Introduction to Mathematical Logic

For CS Students

CS104

Yida TAO (陶伊达)

2025 年 5 月 27 日

Table of Contents

- To construct formal proofs of *partial correctness* specification, axioms and rules of inference are needed.
- This is what Hoare Logic provides:
  - The formulation of the deductive system is due to Hoare
  - Some of the underlying ideas originated with R. Floyd (also called Floyd–Hoare logic)
- A proof in Hoare logic is a sequence of lines, each of which is either an axiom of the logic or follows from earlier lines by a rule of inference of the logic
- A formal proof makes explicit what axioms and rules of inference are used to arrive at a conclusion.

Programs

1 Warm up

We can think of any program of our core programming language as a sequence. All of the C_i below are either assignments, if-statements or while-statements. Of course, we allow the if-statements and while-statements to have embedded compositions.

$$\begin{aligned} C_1; \\ C_2; \\ . \\ . \\ . \\ C_n \end{aligned}$$

Presentation of a proof

1 Warm up

We should design a proof calculus which presents a proof of $\vdash_{par} (\!| \phi_0 |\!)\ P\ (\!| \phi_n |\!)$ by interleaving formulas with code as in

$$
\begin{array}{ll}
\quad (\!|\phi_0|\!) & \\
C_1\,; & \\
\quad (\!|\phi_1|\!) & \quad \texttt{justification} \\
C_2\,; & \\
\cdot & \\
\cdot & \\
\cdot & \\
\quad (\!|\phi_{n-1}|\!) & \quad \texttt{justification} \\
C_n\,; & \\
\quad (\!|\phi_n|\!) & \quad \texttt{justification}
\end{array}
$$

Presentation of a proof

A full proof will have one or more conditions before and after each code statement. Each statement makes a Hoare triple with the preceding and following conditions. Each triple (postcondition) has a justification that explains its correctness.

```
(| program precondition |)
y = 1;
(| ... |)                          ⟨justification⟩
while (x != 0) {
    (| ... |)                      ⟨justification⟩
    y = y * x;
    (| ... |)                      ⟨justification⟩
    x = x - 1;
    (| ... |)                      ⟨justification⟩
}
(| program postcondition |)        ⟨justification⟩
```

The rule for assignment has no premises and is therefore an axiom of our logic.

$$\overline{(\!|\, Q[E/x] \,|\!)\ x = E\ (\!|\, Q \,|\!)}$$

Intuition:
If we wish to show that Q holds in the state after $x = E$, we must show that Q holds before the assignment $x = E$, but with all free occurrences of x replaced by E in Q.

Assignment: Examples

2 Axioms and Inference Rules

What is the precondition ϕ?

- $(\!|\, \phi \,|\!)\ x = 2\ (\!|\, x = y \,|\!)$
- $(\!|\, \phi \,|\!)\ x = x + 1\ (\!|\, x = 2 \,|\!)$
- $(\!|\, \phi \,|\!)\ x = y + z\ (\!|\, x = 1 \,|\!)$
- $(\!|\, \phi \,|\!)\ x = x + 1\ (\!|\, x > 0 \wedge y > 0 \,|\!)$

In program correctness proofs, we usually work backwards from the postcondition.

Implied Rule: Precondition Strengthening

The implied rule allows the precondition to be strengthened (i.e., we assume more than we need to).

$$\frac{P \rightarrow P' \qquad (\!| P' |\!)\ C\ (\!| Q |\!)}{(\!| P |\!)\ C\ (\!| Q |\!)}$$

Example:

- $P : x > 10$
- $P' : x > 0$
- P is stronger than P': $x > 10 \rightarrow x > 0$

Examples

2 Axioms and Inference Rules

Prove $\vdash_{par} (\!| y = 5 |\!)\ x = y + 1\ (\!| x = 6 |\!)$

$$
\begin{array}{ll}
\quad (\!| y = 5 |\!) & \\
\quad (\!| y + 1 = 6 |\!) & \text{Implied} \\
\texttt{x = y + 1} & \\
\quad (\!| x = 6 |\!) & \text{Assignment}
\end{array}
$$

Although the proof is constructed bottom-up, its justifications make sense when read top-down.

Implied Rule: Postcondition Weakening

The implied rule allows for the postcondition to be weakened (i.e. we conclude less than we are entitled to):

$$\frac{(\!| P |\!)\ C\ (\!| Q' |\!) \qquad Q' \to Q}{(\!| P |\!)\ C\ (\!| Q |\!)}$$

Intuition: If you can prove something stronger, then you can also claim something weaker.

The Implied Rule

The implied rule acts as a link between program logic and a suitable extension of FOL logic. It allows us to import proofs in predicate logic enlarged with the basic facts of arithmetic, for example:

- $\forall x(x = x + 0)$
- $r = x \wedge q = 0 \to r = x + y * q$

which are required for reasoning about integer expressions, into the proofs in program logic.

Composition

This rule is also known as the sequencing rule, which enables a partial correctness specification for a sequence $C_1; C_2$ to be derived from specification for C_1 and C_2.

$$\frac{(\!|\, P \,|\!)\ C_1\ (\!|\, Q \,|\!) \qquad (\!|\, Q \,|\!)\ C_2\ (\!|\, R \,|\!)}{(\!|\, P \,|\!)\ C_1; C_2\ (\!|\, R \,|\!)}$$

To prove $(\!|\, P \,|\!)\ C_1; C_2\ (\!|\, R \,|\!)$, we need to find appropriate midcondition Q and prove $(\!|\, P \,|\!)\ C_1\ (\!|\, Q \,|\!)$ and $(\!|\, Q \,|\!)\ C_2\ (\!|\, R \,|\!)$ (i.e., by splitting the problem into two.)

In our examples, the midcondition will usually be determined by a rule, such as the assignment rule.

Prove $\vdash_{par} (\!| x = x_0 \wedge y = y_0 |\!)\ t = x; x = y; y = t\ (\!| x = y_0 \wedge y = x_0 |\!)$

$$
\begin{array}{ll}
(\!| ((x = x_0) \wedge (y = y_0)) |\!) & \\
(\!| ((y = y_0) \wedge (x = x_0)) |\!) & \text{implied } \textit{[proof required]} \\
\texttt{t = x ;} & \\
(\!| ((y = y_0) \wedge (t = x_0)) |\!) & \text{assignment} \\
\texttt{x = y ;} & \\
(\!| ((x = y_0) \wedge (t = x_0)) |\!) & \text{assignment} \\
\texttt{y = t ;} & \\
(\!| ((x = y_0) \wedge (y = x_0)) |\!) & \text{assignment}
\end{array}
$$

If statements

The proof rule for if-statements allows us to prove a triple of the form

$$(\!|\, P \,|\!) \text{ if } B \,\{C_1\}\text{ else }\{C_2\}\ (\!|\, Q \,|\!)$$

by decomposing it into two triples, subgoals corresponding to the cases of B evaluating to true and to false (i.e., the preconditions are augmented by the knowledge that B is true and false, respectively).

$$\frac{(\!|\, P \wedge B \,|\!)\ C_1\ (\!|\, Q \,|\!) \qquad (\!|\, P \wedge \neg B \,|\!)\ C_2\ (\!|\, Q \,|\!)}{(\!|\, P \,|\!) \text{ if } B \,\{C_1\}\text{ else }\{C_2\}\ (\!|\, Q \,|\!)}$$

If statements

2 Axioms and Inference Rules

$(\!| P |\!)$

`if ( B ) {`

$(\!| (P \wedge B) |\!)$ — if-then-else

C_1

$(\!| Q |\!)$ — *(justify depending on C_1—a "subproof")*

`} else {`

$(\!| (P \wedge (\neg B)) |\!)$ — if-then-else

C_2

$(\!| Q |\!)$ — *(justify depending on C_2—a "subproof")*

`}`

$(\!| Q |\!)$ — if-then-else [*justifies this Q, given previous two*]

Prove the following is satisfied under partial correctness.

$(\!|\ \mathsf{true}\ |\!)$

```
if ( x > y ) {
    max = x;
} else {
    max = y;
}
```

$(\!|\ (((x > y) \wedge (max = x)) \vee ((x \leq y) \wedge (max = y)))\ |\!)$

Examples

2 Axioms and Inference Rules

$(\!\vert\ \mathit{true}\ \vert\!)$	
`if ( x > y ) {`	
$\quad (\!\vert\ (x > y)\ \vert\!)$	if-then-else
$\quad (\!\vert\ (((x > y) \wedge (x = x)) \vee ((x \leq y) \wedge (x = y)))\ \vert\!)$	implied (a)
`    max = x ;`	
$\quad (\!\vert\ (((x > y) \wedge (\mathit{max} = x)) \vee ((x \leq y) \wedge (\mathit{max} = y)))\ \vert\!)$	assignment
`} else {`	
$\quad (\!\vert\ (\neg(x > y))\ \vert\!)$	if-then-else
$\quad (\!\vert\ (((x > y) \wedge (y = x)) \vee ((x \leq y) \wedge (y = y)))\ \vert\!)$	implied (b)
`    max = y ;`	
$\quad (\!\vert\ (((x > y) \wedge (\mathit{max} = x)) \vee ((x \leq y) \wedge (\mathit{max} = y)))\ \vert\!)$	assignment
`}`	
$(\!\vert\ (((x > y) \wedge (\mathit{max} = x)) \vee ((x \leq y) \wedge (\mathit{max} = y)))\ \vert\!)$	if-then-else

While statements

Suppose our program is: `while B do C`, with:

- B: The loop condition (Boolean expression).
- C: The loop body (the code that runs while B is true).
- I: Loop invariant (manually identified)

A loop invariant (循环不变式) is a logical relationship among the variables (e.g., $x \geq y + 1$) that stays the same throughout the loop:

- It is true before the loop begins.
- It is true at the start of every iteration of the loop and at the end of every iteration..
- It is true after the loop ends.

While statements

2 Axioms and Inference Rules

In the proof rule of partial-while (do not yet require termination):

- Premise: if I and B are true before we execute C, and C terminates, then I still holds
- Conclusion: no matter how many times the body C is executed, if I is true initially and the while statement terminates, then I will be true at the end. Moreover, since the while-statement has terminated, B will be false, $\neg B$ will be true.

$$\frac{(\!|\, I \wedge B \,|\!)\ C\ (\!|\, I \,|\!)}{(\!|\, I \,|\!)\ \text{while}\ B\ \{C\}\ (\!|\, I \wedge \neg B \,|\!)}$$

Proving partial correctness of a while loop

Steps to follow:

- Find a loop invariant (which is both an art and a skill).
- Complete the annotations.
- Prove any implied's.

Prove that the following triple is satisfied under partial correctness.

$$\llparenthesis\, (x \geq 0)\, \rrparenthesis$$

```
y = 1 ;
z = 0 ;
while (z != x) {
    z = z + 1 ;
    y = y * z ;
}
```

$$\llparenthesis\, (y = x!)\, \rrparenthesis$$

Example I

2 Axioms and Inference Rules

Step 1: Write down the values of all the variables every time the `while` test is reached.

$(\!|\ (x \geq 0)\ |\!)$

```
y = 1 ;
z = 0 ;
while (z != x) {
    z = z + 1 ;
    y = y * z ;
}
```

$(\!|\ (y = x!)\ |\!)$

At the while statement:

x	y	z	$z \neq x$
5	1	0	true
5	1	1	true
5	2	2	true
5	6	3	true
5	24	4	true
5	120	5	false

Example I

2 Axioms and Inference Rules

Step 2: Find relationships among the variables that are true for every `while` test. These are our candidate invariants.

$(\!| (x \geq 0) |\!)$

```
y = 1 ;
z = 0 ;
while (z != x) {
    z = z + 1 ;
    y = y * z ;
}
```

$(\!| (y = x!) |\!)$

At the `while` statement:

x	y	z	$z \neq x$
5	1	0	true
5	1	1	true
5	2	2	true
5	6	3	true
5	24	4	true
5	120	5	false

Is $\neg(z = x)$ a loop invariant?

$(\!|\ (x \geq 0)\ |\!)$

```
y = 1 ;
z = 0 ;
while (z != x) {
    z = z + 1 ;
    y = y * z ;
}
```

$(\!|\ (y = x!)\ |\!)$

At the `while` statement:

x	y	z	$z \neq x$
5	1	0	true
5	1	1	true
5	2	2	true
5	6	3	true
5	24	4	true
5	120	5	false

Example I

Is $x \geq 0$ a loop invariant?

$(\!|\ (x \geq 0)\ |\!)$
```
y = 1 ;
z = 0 ;
while (z != x) {
    z = z + 1 ;
    y = y * z ;
}
```
$(\!|\ (y = x!)\ |\!)$

At the `while` statement:

x	y	z	$z \neq x$
5	1	0	true
5	1	1	true
5	2	2	true
5	6	3	true
5	24	4	true
5	120	5	false

Example I

Is $y \geq z$ a loop invariant?

$(\!| (x \geq 0) |\!)$

```
y = 1 ;
z = 0 ;
while (z != x) {
    z = z + 1 ;
    y = y * z ;
}
```

$(\!| (y = x!) |\!)$

At the `while` statement:

x	y	z	$z \neq x$
5	1	0	true
5	1	1	true
5	2	2	true
5	6	3	true
5	24	4	true
5	120	5	false

Example I

2 Axioms and Inference Rules

Is $y = z!$ a loop invariant?

$$\llparenthesis\ (x \geq 0)\ \rrparenthesis$$

```
y = 1 ;
z = 0 ;
while (z != x) {
    z = z + 1 ;
    y = y * z ;
}
```

$$\llparenthesis\ (y = x!)\ \rrparenthesis$$

At the `while` statement:

x	y	z	$z \neq x$
5	1	0	true
5	1	1	true
5	2	2	true
5	6	3	true
5	24	4	true
5	120	5	false

Example I

2 Axioms and Inference Rules

Step 3: Try each candidate invariant until we find one that works for our proof.

$(\!|\ (x \geq 0)\ |\!)$
```
y = 1 ;
z = 0 ;
while (z != x) {
    z = z + 1 ;
    y = y * z ;
}
```
$(\!|\ (y = x!)\ |\!)$

At the while statement:

x	y	z	$z \neq x$
5	1	0	true
5	1	1	true
5	2	2	true
5	6	3	true
5	24	4	true
5	120	5	false

Example I

2 Axioms and Inference Rules

First, annotate by partial-while, with the chosen loop invariant $y = z!$.

$(\!|\ x \geq 0\ |\!)$

```
y = 1 ;

z = 0 ;
```

$(\!|\ y = z!\ |\!)$ *[justification required]*

```
while (z != x) {
```

$(\!|\ (y = z!) \wedge \neg(z = x)\ |\!)$ partial-while $((\!|\ I \wedge B\ |\!))$

```
    z = z + 1 ;

    y = y * z ;
```

$(\!|\ y = z!\ |\!)$ *[justification required]*

```
}
```

$(\!|\ y = z! \wedge (z = x)\ |\!)$ partial-while $((\!|\ I \wedge \neg B\ |\!))$

$(\!|\ y = x!\ |\!)$

Example I

2 Axioms and Inference Rules

Next, annotate assignment statements.

$(\!| x \geq 0 |\!)$
$(\!| 1 = 0! |\!)$
```
y = 1 ;
```
$(\!| y = 0! |\!)$ assignment
```
z = 0 ;
```
$(\!| y = z! |\!)$ assignment
```
while (z != x) {
```
$\quad (\!| (y = z!) \wedge \neg(z = x) |\!)$ partial-while
$\quad (\!| y(z+1) = (z+1)! |\!)$
```
    z = z + 1 ;
```
$\quad (\!| yz = z! |\!)$ assignment
```
    y = y * z ;
```
$\quad (\!| y = z! |\!)$ assignment
```
}
```
$(\!| y = z! \wedge (z = x) |\!)$ partial-while
$(\!| y = x! |\!)$

Example I

2 Axioms and Inference Rules

Then, note the implied, to be proven separately.

$(\!\| x \geq 0 \|\!)$	
$(\!\| 1 = 0! \|\!)$	implied (a)
`y = 1 ;`	
$(\!\| y = 0! \|\!)$	assignment
`z = 0 ;`	
$(\!\| y = z! \|\!)$	assignment
`while (z != x) {`	
$\quad(\!\| (y = z!) \wedge \neg(z = x) \|\!)$	partial-while
$\quad(\!\| y(z+1) = (z+1)! \|\!)$	implied (b)
`    z = z + 1 ;`	
$\quad(\!\| yz = z! \|\!)$	assignment
`    y = y * z ;`	
$\quad(\!\| y = z! \|\!)$	assignment
`}`	
$(\!\| y = z! \wedge (z = x) \|\!)$	partial-while
$(\!\| y = x! \|\!)$	implied (c)

Example I

2 Axioms and Inference Rules

Finally, prove the implied assertions using the inference rules of ordinary logic (FOL, arithmetic).

Proof of implied (a): $(x \geq 0) \vdash (1 = 0!)$
By definition of factorial.

$(\!\| x \geq 0 \|\!)$	
$(\!\| 1 = 0! \|\!)$	implied (a)
`y = 1 ;`	
$(\!\| y = 0! \|\!)$	assignment
`z = 0 ;`	
$(\!\| y = z! \|\!)$	assignment
`while (z != x) {`	
$(\!\| (y = z!) \wedge \neg(z = x) \|\!)$	partial-while
$(\!\| y(z+1) = (z+1)! \|\!)$	implied (b)
`z = z + 1 ;`	
$(\!\| yz = z! \|\!)$	assignment
`y = y * z ;`	
$(\!\| y = z! \|\!)$	assignment
`}`	
$(\!\| y = z! \wedge (z = x) \|\!)$	partial-while
$(\!\| y = x! \|\!)$	implied (c)

Example I

2 Axioms and Inference Rules

Proof of implied (c):

$$(y = z!) \wedge (z = x) \vdash (y = x!)$$

1.	$(y = z!) \wedge (z = x)$	Premise
2.	$(y = z!)$	$\wedge$e:1
3.	$(z = x)$	$\wedge$e:1
4.	$(z! = x!)$	eq. of substitution 3
5.	$(y = x!)$	transitivity of eq. 2,4

$\llparenthesis\ x \geq 0\ \rrparenthesis$	
$\llparenthesis\ 1 = 0!\ \rrparenthesis$	implied (a)
`y = 1 ;`	
$\llparenthesis\ y = 0!\ \rrparenthesis$	assignment
`z = 0 ;`	
$\llparenthesis\ y = z!\ \rrparenthesis$	assignment
`while (z != x) {`	
$\llparenthesis\ (y = z!) \wedge \neg(z = x)\ \rrparenthesis$	partial-while
$\llparenthesis\ y(z+1) = (z+1)!\ \rrparenthesis$	implied (b)
`z = z + 1 ;`	
$\llparenthesis\ yz = z!\ \rrparenthesis$	assignment
`y = y * z ;`	
$\llparenthesis\ y = z!\ \rrparenthesis$	assignment
`}`	
$\llparenthesis\ y = z! \wedge (z = x)\ \rrparenthesis$	partial-while
$\llparenthesis\ y = x!\ \rrparenthesis$	implied (c)

Example I

2 Axioms and Inference Rules

Proof of implied (b):

$$(y = z!) \wedge \neg(z = x) \vdash (z+1)y = (z+1)!$$

1.	$(y = z!) \wedge \neg(z = x)$	Premise
2.	$(y = z!)$	$\wedge$e:1
3.	$(z+1)y = (z+1)z!$	eq. of subs 2
4.	$(z+1)z! = (z+1)!$	def. of factorial 3
5.	$(z+1)y = (z+1)!$	trans of eq. 3,4

```
(| x ≥ 0 |)
(| 1 = 0! |)                              implied (a)
y = 1 ;
(| y = 0! |)                              assignment
z = 0 ;
(| y = z! |)                              assignment
while (z != x) {
    (| (y = z!) ∧ ¬(z = x) |)             partial-while
    (| y(z + 1) = (z + 1)! |)             implied (b)
    z = z + 1 ;
    (| yz = z! |)                         assignment
    y = y * z ;
    (| y = z! |)                          assignment
}
(| y = z! ∧ (z = x) |)                    partial-while
(| y = x! |)                              implied (c)
```

Example II

Prove the following is satisfied under partial correctness.

$$\llparenthesis\ ((n \geq 0) \wedge (a \geq 0))\ \rrparenthesis$$

```
s = 1 ;
i = 0 ;
while (i < n) {
    s = s * a ;
    i = i + 1 ;
}
```

$$\llparenthesis\ (s = a^n)\ \rrparenthesis$$

Example II

2 Axioms and Inference Rules

Step 1: Draw an execution trace to help find the invariant.

$(\!|\, ((n \geq 0) \wedge (a \geq 0)) \,|\!)$

```
s = 1 ;
i = 0 ;
while (i < n) {
    s = s * a ;
    i = i + 1 ;
}
```

$(\!|\, (s = a^n) \,|\!)$

Trace of the loop:

a	n	i	s
2	3	0	1
2	3	1	1∗2
2	3	2	1∗2∗2
2	3	3	1∗2∗2∗2

Example II

2 Axioms and Inference Rules

Attempt 1: try the invariant $s = a^i$.

But implied (c) cannot be proved.

We must use a different invariant.

$(\!|\ ((n \geq 0) \wedge (a \geq 0))\ |\!)$
$(\!|\ \dots\ |\!)$
```
s = 1 ;
```
$(\!|\ \dots\ |\!)$
```
i = 0 ;
```
$(\!|\ (s = a^i)\ |\!)$
```
while (i < n) {
```
$\quad(\!|\ ((s = a^i) \wedge (i < n))\ |\!)$ partial-while
$\quad(\!|\ \dots\ |\!)$
```
  s = s * a ;
```
$\quad(\!|\ \dots\ |\!)$
```
  i = i + 1 ;
```
$\quad(\!|\ (s = a^i)\ |\!)$
```
}
```
$(\!|\ ((s = a^i) \wedge (i \geq n))\ |\!)$ partial-while
$(\!|\ (s = a^n)\ |\!)$ implied (c)

Example II

2 Axioms and Inference Rules

Attempt 2: try the invariant $(s = a^i) \wedge (i \leq n)$.

Now, the proof succeeds.

$$
\begin{array}{ll}
(\!|\, ((n \geq 0) \wedge (a \geq 0)) \,|\!) & \\
(\!|\, ((1 = a^0) \wedge (0 \leq n)) \,|\!) & \text{implied (a)} \\
\texttt{s = 1 ;} & \\
(\!|\, ((s = a^0) \wedge (0 \leq n)) \,|\!) & \text{assignment} \\
\texttt{i = 0 ;} & \\
(\!|\, ((s = a^i) \wedge (i \leq n)) \,|\!) & \text{assignment} \\
\texttt{while (i < n) \{} & \\
\quad (\!|\, (((s = a^i) \wedge (i \leq n)) \wedge (i < n)) \,|\!) & \text{partial-while} \\
\quad (\!|\, (((s \cdot a) = a^{i+1}) \wedge ((i+1) \leq n)) \,|\!) & \text{implied (b)} \\
\quad \texttt{s = s * a ;} & \\
\quad (\!|\, ((s = a^{i+1}) \wedge ((i+1) \leq n)) \,|\!) & \text{assignment} \\
\quad \texttt{i = i + 1 ;} & \\
\quad (\!|\, ((s = a^i) \wedge (i \leq n)) \,|\!) & \text{assignment} \\
\} & \\
(\!|\, (((s = a^i) \wedge (i \leq n)) \wedge (i \geq n)) \,|\!) & \text{partial-while} \\
(\!|\, (s = a^n) \,|\!) & \text{implied (c)}
\end{array}
$$

Example III

For the following program C:

```
z=1;
while (z*z<16) {
    z=z+1;
}
```

Find a loop invariant to prove $\vdash_{par} (\!|\, true \,|\!)\ C\ (\!|\, z = 4 \,|\!)$.

Example III

2 Axioms and Inference Rules

- Loop invariant candidate 1: $z \geq 1$
- This loop invariant $z \geq 1$ is not useful, cannot prove Implied (b).

$(\!\| \text{true} \|\!)$	
`z = 1;`	
$(\!\| (z \geq 1) \|\!)$	Assignment
`while (z * z < 16){`	
$\quad(\!\| ((z \geq 1) \wedge ((z \cdot z) < 16)) \|\!)$	Partial-While
`    z = z + 1;`	
$\quad(\!\| (z \geq 1) \|\!)$	
`}`	
$(\!\| ((z \geq 1) \wedge (\neg((z \cdot z) < 16))) \|\!)$	Partial-While
$(\!\| (z = 4) \|\!)$	???

Example III

2 Axioms and Inference Rules

- Loop invariant candidate 2: $z * z \leq 16$
- This loop invariant $z * z \leq 16$ is not useful, cannot prove Implied (b), since z might be -4.

$(\!|\ \text{true}\ |\!)$	
`z = 1;`	
$(\!|\ ((z \cdot z) \leq 16)\ |\!)$	Assignment
`while (z * z < 16){`	
$\quad (\!|\ (((z \cdot z) \leq 16) \wedge ((z \cdot z) < 16))\ |\!)$	Partial-While
`    z = z + 1;`	
$\quad (\!|\ ((z \cdot z) \leq 16)\ |\!)$	
`}`	
$(\!|\ (((z \cdot z) \leq 16) \wedge (\neg((z \cdot z) < 16)))\ |\!)$	Partial-While
$(\!|\ (z = 4)\ |\!)$	???

Example III

Combine both invariants: $(z \geq 1) \wedge (z * z \leq 16)$

$(\!\| \text{true} \|\!)$	
$(\!\| ((1 \geq 1) \wedge ((1 \cdot 1) \leq 16)) \|\!)$	Implied(a)
`z = 1;`	
$(\!\| ((z \geq 1) \wedge ((z \cdot z) \leq 16)) \|\!)$	Assignment
`while (z * z < 16){`	
$\quad (\!\| (((z \geq 1) \wedge ((z \cdot z) \leq 16)) \wedge ((z \cdot z) < 16)) \|\!)$	Partial-While
$\quad (\!\| (((z+1) \geq 1) \wedge (((z+1) \cdot (z+1)) \leq 16)) \|\!)$	Implied (b)
`    z = z + 1;`	
$\quad (\!\| ((z \geq 1) \wedge ((z \cdot z) \leq 16)) \|\!)$	Assignment
`}`	
$(\!\| (((z \geq 1) \wedge ((z \cdot z) \leq 16)) \wedge (\neg((z \cdot z) < 16))) \|\!)$	Partial-While
$(\!\| (z = 4) \|\!)$	Implied (c)

Choosing loop invariants

The discovery of a suitable invariant:

- a necessary step in order to use the proof rule Partial-while.
- in general it requires intelligence and ingenuity
- This contrasts markedly with the case of the proof rules for if-statements and assignments, which are purely mechanical in nature: their usage is just a matter of symbol-pushing and does not require any deeper insight.

Proving Total Correctness

The proof calculus for total correctness is the same as for partial correctness for all the rules except the rule for `while` statements.

A proof of total correctness for a `while` consists of two parts:

- Proving partial correctness (identify invariant)
- Proving termination (identify variant)

Proving Termination

A variant is an *integer expression* that:

- Always stay non-negative
- Strictly decrease in every loop iteration

If we can find such an expression with these properties, it follows that the `while` statement must terminate: because the expression can only be decremented *a finite number of times* before it becomes 0.

Example of Variants

2 Axioms and Inference Rules

Let's choose the variant: $x - z$.

- At the start of the loop: $x - z \geq 0$ $(x \geq 0, z = 0)$
- At each iteration: z increases by 1, x stays the same, so $x - z$ decreases.

Hence, $x - z$ will eventually reach 0, meaning that the loop terminates.

$\llparenthesis\ (x \geq 0)\ \rrparenthesis$

```
y = 1 ;
z = 0 ;
while (z != x) {
     z = z + 1 ;
     y = y * z ;
}
```

$\llparenthesis\ (y = x!)\ \rrparenthesis$

Choosing loop variants

Finding a working variant is a creative activity which requires skill, intuition and practice.

There is no general method to always find a variant proving termination; in other words, the automatic extraction of useful variants or termination expressions cannot be realized.

Readings

2 Axioms and Inference Rules

- Text B: chapter 4.3, 4.4
- Reference: lecture notes of CS245, University of Waterloo.

Introduction to Mathematical Logic

Thank you for listening!
Any questions?