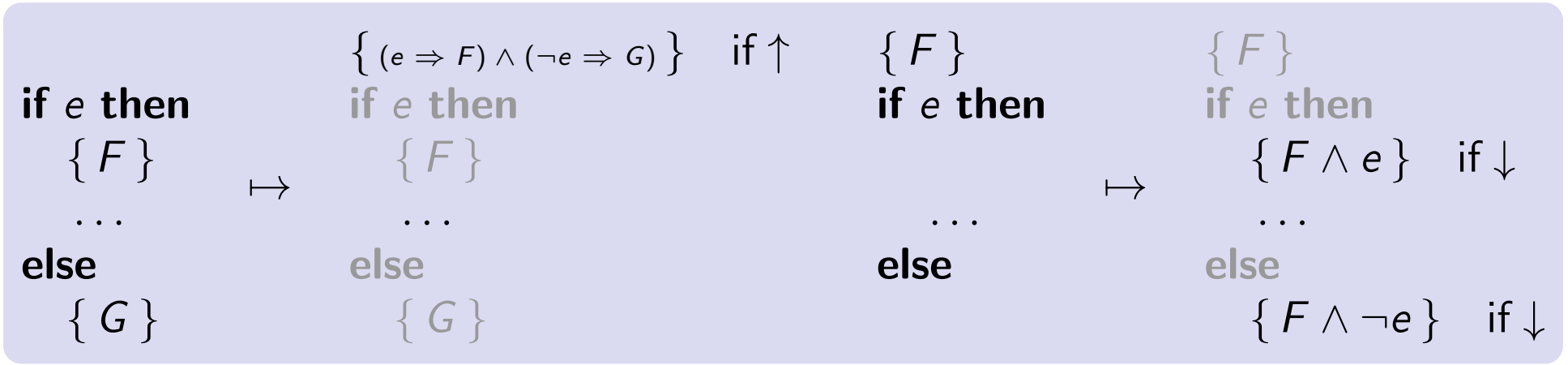
Partial correctness:

$$\begin{array}{lll} & \{\text{true}\} & \text{ab} \\ \textbf{abort} & \mapsto & \textbf{abort} \\ & \{\text{false}\} & \text{ab} \end{array}$$

Total correctness:

$$\begin{array}{lll} & \{\text{false}\} & \text{abt} \\ \textbf{abort} & \mapsto & \textbf{abort} \\ & \{\text{false}\} & \text{abt} \end{array}$$

$$\begin{array}{l} \{F\} \\ \{G\} \end{array} \quad \mapsto \quad \begin{array}{l} \{F\} \\ \{G\} \end{array} \qquad \text{Prove } F \Rightarrow G \quad \text{lc}$$


$\{(e \Rightarrow F) \wedge (\neg e \Rightarrow G)\}$ if ↑
if e then
$\{F\}$
...
else
$\{G\}$
↦
if e then
$\{F\}$
...
else
$\{G\}$
$\{F\}$
if e then
...
else
↦
$\{F\}$
if e then
$\{F \wedge e\}$ if ↓
...
else
$\{F \wedge \neg e\}$ if ↓


$\{G\}$ fi ↑
else
...
fi
$\{G\}$
↦
...
$\{G\}$ fi ↑
fi
$\{G\}$
$\{G_1\}$
else
...
$\{G_2\}$
fi
↦
$\{G_1\}$
else
...
$\{G_2\}$
fi
$\{G_1 \vee G_2\}$ fi ↓

Topics today

1. Why formal methods?
2. Syntax of TPL(toy programming language)
3. Operational Semantics of TPL
4. Correctness assertions
5. How to prove correctness assertions
6. Assignments
7. Sequential Composition
8. Annotation calculus
9. Conditionals
10. Abort statements
11. Some more examples
12. Loops
13. Live demo

$\mathsf{wp}(z := z + x; y := y - 1,\ xy + z = c)$

$$
\begin{aligned}
&= \mathsf{wp}(z := z + x,\ \mathsf{wp}(y := y - 1,\ xy + z = c)) \\
&= \mathsf{wp}(z := z + x,\ x(y - 1) + z = c) \\
&= (x(y - 1) + (z + x) = c) \\
&= (xy + z = c)
\end{aligned}
$$

$\mathsf{sp}(xy + z = c,\ z := z + x; y := y - 1)$

$$
\begin{aligned}
&= \mathsf{sp}(\mathsf{sp}(xy + z = c,\ z := z + x),\ y := y - 1) \\
&= \mathsf{sp}(\exists z'(xy + z' = c \wedge z = z' + x),\ y := y - 1) \\
&= \mathsf{sp}(\exists z'(xy + (z - x) = c \wedge z - x = z'),\ y := y - 1) \\
&= \mathsf{sp}(xy + (z - x) = c \wedge \exists z'\ z - x = z',\ y := y - 1) \\
&= \mathsf{sp}(xy + (z - x) = c,\ y := y - 1) \\
&= \exists y'(xy' + (z - x) = c \wedge y = y' - 1) \\
&= \exists y'(x(y + 1) + (z - x) = c \wedge y + 1 = y') \\
&= x(y + 1) + (z - x) = c \wedge \exists y'\ y + 1 = y' \\
&= (xy + z = c)
\end{aligned}
$$

$\mathrm{wp}(d := (a{+}b)/2; \textbf{if } dm{\leq}n \textbf{ then } a := d \textbf{ else } b := d \textbf{ fi},\ am{\leq}n{<}bm)$

$$
\begin{aligned}
&= \mathrm{wp}(d := a{+}b/2,\\
&\qquad\qquad \mathrm{wp}(\textbf{if } dm \leq n \textbf{ then } a := d \textbf{ else } b := d \textbf{ fi},\ am \leq n < bm))\\
&= \mathrm{wp}(d := a{+}b/2,\ \ (dm \leq n \wedge \mathrm{wp}(a := d, am \leq n < bm))\\
&\qquad\qquad \vee\ (dm \not\leq n \wedge \mathrm{wp}(b := d, am \leq n < bm)))\\
&= \mathrm{wp}(d := a{+}b/2,\ \ (dm \leq n \wedge dm \leq n < bm)\\
&\qquad\qquad \vee\ (dm > n \wedge am \leq n < dm))\\
&= \mathrm{wp}(d := a{+}b/2,\ dm \leq n < bm \vee am \leq n < dm)\\
&= ((a{+}b)/2)m \leq n < bm \quad \vee \quad am \leq n < ((a{+}b)/2)m
\end{aligned}
$$

$\{\ am{\leq}n{<}bm\ \}\ \ d := (a{+}b)/2; \textbf{if } dm{\leq}n \textbf{ then } \cdots \textbf{ fi}\ \{\ am{\leq}n{<}bm\ \}$

$$
\begin{aligned}
&am \leq n < bm \ \Rightarrow\ \mathrm{wp}(d := (a{+}b)/2; \textbf{if } dm{\leq}n \textbf{ then } \cdots \textbf{ fi},\ am{\leq}n{<}bm)\\
&am \leq n < bm \ \Rightarrow\ ((a{+}b)/2)m \leq n < bm \ \vee\ am \leq n < ((a{+}b)/2)m)\\
&am \leq n < bm \ \Rightarrow\ ((a{+}b)/2)m \leq n \ \vee\ n < ((a{+}b)/2)m)\\
&am \leq n < bm \ \Rightarrow\ \mathrm{true}\\
&\mathrm{true}
\end{aligned}
$$

$\mathrm{sp}(am{\le}n{<}bm,\ d := (a{+}b)/2; \textbf{if } dm{\le}n \textbf{ then } a := d \textbf{ else } b := d \textbf{ fi})$

$$
\begin{aligned}
&= \mathrm{sp}(\mathrm{sp}(am \le n < bm,\ d := (a{+}b)/2),\\
&\qquad\qquad \textbf{if } dm \le n \textbf{ then } a := d \textbf{ else } b := d \textbf{ fi})\\
&= \mathrm{sp}(am \le n < bm \wedge d = (a{+}b)/2,\\
&\qquad\qquad \textbf{if } dm \le n \textbf{ then } a := d \textbf{ else } b := d \textbf{ fi})\\
&= \mathrm{sp}(am \le n < bm \wedge d = (a{+}b)/2 \wedge dm \le n,\ a := d)\\
&\vee\ \mathrm{sp}(am \le n < bm \wedge d = (a{+}b)/2 \wedge dm > n,\ b := d)\\
&= \exists a' \,(a'm \le n < bm \wedge d = (a'{+}b)/2 \wedge dm \le n \wedge a = d)\\
&\vee\ \exists b' \,(am \le n < b'm \wedge d = (a{+}b')/2 \wedge dm > n \wedge b = d)\\
&= dm \le n < bm \wedge a = d \wedge \exists a' \,(a'm \le n \wedge d = (a'{+}b)/2)\\
&\vee\ am \le n < dm \wedge b = d \wedge \exists b' \,(n < b'm \wedge d = (a{+}b')/2)\\
&= am \le n < bm \wedge (\quad (a = d \wedge \exists a' \,(a'm \le n \wedge d = (a'{+}b)/2))\\
&\qquad\qquad \vee\ (b = d \wedge \exists b' \,(n < b'm \wedge d = (a{+}b')/2)))
\end{aligned}
$$

$\{\, am{\le}n{<}bm \,\}\ d := (a{+}b)/2; \textbf{if } dm{\le}n \textbf{ then } \cdots \textbf{ fi}\ \{\, am{\le}n{<}bm \,\}$

$\mathrm{sp}(am{\le}n{<}bm,\ d := (a{+}b)/2; \textbf{if } dm{\le}n \textbf{ then } \cdots \textbf{ fi}) \ \Rightarrow\ am \le n < bm$

$am \le n < bm \wedge \cdots \ \Rightarrow\ am \le n < bm$

true

$\{\, xy + z = c \,\}\, z := z + x; y := y - 1 \,\{\, xy + z = c \,\}$

$$H = x(y-1) + (z+x) = c \qquad\qquad G = x(y-1) + z = c$$

$$\dfrac{\dfrac{xy{+}z = c \Rightarrow H \quad (1) \qquad \{\, H \,\}\, z := z{+}x \,\{\, G \,\} \quad (\mathsf{as})}{\{\, xy{+}z = c \,\}\, z := z{+}x \,\{\, G \,\}}\,(\mathsf{lc}) \qquad \{\, G \,\}\, y := y{-}1 \,\{\, xy{+}z = c \,\} \quad (\mathsf{as})}{\{\, xy + z = c \,\}\, z := z + x; y := y - 1 \,\{\, xy + z = c \,\}}\,(\mathsf{sc})$$

Prove validity of ...

(1) $xy{+}z = c \Rightarrow x(y-1) + (z+x) = c$

$\{\, am \le n < bm \,\}\ d := (a+b)/2; \textbf{if } dm \le n \textbf{ then } \cdots \textbf{ fi } \{\, am \le n < bm \,\}$

$$F_a = F[a/d] \qquad\qquad F_b = F[b/d]$$

$$\dfrac{\dfrac{G \wedge dm \le n \Rightarrow F_a \quad \{\, F_a \,\}\, a := d \,\{\, F \,\}}{\{\, G \wedge dm \le n \,\}\, a := d \,\{\, F \,\}}\ (\mathrm{lc}) \qquad \dfrac{G \wedge dm \not\le n \Rightarrow F_b \quad \{\, F_b \,\}\, b := d \,\{\, F \,\}}{\{\, G \wedge dm \not\le n \,\}\, b := d \,\{\, F \,\}}\ (\mathrm{lc})}{\{\, G \,\}\, \textbf{if } dm \le n \textbf{ then } a := d \textbf{ else } b := d \textbf{ fi} \,\{\, F \,\}}\ (\mathrm{if})$$

(1) (as) (2) (as) (3) (as)

$$F = am \le n < bm \qquad H = G[d/\tfrac{a+b}{2}] \qquad G = F \wedge d = \tfrac{a+b}{2}$$

$$\dfrac{\dfrac{F \Rightarrow H \quad \{\, H \,\}\, d := \frac{a+b}{2} \,\{\, G \,\}}{\{\, F \,\}\, d := \frac{a+b}{2} \,\{\, G \,\}}\ (\mathrm{lc}) \qquad \{\, G \,\}\, \textbf{if } dm \le n \textbf{ then } \cdots \textbf{ fi} \,\{\, F \,\}}{\{\, F \,\}\, d := \frac{a+b}{2}; \textbf{if } dm \le n \textbf{ then } a := d \textbf{ else } b := d \textbf{ fi} \,\{\, F \,\}}\ (\mathrm{sc})$$

Prove validity of ...

(1) $F \Rightarrow G[d/\frac{a+b}{2}]$

(2) $(G \wedge dm \le n) \Rightarrow F[a/d]$

(3) $(G \wedge dm \not\le n) \Rightarrow F[b/d]$

$\{\, am \le n < bm \,\}\ d := (a+b)/2; \textbf{if } dm \le n \textbf{ then } \cdots \textbf{ fi } \{\, am \le n < bm \,\}$

$\{\, F : am \le n < bm \,\}$
$d := (a+b)/2;$
$\{\, 5 : F \wedge d = (a+b)/2 \,\}$ (as′↓)
if $dm \le n$ **then**
  $\{\, 6 : F \wedge d = (a+b)/2 \wedge dm \le n \,\}$ (if↓)
  $\{\, 3 : F[a/d] \,\}$ (as↑)
  $a := d$
  $\{\, 2 : F \,\}$ (fi↑)
else
  $\{\, 7 : F \wedge d = (a+b)/2 \wedge dm \not\le n \,\}$ (if↓)
  $\{\, 4 : F[b/d] \,\}$ (as↑)
  $b := d$
  $\{\, 1 : F \,\}$ (fi↑)
fi
$\{\, F : am \le n < bm \,\}$

Prove validity of ...

(6) ⇒ (3): $(F \wedge d = (a+b)/2 \wedge dm \le n) \Rightarrow F[a/d]$

(7) ⇒ (4): $(F \wedge d = (a+b)/2 \wedge dm \not\le n) \Rightarrow F[b/d]$

Topics today

1. Why formal methods?
2. Syntax of TPL(toy programming language)
3. Operational Semantics of TPL
4. Correctness assertions
5. How to prove correctness assertions
6. Assignments
7. Sequential Composition
8. Annotation calculus
9. Conditionals
10. Abort statements
11. Some more examples
12. Loops
13. Live demo

Verifying loops: Weakest preconditions

wp(**while** e **do** p **od**, G): All states such that the loop terminates after a finite number of iterations in a G-state.

$\{ F_i \}$ ... set of states such that p executes i times and leads to G-state

0 iterations: $F_0 = \neg e \wedge G$

1 iteration: $F_1 = e \wedge \mathrm{wp}(p, F_0)$

2 iterations: $F_2 = e \wedge \mathrm{wp}(p, F_1)$

$\vdots \qquad \vdots$

i iterations: $F_i = e \wedge \mathrm{wp}(p, F_{i-1}) \quad (\text{for } i > 0)$

$\{ F_i \} = \{ e \wedge \mathrm{wp}(p, F_{i-1}) \}$ ... set of states such that

- p is executed once (because e is true), resulting in a state where
- $i - 1$ further iterations will lead to a G-state.

$$\mathrm{wp}(\textbf{while } e \textbf{ do } p \textbf{ od}, G) = F_0 \vee F_1 \vee F_2 \vee \cdots = \exists i \, (i \geq 0 \wedge F_i)$$

$\mathrm{wp}(\textbf{while } e \textbf{ do } p \textbf{ od}, G) = \exists i\,(i \geq 0 \wedge F_i)$

$$F_0 = \neg e \wedge G$$
$$F_{i+1} = e \wedge \mathrm{wp}(p, F_i)$$

$\mathrm{wp}(\textbf{while } y \neq 0 \textbf{ do } z := z + x; y := y - 1 \textbf{ od}, z = xy_0)$

$F_0 = (y = 0 \wedge z = xy_0)$

$$\begin{aligned} F_1 &= (y \neq 0 \wedge \mathrm{wp}(z := z + x; y := y - 1, F_0)) \\ &= (y \neq 0 \wedge y{-}1 = 0 \wedge z{+}x = xy_0) \\ &= (y \neq 0 \wedge y = 1 \wedge z{+}x = xy_0) \\ &= (y = 1 \wedge z = x(y_0 - 1)) \end{aligned}$$
$F_1 = (y = 1 \wedge z = x(y_0 - 1))$

$$\begin{aligned} F_2 &= (y \neq 0 \wedge \mathrm{wp}(z := z + x; y := y - 1, F_1)) \\ &= (y \neq 0 \wedge y{-}1 = 1 \wedge z{+}x = x(y_0 - 1)) \\ &= (y \neq 0 \wedge y = 2 \wedge z{+}x = x(y_0 - 1)) \\ &= (y = 2 \wedge z = x(y_0 - 2)) \end{aligned}$$
$F_2 = (y = 2 \wedge z = x(y_0 - 2))$

Guess: $F_i = (y = i \wedge z = x(y_0 - i))$ for all $i \geq 0$

$\mathrm{wp}(\textbf{while } e \textbf{ do } p \textbf{ od}, G) = \exists i\,(i \geq 0 \wedge F_i)$

$F_0 = \neg e \wedge G$

$F_{i+1} = e \wedge \mathrm{wp}(p, F_i)$

$\mathrm{wp}(\textbf{while } y \neq 0 \textbf{ do } z := z + x; y := y - 1 \textbf{ od}, z = xy_0)$

$F_0 = (y = 0 \wedge z = xy_0)$

$F_1 = (y = 1 \wedge z = x(y_0 - 1))$

$F_2 = (y = 2 \wedge z = x(y_0 - 2))$

$F_i = (y = i \wedge z = x(y_0 - i))$ for all $i \geq 0$

Inductive proof:

$F_0 = (y = 0 \wedge z = x(y_0 - 0)) = (y = 0 \wedge z = xy_0)$ base case ✓

Inductive step: prove equation for $i + 1$ (assuming it holds for i)

$$\begin{aligned} F_{i+1} &= (y \neq 0 \wedge \mathrm{wp}(z := z + x; y := y - 1, F_i)) \\ &= (y \neq 0 \wedge y{-}1 = i \wedge z{+}x = x(y_0 - i)) \\ &= (y \neq 0 \wedge y = i{+}1 \wedge z = x(y_0 - i) - x) \\ &= (y \neq 0 \wedge y = i{+}1 \wedge z = x(y_0 - (i + 1))) \\ &= (y = i{+}1 \wedge z = x(y_0 - (i + 1))) \quad \checkmark \end{aligned}$$

$\mathrm{wp}(\textbf{while } e \textbf{ do } p \textbf{ od}, G) = \exists i\,(i \geq 0 \wedge F_i)$

$$F_0 = \neg e \wedge G$$
$$F_{i+1} = e \wedge \mathrm{wp}(p, F_i)$$

$\mathrm{wp}(\textbf{while } y \neq 0 \textbf{ do } z := z + x; y := y - 1 \textbf{ od}, z = xy_0)$

$F_0 = (y = 0 \wedge z = xy_0)$

$F_1 = (y = 1 \wedge z = x(y_0 - 1))$

$F_2 = (y = 2 \wedge z = x(y_0 - 2))$

$F_i = (y = i \wedge z = x(y_0 - i))$ for all $i \geq 0$ proved ✓

$$\begin{aligned}
\mathrm{wp}(\textbf{while } \cdots, z = xy_0) &= \exists i\,(i \geq 0 \wedge F_i) \\
&= \exists i\,(i \geq 0 \wedge y = i \wedge z = x(y_0 - i)) \\
&= \exists i\,(y \geq 0 \wedge y = i \wedge z = x(y_0 - y)) \\
&= y \geq 0 \wedge \exists i\,(y = i) \wedge z = x(y_0 - y) \\
&= y \geq 0 \wedge \mathrm{true} \wedge z = x(y_0 - y) \\
&= y \geq 0 \wedge z = x(y_0 - y)
\end{aligned}$$

$\mathrm{wp}(z := 0;\ \textbf{while}\ y \neq 0\ \textbf{do}\ z := z + x; y := y - 1\ \textbf{od},\ z = xy_0)$

$= \mathrm{wp}(z := 0,\ \mathrm{wp}(\textbf{while}\ y \neq 0\ \textbf{do}\ z := z + x; y := y - 1\ \textbf{od},\ z = xy_0))$
$= \mathrm{wp}(z := 0,\ y \geq 0 \wedge z = x(y_0 - y))$
$= y \geq 0 \wedge 0 = x(y_0 - y)$
$= y \geq 0 \wedge (x = 0 \vee y_0 = y)$

$\{\, y \geq 0 \wedge y = y_0 \,\}\, z := 0;\ \textbf{while}\ y \neq 0\ \textbf{do}\ \cdots\ \textbf{od}\, \{\, z = xy_0 \,\}$

. . . is totally correct, since the following implication is valid:

$(y \geq 0 \wedge y = y_0) \Rightarrow \mathrm{wp}(z := 0;\ \textbf{while}\ y \neq 0\ \textbf{do}\ \cdots\ \textbf{od},\ z = xy_0)$
$(y \geq 0 \wedge y = y_0) \Rightarrow (y \geq 0 \wedge (x = 0 \vee y_0 = y))$

- $y \geq 0$ in the conclusion holds because it also occurs in the premise.
- $x = 0 \vee y_0 = y$ holds because $y = y_0$ occurs in the premise.

First correctness proof
of our multiplication program!

Verifying loops: Hoare calculus

$\mathrm{wp}(\textbf{while } e \textbf{ do } p \textbf{ od}, G) = F_0 \vee F_1 \vee F_2 \vee \cdots = \exists i\,(i \geq 0 \wedge F_i)$,
where $F_0 = \neg e \wedge G$
$F_{i+1} = e \wedge \mathrm{wp}(p, F_i) \quad (\text{for } i \geq 0)$

This definition mirrors the operational semantics: each iteration contributes a partial precondition, leading to an infinite formula.

Difficulty with loops: Situation changes with every iteration.

Does it really? Some things never change.

Loop of multiplication program:
$\mathrm{wp}(z := z + x; y := y - 1, xy + z = c) = (xy + z = c)$

Idea: Concentrate on invariant properties.

Loop unwinding and invariants

$\{ F \}$
while e **do** p **od**
$\{ G \}$

Loop unwinding and invariants

$\{ F \}$
$\{ e \}$
p;
$\{ e \}$
p;
$\{ e \}$
$\vdots$
$\{ e \}$
p;
$\{ \neg e \}$
$\{ G \}$

If $\neg e \Rightarrow G$, the number of iterations is irrelevant.
Partial correctness guaranteed!

Loop unwinding and invariants

$\{ F \}$
$\{ e \wedge \mathrm{wp}(p, F_{i-1}) =: F_i \}$
$p;$
$\{ e \wedge \mathrm{wp}(p, F_{i-2}) =: F_{i-1} \}$
$p;$
$\{ e \wedge \mathrm{wp}(p, F_{i-3}) =: F_{i-2} \}$
$\vdots$
$\{ e \wedge \mathrm{wp}(p, F_0) =: F_1 \}$
$p;$
$\{ \neg e \wedge G =: F_0 \}$
$\{ G \}$

Otherwise: Compute wp.
Prove $F \Rightarrow F_i$ for some F_i.
Problem: We don't know i.
Trying all i not feasible.

Loop unwinding and invariants

$\{ F \}$
$\{ e \wedge Inv \}$
p;
$\{ e \wedge Inv \}$
p;
$\{ e \wedge Inv \}$
$\vdots$
$\{ e \wedge Inv \}$
p;
$\{ \neg e \wedge Inv \}$
$\{ G \}$

The number of iterations is irrelevant.
Partial correctness guaranteed!

Idea: Find formula Inv such that

- $\{ e \wedge Inv \}$ p $\{ Inv \}$ is correct
- $F \Rightarrow Inv$ is valid
- $\neg e \wedge Inv \Rightarrow G$ is valid

Inv ... "invariant"

Loops and Partial Correctness

Hoare calculus:

$$\frac{\{\, \mathit{Inv} \wedge e \,\}\, p \,\{\, \mathit{Inv} \,\}}{\{\, \mathit{Inv} \,\}\, \textbf{while}\ e\ \textbf{do}\ p\ \textbf{od}\, \{\, \mathit{Inv} \wedge \neg e \,\}}\ \text{(wh)}$$

Annotation calculus:

$$\textbf{while}\ e\ \textbf{do} \cdots \textbf{od} \mapsto \{\mathit{Inv}\}\, \textbf{while}\ e\ \textbf{do}\, \{\mathit{Inv} \wedge e\} \cdots \{\mathit{Inv}\}\, \textbf{od}\, \{\mathit{Inv} \wedge \neg e\} \qquad \text{wh}$$

Inv: "loop invariant"; holds before, within, and after the loop.

Example: Multiplication

$\{\, 1 \colon y = y_0 \,\}$
$\{\, 9 \colon \mathit{Inv}[z/0] \,\}$
$z := 0;$
$\{\, 3 \colon \mathit{Inv} \,\}$
while $y \neq 0$ **do**
  $\{\, 4 \colon \mathit{Inv} \wedge y \neq 0 \,\}$
  $\{\, 8 \colon \mathit{Inv}[y/y{-}1][z/z{+}x] \,\}$
  $z := z + x;$
  $\{\, 7 \colon \mathit{Inv}[y/y{-}1] \,\}$
  $y := y - 1$
  $\{\, 5 \colon \mathit{Inv} \,\}$
od
$\{\, 6 \colon \mathit{Inv} \wedge \neg(y \neq 0) \,\}$
$\{\, 2 \colon z = x * y_0 \,\}$

It remains to ...

- find adequate invariant Inv
- prove $1 \Rightarrow 9$, $4 \Rightarrow 8$, $6 \Rightarrow 2$

We know from before:

$\{\, xy + z = c \,\}$
$z := z + x;\ y := y - 1$
$\{\, xy + z = c \,\}$

is correct.

Guess:
$\mathit{Inv} = (xy + z = c)$
(for some suitable c)

Example: Multiplication (2)

$1 \Rightarrow 9$: $y = y_0 \Rightarrow \mathit{Inv}[z/0]$

$y = y_0 \Rightarrow (xy + 0 = c)$

Valid if we choose $c = xy_0$.

$4 \Rightarrow 8$: $(\mathit{Inv} \wedge y \neq 0) \Rightarrow \mathit{Inv}[y/y{-}1][z/z{+}x]$

$(\mathit{Inv} \wedge y \neq 0) \Rightarrow (xy + z = xy_0)[y/y{-}1][z/z{+}x]$

$(\mathit{Inv} \wedge y \neq 0) \Rightarrow (x(y{-}1) + (z{+}x) = xy_0)$

$(\mathit{Inv} \wedge y \neq 0) \Rightarrow (xy + z = xy_0)$

Valid.

$6 \Rightarrow 2$: $(\mathit{Inv} \wedge \neg(y \neq 0)) \Rightarrow z = xy_0$

$(xy + z = xy_0 \wedge y = 0) \Rightarrow z = xy_0$

$(0 + z = xy_0 \wedge y = 0) \Rightarrow z = xy_0$

Valid.

Second correctness proof:
partial correctness via invariants.

Termination

while $y \neq 0$ **do**
  $z := z + x;$
  $y := y - 1$
od

Why does this loop terminate?

- Loop condition is false in states σ where $\sigma(y) = 0$.
- Loop body decreases value of y by one.
- Hence loop condition will break the loop if y is non-negative initially.

In general:

- Find a way to measure the "size" of the current state. $|\sigma| = \sigma(y)$
- Show that the size decreases in each iteration in discrete steps.
- Show that the size is bounded from below. $|\sigma| \geq 0$

Then the loop is guaranteed to terminate.

Loops and Total Correctness

t: "bound function" or "variant", computes the size of state

The value of the expression t has to

- ... be an integer (ensures that t only takes discrete values),
- ... be non-negative before and within the loop,
- ... decrease strictly with each iteration of the loop.

$$\frac{\underbrace{\{\, Inv \wedge e \,\}\, p \,\{\, Inv \,\}}_{\text{partial correctness}} \quad \underbrace{\{\, Inv \wedge e \wedge t = t_0 \,\}\, p \,\{\, t < t_0 \,\}}_{\text{state size decreases strictly}} \quad \underbrace{Inv \Rightarrow t \geq 0}_{\text{and is bounded}}}{\{\, Inv \,\}\, \textbf{while}\ e\ \textbf{do}\ p\ \textbf{od}\, \{\, Inv \wedge \neg e \,\}}\ \text{(wht)}$$

t_0: auxiliary variable, representing the value of t before iteration

Alternative while-rules

$$\frac{\{\, Inv \wedge e \wedge t = t_0 \,\}\, p \,\{\, Inv \wedge t < t_0 \,\} \quad Inv \Rightarrow t \geq 0}{\{\, Inv \,\}\, \textbf{while}\ e\ \textbf{do}\ p\ \textbf{od}\, \{\, Inv \wedge \neg e \,\}}\ (\text{wht}')$$

$$\frac{\{\, Inv \wedge e \wedge t = t_0 \,\}\, p \,\{\, Inv \wedge 0 \leq t < t_0 \,\}}{\{\, Inv \,\}\, \textbf{while}\ e\ \textbf{do}\ p\ \textbf{od}\, \{\, Inv \wedge \neg e \,\}}\ (\text{wht}'')$$

$$\frac{\{\, Inv \wedge e \wedge t = t_0 \,\}\, p \,\{\, Inv \wedge (e \Rightarrow 0 \leq t < t_0) \,\}}{\{\, Inv \,\}\, \textbf{while}\ e\ \textbf{do}\ p\ \textbf{od}\, \{\, Inv \wedge \neg e \,\}}\ (\text{wht}''')$$

Annotation calculus:

$$\textbf{while}\ e\ \textbf{do} \cdots \textbf{od} \mapsto \{Inv\}\, \textbf{while}\ e\ \textbf{do}\, \{Inv \wedge e \wedge t = t_0\} \cdots \{Inv \wedge 0 \leq t < t_0\}\, \textbf{od}\, \{Inv \wedge \neg e\} \qquad \text{wht}''$$

$$\textbf{while}\ e\ \textbf{do} \cdots \textbf{od} \mapsto \{Inv\}\, \textbf{while}\ e\ \textbf{do}\, \{Inv \wedge e \wedge t = t_0\} \cdots \{Inv \wedge (e \Rightarrow 0 \leq t < t_0)\}\, \textbf{od}\, \{Inv \wedge \neg e\} \qquad \text{wht}'''$$

Example: Multiplication

$\{\, 1 \colon y = y_0 \,\}$
$\{\, 9 \colon \mathit{Inv}[z/0] \,\}$
$z := 0;$
$\{\, 3 \colon \mathit{Inv} \,\}$
while $y \neq 0$ **do**
  $\{\, 4 \colon \mathit{Inv} \wedge y \neq 0 \wedge t = t_0 \,\}$
  $\{\, 8 \colon (\mathit{Inv} \wedge 0 \leq t < t_0)[y/y-1][z/z+x] \,\}$
  $z := z + x;$
  $\{\, 7 \colon (\mathit{Inv} \wedge 0 \leq t < t_0)[y/y-1] \,\}$
  $y := y - 1$
  $\{\, 5 \colon \mathit{Inv} \wedge 0 \leq t < t_0 \,\}$
od
$\{\, 6 \colon \mathit{Inv} \wedge \neg(y \neq 0) \,\}$
$\{\, 2 \colon z = x * y_0 \,\}$

We choose:

- $\mathit{Inv} = (xy + z = xy_0)$
- $t = y$

It remains to prove $1 \Rightarrow 9$, $4 \Rightarrow 8$, and $6 \Rightarrow 2$.

$1 \Rightarrow 9$ and $6 \Rightarrow 2$: see above.

$4 \Rightarrow 8$ consists of two parts.

$4 \Rightarrow 8a$ (partial correctness):
$\mathit{Inv} \wedge y \neq 0 \quad \Rightarrow \mathit{Inv}[\,][\,]$
See above.

$4 \Rightarrow 8b$ (termination):
$\mathit{Inv} \wedge y \neq 0 \wedge t = t_0 \Rightarrow 0 \leq t[\,][\,] < t_0$
Not yet proved.

Example: Multiplication (2)

$4 \Rightarrow 8b$: $(Inv \wedge y \neq 0 \wedge t = t_0) \Rightarrow 0 \leq t[y/y{-}1][z/z{+}x] < t_0$

$(Inv \wedge y \neq 0 \wedge y = t_0) \Rightarrow 0 \leq y[y/y{-}1][z/z{+}x] < t_0$

$(Inv \wedge y \neq 0 \wedge y = t_0) \Rightarrow 0 \leq y{-}1 < t_0$

We have to show:

$(Inv \wedge y \neq 0 \wedge y = t_0) \Rightarrow y{-}1 < t_0$ ✓

$(Inv \wedge y \neq 0 \quad\quad\quad\quad) \Rightarrow 0 \leq y{-}1$ Not valid!

We have to strengthen ...

- the invariant: $Inv' = (Inv \wedge y \geq 0)$
- the precondition: $(1') = (y = y_0 \wedge y \geq 0)$

... and to reprove all implications (not done here).

$(Inv \wedge y \geq 0 \wedge y \neq 0) \Rightarrow 0 \leq y{-}1$ Now valid!

Third correctness proof:
total correctness via invariants and variants.