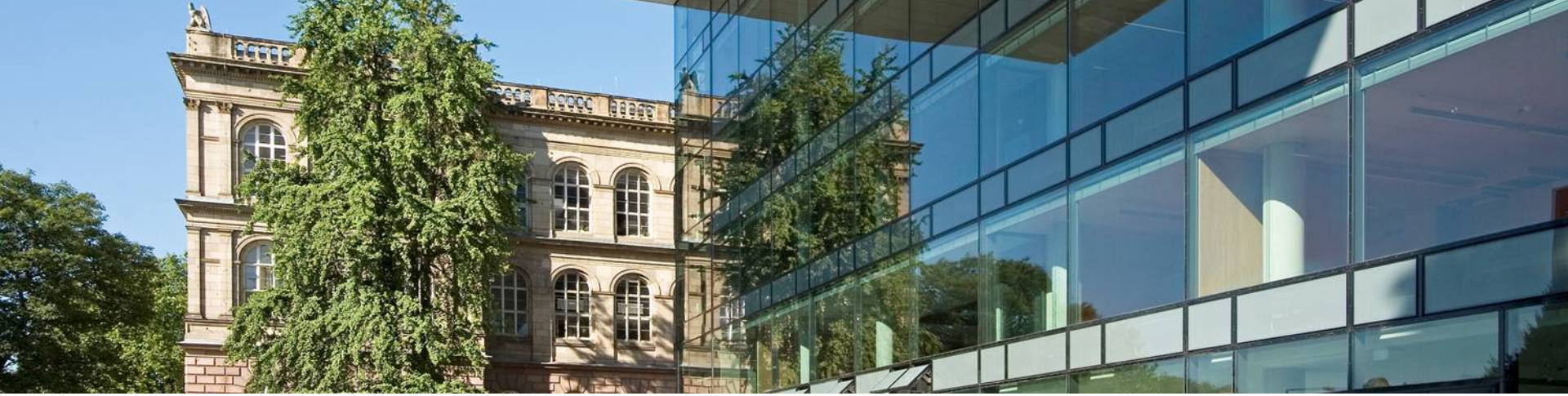


Semantics and Verification of Software

Summer Semester 2019

Lecture 11: Axiomatic Semantics of WHILE III (Completeness & Total Correctness)

Thomas Noll
Software Modeling and Verification Group
RWTH Aachen University

https://moves.rwth-aachen.de/teaching/ss-19/sv-sw/

Software Modeling and Verification Chair

RWTH AACHEN UNIVERSITY

Recap: Hoare Logic

Hoare Logic

Goal: syntactic derivation of valid partial correctness properties.
Here $A[x \mapsto a]$ denotes the syntactic replacement of every occurrence of x by a in A.

Tony Hoare (* 1934)

Definition (Hoare Logic)

The Hoare rules are given by

$$\text{(skip)}\ \frac{}{\{A\}\ \texttt{skip}\ \{A\}} \qquad \text{(asgn)}\ \frac{}{\{A[x \mapsto a]\}\ x := a\ \{A\}}$$

$$\text{(seq)}\ \frac{\{A\}\ c_1\ \{C\} \quad \{C\}\ c_2\ \{B\}}{\{A\}\ c_1 ; c_2\ \{B\}} \qquad \text{(if)}\ \frac{\{A \wedge b\}\ c_1\ \{B\} \quad \{A \wedge \neg b\}\ c_2\ \{B\}}{\{A\}\ \texttt{if}\ b\ \texttt{then}\ c_1\ \texttt{else}\ c_2\ \texttt{end}\ \{B\}}$$

$$\text{(while)}\ \frac{\{A \wedge b\}\ c\ \{A\}}{\{A\}\ \texttt{while}\ b\ \texttt{do}\ c\ \texttt{end}\ \{A \wedge \neg b\}} \qquad \text{(cons)}\ \frac{\models (A \Rightarrow A') \quad \{A'\}\ c\ \{B'\} \quad \models (B' \Rightarrow B)}{\{A\}\ c\ \{B\}}$$

A partial correctness property is provable (notation: $\vdash \{A\}\ c\ \{B\}$) if it is derivable by the Hoare rules. In (while), A is called a (loop) invariant.

Soundness of Hoare Logic

Theorem (Soundness of Hoare Logic)

For every partial correctness property $\{A\}\, c\, \{B\}$,

$$\vdash \{A\}\, c\, \{B\} \quad \Rightarrow \quad \models \{A\}\, c\, \{B\}.$$

Proof.

Let $\vdash \{A\}\, c\, \{B\}$. By induction over the structure of the corresponding proof tree we show that, for every $\sigma \in \Sigma$ and $I \in \mathit{Int}$ such that $\sigma \models^I A$, $\mathfrak{C}[\![c]\!]\sigma = \bot$ or $\mathfrak{C}[\![c]\!]\sigma \models^I B$ (on the board). □

Incompleteness of Hoare Logic I

Soundness: only valid partial correctness properties are provable ✓

Completeness: all valid partial correctness properties are systematically derivable ↯

Theorem 11.1 (Gödel's Incompleteness Theorem)

The set of all valid assertions

$$\{A \in \mathit{Assn} \mid \models A\}$$

is not recursively enumerable, i.e., there exists no proof system for Assn in which all valid assertions are systematically derivable.

Proof.

see [Winskel 1996, p. 110 ff] □

Kurt Gödel
(1906–1978)

Incompleteness of Hoare Logic II

Corollary 11.2

There is no proof system in which all valid partial correctness properties can be enumerated.

Proof.

Given $A \in \mathit{Assn}$, $\models A$ is obviously equivalent to $\{\mathrm{true}\}\ \mathtt{skip}\ \{A\}$. Thus the enumerability of all valid partial correctness properties would imply the enumerability of all valid assertions. □

Remark: alternative proof (using computability theory):
$\{\mathrm{true}\}\ c\ \{\mathrm{false}\}$ is valid iff c does not terminate on any input state. But the set of all non-terminating WHILE statements is not enumerable.

Relative Completeness of Hoare Logic I

- We will see: actual reason of incompleteness is rule

$$\text{(cons)}\ \frac{\models (A \Rightarrow A') \quad \{A'\}\, c\, \{B'\} \quad \models (B' \Rightarrow B)}{\{A\}\, c\, \{B\}}$$

  since it is based on the validity of implications within *Assn*
- The other language constructs are “enumerable”
- Therefore: separation of proof system (Hoare Logic) and assertion language (*Assn*)
- One can show: if an “oracle” is available which decides whether a given assertion is valid, then all valid partial correctness properties can be systematically derived

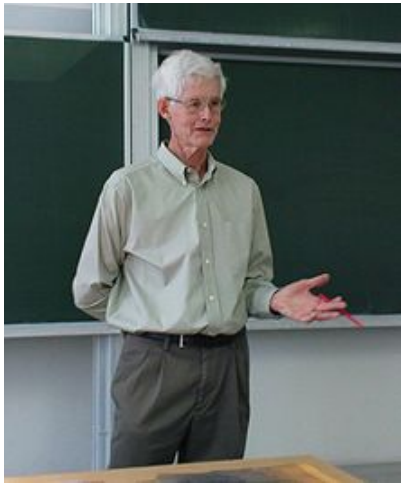
$\Rightarrow$ “Relative completeness”

Relative Completeness of Hoare Logic II

Theorem 11.3 (Cook's Completeness Theorem)

Hoare Logic is relatively complete, i.e., for every partial correctness property $\{A\}\, c\, \{B\}$*:*

$$\models \{A\}\, c\, \{B\} \quad \Rightarrow \quad \vdash \{A\}\, c\, \{B\}.$$



Stephen A. Cook (* 1939)

Thus: if we know that a partial correctness property is valid, then we know that there is a corresponding proof.

The proof uses the following concept: assume that, e.g., $\{A\}\, c_1 ; c_2\, \{B\}$ has to be derived. This requires an intermediate assertion $C \in \mathit{Assn}$ such that $\{A\}\, c_1\, \{C\}$ and $\{C\}\, c_2\, \{B\}$. How to find it?

Weakest Liberal Preconditions I

Definition 11.4 (Weakest liberal precondition)

Given $c \in \mathit{Cmd}$ and $S \subseteq \Sigma$, the weakest (liberal) precondition of S with respect to c collects all states σ such that running c in σ does not terminate or yields a state in S:

$$\mathit{wlp}[\![c]\!]S := \{\sigma \in \Sigma \mid \mathfrak{C}[\![c]\!]\sigma \in S \cup \{\bot\}\}.$$

Corollary 11.5

For every $c \in \mathit{Cmd}$, $A, B \in \mathit{Assn}$, and $I \in \mathit{Int}$:

1. $\models^I \{A\}\, c\, \{B\} \iff A^I \subseteq \mathit{wlp}[\![c]\!]B^I$
2. *If $A_0 \in \mathit{Assn}$ such that $A_0^I = \mathit{wlp}[\![c]\!]B^I$ for every $I \in \mathit{Int}$, then* $\models \{A\}\, c\, \{B\} \iff \models (A \Rightarrow A_0)$

Remarks:

- Corollary 11.5 justifies the notion of weakest precondition: it is entailed by every precondition A that makes $\{A\}\, c\, \{B\}$ valid.
- In the following, we do not distinguish between sets of program states (such as S or A^I) and predicates on program states (such as $\mathfrak{B}[\![b]\!]$).

Weakest Liberal Preconditions II

Lemma 11.6 (Weakest liberal precondition transformer)

Weakest liberal preconditions $wlp[\![.]\!] : Cmd \times 2^\Sigma \to 2^\Sigma$ *can be computed as follows:*

$$
\begin{aligned}
wlp[\![\texttt{skip}]\!]S &= S \\
wlp[\![x := a]\!]S &= \{\sigma \in \Sigma \mid \sigma[x \mapsto \mathfrak{A}[\![a]\!]\sigma] \in S\} \\
wlp[\![c_1 ; c_2]\!]S &= wlp[\![c_1]\!](wlp[\![c_2]\!]S) \\
wlp[\![\texttt{if } b \texttt{ then } c_1 \texttt{ else } c_2 \texttt{ end}]\!]S &= (\mathfrak{B}[\![b]\!] \cap wlp[\![c_1]\!]S) \cup (\mathfrak{B}[\![\neg b]\!] \cap wlp[\![c_2]\!]S) \\
wlp[\![\texttt{while } b \texttt{ do } c \texttt{ end}]\!]S &= \mathrm{FIX}(\Psi)
\end{aligned}
$$

where $\mathrm{FIX}(\Psi)$ *denotes the greatest fixpoint (w.r.t.* $(2^\Sigma, \subseteq)$*) of*

$$\Psi : 2^\Sigma \to 2^\Sigma : T \mapsto (\mathfrak{B}[\![b]\!] \cap wlp[\![c]\!]T) \cup (\mathfrak{B}[\![\neg b]\!] \cap S)$$

Remark: $\mathrm{FIX}(\Psi)$ of a continuous function Ψ on lattice $(2^\Sigma, \subseteq)$ can be computed by fixpoint iteration (see following slide)

Proof.

omitted □

Weakest Liberal Preconditions III

Example 11.7

Using Lemma 11.6, we want to determine the weakest liberal precondition for

$$\{?\}\ \underbrace{\texttt{while x} \neq 0 \wedge \texttt{x} \neq 1 \texttt{ do } \overbrace{\texttt{x := x-2}}^{c_0} \texttt{ end}}_{c}\ \{\texttt{x} = 1\}$$

i.e., $wlp[\![c]\!]S$ for $S := \mathfrak{B}[\![\texttt{x} = 1]\!]$.

- $wlp[\![c]\!]S = \mathrm{FIX}(\Psi)$ for $\Psi(T) = (\mathfrak{B}[\![\texttt{x} \neq 0 \wedge \texttt{x} \neq 1]\!] \cap wlp[\![c_0]\!]T) \cup \underbrace{(\mathfrak{B}[\![\texttt{x} \in \{0,1\}]\!] \cap S)}_{=S}$
- $wlp[\![c_0]\!]T = \{\sigma \in \Sigma \mid \sigma[\texttt{x} \mapsto \sigma(\texttt{x}) - 2] \in T\}$
- Fixpoint iteration (with initial value $\bigsqcap \emptyset = \Sigma$):

$$\begin{aligned}
\Psi(\Sigma) &= (\mathfrak{B}[\![\texttt{x} \neq 0 \wedge \texttt{x} \neq 1]\!] \cap wlp[\![c_0]\!]\Sigma) \cup S = \mathfrak{B}[\![\texttt{x} \neq 0]\!] \\
\Psi^2(\Sigma) &= (\mathfrak{B}[\![\texttt{x} \neq 0 \wedge \texttt{x} \neq 1]\!] \cap wlp[\![c_0]\!](\mathfrak{B}[\![\texttt{x} \neq 0]\!])) \cup S = \mathfrak{B}[\![\texttt{x} \neq 0 \wedge \texttt{x} \neq 2]\!] \\
\Psi^3(\Sigma) &= (\mathfrak{B}[\![\texttt{x} \neq 0 \wedge \texttt{x} \neq 1]\!] \cap wlp[\![c_0]\!](\mathfrak{B}[\![\texttt{x} \neq 0 \wedge \texttt{x} \neq 2]\!])) \cup S = \mathfrak{B}[\![\texttt{x} \neq 0 \wedge \texttt{x} \neq 2 \wedge \texttt{x} \neq 4]\!] \\
&\vdots
\end{aligned}$$

$$\Rightarrow \mathrm{FIX}(\Psi) = \bigcap_{n \in \mathbb{N}} \Psi^n(\Sigma) = \{\sigma \in \Sigma \mid \sigma(\texttt{x}) \in \mathbb{Z}_{<0} \cup \{1, 3, 5, \ldots\}\}$$

Weakest Liberal Preconditions IV

Definition 11.8 (Expressivity of assertion languages)

An assertion language *Assn* is called expressive if it allows to "syntactify" weakest preconditions, that is, for every $c \in Cmd$ and $B \in Assn$, there exists $A_{c,B} \in Assn$ such that $A^I_{c,B} = wlp[\![c]\!]B^I$ for every $I \in Int$.

Theorem 11.9 (Expressivity of *Assn*)

Assn is expressive.

Proof (idea; see (Winskel 1996, p. 103 ff) for details).

Given $c \in Cmd$ and $B \in Assn$, construct $A_{c,B} \in Assn$ with $\sigma \models^I A_{c,B} \iff \mathfrak{C}[\![c]\!]\sigma \models^I B$ (for every $\sigma \in \Sigma$, $I \in Int$). For example:

$$A_{\texttt{skip},B} := B \qquad A_{x:=a,B} := B[x \mapsto a]$$
$$A_{c_1;c_2,B} := A_{c_1,A_{c_2,B}} \qquad \ldots$$

(for `while`: "Gödelisation" of sequences of intermediate states) □

Relative Completeness of Hoare Logic III

The following lemma shows that syntactic weakest preconditions are “provable”:

Lemma 11.10

For every $c \in \mathit{Cmd}$ *and* $B \in \mathit{Assn}$: $\vdash \{A_{c,B}\}\, c\, \{B\}$

Proof.

by structural induction over c (omitted) □

Proof (Cook’s Completeness Theorem 11.3).

We have to show that Hoare Logic is relatively complete, i.e., that

$$\models \{A\}\, c\, \{B\} \quad \Rightarrow \quad \vdash \{A\}\, c\, \{B\}.$$

- Lemma 11.10: $\vdash \{A_{c,B}\}\, c\, \{B\}$
- Corollary 11.5: $\models \{A\}\, c\, \{B\} \quad \Rightarrow \quad \models (A \Rightarrow A_{c,B})$
- $$\text{(cons)}\ \frac{\models (A \Rightarrow A_{c,B}) \qquad \{A_{c,B}\}\, c\, \{B\} \qquad \models (B \Rightarrow B)}{\{A\}\, c\, \{B\}}$$

□

Total Correctness

- **Observation:** partial correctness properties only speak about terminating computations of a given program
- Total correctness additionally requires the proof that the program indeed stops (on the input states admitted by the precondition)
- Consider total correctness properties of the form

$$\{A\}\, c\, \{\Downarrow B\}$$

  where $c \in \mathit{Cmd}$ and $A, B \in \mathit{Assn}$
- Interpretation:

Validity of property $\{A\}\, c\, \{\Downarrow B\}$

For all states $\sigma \in \Sigma$ which satisfy A:
the execution of c in σ terminates and yields a state which satisfies B.

Semantics of Total Correctness Properties

Definition 11.11 (Semantics of total correctness properties)

Let $A, B \in \mathit{Assn}$ and $c \in \mathit{Cmd}$.

- $\{A\}\, c\, \{\Downarrow B\}$ is called valid in $\sigma \in \Sigma$ and $I \in \mathit{Int}$ (notation: $\sigma \models^I \{A\}\, c\, \{\Downarrow B\}$) if $\sigma \models^I A$ implies that $\mathfrak{C}[\![c]\!]\sigma \models^I B$.
- $\{A\}\, c\, \{\Downarrow B\}$ is called valid in $I \in \mathit{Int}$ (notation: $\models^I \{A\}\, c\, \{\Downarrow B\}$) if $\sigma \models^I \{A\}\, c\, \{\Downarrow B\}$ for every $\sigma \in \Sigma$.
- $\{A\}\, c\, \{\Downarrow B\}$ is called valid (notation: $\models \{A\}\, c\, \{\Downarrow B\}$) if $\models^I \{A\}\, c\, \{\Downarrow B\}$ for every $I \in \mathit{Int}$.

Obviously, total implies partial correctness (but not vice versa):

Corollary 11.12

For all $A, B \in \mathit{Assn}$ and $c \in \mathit{Cmd}$,

$$\models \{A\}\, c\, \{\Downarrow B\} \quad \Rightarrow \quad \models \{A\}\, c\, \{B\}.$$

Proving Total Correctness I

Definition 11.13 (Hoare Logic for total correctness)

The Hoare rules for total correctness are given by (where $i \in LVar$)

$$\text{(skip)}\ \frac{}{\{A\}\ \texttt{skip}\ \{\Downarrow A\}} \qquad \text{(asgn)}\ \frac{}{\{A[x \mapsto a]\}\ x := a\ \{\Downarrow A\}}$$

$$\text{(seq)}\ \frac{\{A\}\ c_1\ \{\Downarrow C\} \quad \{C\}\ c_2\ \{\Downarrow B\}}{\{A\}\ c_1 ; c_2\ \{\Downarrow B\}} \qquad \text{(if)}\ \frac{\{A \wedge b\}\ c_1\ \{\Downarrow B\} \quad \{A \wedge \neg b\}\ c_2\ \{\Downarrow B\}}{\{A\}\ \texttt{if}\ b\ \texttt{then}\ c_1\ \texttt{else}\ c_2\ \texttt{end}\ \{\Downarrow B\}}$$

$$\text{(while)}\ \frac{\models (i \geq 0 \wedge A(i+1) \Rightarrow b) \quad \{i \geq 0 \wedge A(i+1)\}\ c\ \{\Downarrow A(i)\} \quad \models (A(0) \Rightarrow \neg b)}{\{\exists i.\, i \geq 0 \wedge A(i)\}\ \texttt{while}\ b\ \texttt{do}\ c\ \texttt{end}\ \{\Downarrow A(0)\}}$$

$$\text{(cons)}\ \frac{\models (A \Rightarrow A') \quad \{A'\}\ c\ \{\Downarrow B'\} \quad \models (B' \Rightarrow B)}{\{A\}\ c\ \{\Downarrow B\}}$$

A total correctness property is provable (notation: $\vdash \{A\}\ c\ \{\Downarrow B\}$) if it is derivable by the Hoare rules. In case of (while), $A(i)$ is called a (loop) invariant.

Proving Total Correctness II

- In rule

$$\text{(while)}\ \frac{\models (i \geq 0 \land A(i+1) \Rightarrow b) \quad \{i \geq 0 \land A(i+1)\}\, c\, \{\Downarrow A(i)\} \quad \models (A(0) \Rightarrow \neg b)}{\{\exists i.\, i \geq 0 \land A(i)\}\ \texttt{while}\ b\ \texttt{do}\ c\ \texttt{end}\ \{\Downarrow A(0)\}}$$

  the notation $A(i)$ indicates that assertion A parametrically depends on the value of the logical variable $i \in LVar$.
- Idea: i represents the remaining number of loop iterations
- Loop to be traversed $i+1$ times ($i \geq 0$)
  $\Rightarrow A(i+1)$ holds
  $\Rightarrow$ execution condition b satisfied

  Thus: $\models (i \geq 0 \land A(i+1) \Rightarrow b)$, and $i+1$ decreased to i after execution of c
- Execution terminated
  $\Rightarrow A(0)$ holds
  $\Rightarrow$ execution condition b violated

  Thus: $\models (A(0) \Rightarrow \neg b)$

Total Correctness of Factorial Program I

Example 11.14

Proof of $\{A\}\ \mathtt{y:=1}\,;c\,\{\Downarrow B\}$ where

$$A := (\mathtt{x} > 0 \wedge \mathtt{x} = i)$$
$$c := \mathtt{while}\ \neg(\mathtt{x=1})\ \mathtt{do}\ \mathtt{y:=y*x;}\ \mathtt{x:=x-1}\ \mathtt{end}$$
$$B := (\mathtt{y} = i!)$$

First we show that the assertion $C(j) = (\mathtt{x} > 0 \wedge \mathtt{y} * \mathtt{x}! = i! \wedge \mathtt{x} = j + 1)$ is an invariant of c. Applying (asgn) twice yields

$$\vdash \{j \geq 0 \wedge C(j)[\mathtt{x} \mapsto \mathtt{x-1}]\}\ \mathtt{x:=x-1}\ \{\Downarrow j \geq 0 \wedge C(j)\} \quad \text{and}$$
$$\vdash \{j \geq 0 \wedge C(j)[\mathtt{x} \mapsto \mathtt{x-1}][\mathtt{y} \mapsto \mathtt{y*x}]\}\ \mathtt{y:=y*x}\ \{\Downarrow j \geq 0 \wedge C(j)[\mathtt{x} \mapsto \mathtt{x-1}]\}$$

such that (seq) implies

$$\vdash \{j \geq 0 \wedge C(j)[\mathtt{x} \mapsto \mathtt{x-1}][\mathtt{y} \mapsto \mathtt{y*x}]\}\ \mathtt{y:=y*x;}\ \ \mathtt{x:=x-1}\ \{\Downarrow j \geq 0 \wedge C(j)\}.$$

Now $C(j+1) = (\mathtt{x} > 0 \wedge \mathtt{y*x}! = i! \wedge \mathtt{x} = j + 2)$ and
$C(j)[\mathtt{x} \mapsto \mathtt{x-1}][\mathtt{y} \mapsto \mathtt{y*x}] = (\mathtt{x} - 1 > 0 \wedge \mathtt{y} * \mathtt{x} * (\mathtt{x} - 1)! = i! \wedge \mathtt{x} - 1 = j + 1)$
such that

$$\models ((j \geq 0 \wedge C(j+1)) \Rightarrow (j \geq 0 \wedge C(j)[\mathtt{x} \mapsto \mathtt{x-1}][\mathtt{y} \mapsto \mathtt{y*x}])) \text{ and}$$
$$\models ((j \geq 0 \wedge C(j)) \Rightarrow C(j)).$$

Total Correctness of Factorial Program II

Example 11.14 (continued; $C(j) = (\mathtt{x} > 0 \wedge \mathtt{y} * \mathtt{x}! = i! \wedge \mathtt{x} = j + 1)$)

Hence (cons) implies

$$\vdash \{j \geq 0 \wedge C(j+1)\}\ \mathtt{y:=y*x;\ \ x:=x-1}\ \{\Downarrow C(j)\}.$$

Moreover we have

$$\models ((j \geq 0 \wedge C(j+1)) \Rightarrow \neg(\mathtt{x} = 1)) \text{ and } \models (C(0) \Rightarrow \neg(\neg(\mathtt{x} = 1)))$$

such that (while) yields

$$\vdash \{\exists j.j \geq 0 \wedge C(j)\}\ c\ \{\Downarrow C(0)\}.$$

For the initializing assignment, (asgn) implies

$$\vdash \{\exists j.j \geq 0 \wedge C(j)[\mathtt{y} \mapsto 1]\}\ \mathtt{y:=1}\ \{\Downarrow \exists j.j \geq 0 \wedge C(j)\},$$

such that (seq) allows to conclude

$$\vdash \{\exists j.j \geq 0 \wedge C(j)[\mathtt{y} \mapsto 1]\}\ \mathtt{y:=1;}\, c\ \{\Downarrow C(0)\}.$$

On the other hand we have (choose $j := i - 1$):

$$\models ((\mathtt{x} > 0 \wedge x = i) \Rightarrow (\exists j.j \geq 0 \wedge C(j)[\mathtt{y} \mapsto 1])) \text{ and } \models (C(0) \Rightarrow \mathtt{y} = i!)$$

such that (cons) yields the desired result:

$$\vdash \{\mathtt{x} > 0 \wedge \mathtt{x} = i\}\ \mathtt{y:=1;}\, c\ \{\Downarrow \mathtt{y} = i!\}.$$