FMSoft
Lecture 11 — Kleene theorem and denotational semantics of loops

(lecture version)

Tadeusz Litak

Jan 8, 2019

Informatik 8, FAU Erlangen-Nürnberg

reminder 1: semantics of non-WHILE programs

SKIP

$$[\![\texttt{SKIP}]\!]\sigma := \sigma$$

i.e., $[\![\texttt{SKIP}]\!] = \lambda\sigma.\sigma = \mathsf{id}_{\mathsf{States}_{\mathsf{IMP}}} = \sigma \mapsto \sigma$

ASSIGNMENT

For this, we need to recall an useful construction from GLoIn, which we can call function update. Using notation from the skript:

$$\sigma[X \mapsto n](Y) := \begin{cases} n, & \text{if } X = Y \\ \sigma(Y) & \text{otherwise.} \end{cases}$$

It is also often denoted as $\sigma[n/X]$, for example in Winskel's book. This is the notation we will use here

$$\text{Now set} \quad [\![\mathsf{X} := a]\!]\sigma := \sigma[n/\mathsf{X}], \quad \text{where } n = [\![a]\!]\sigma$$

i.e., $[\![\mathsf{X} := \mathrm{a}]\!] = \lambda\sigma.\sigma[[\![a]\!]\sigma/\mathsf{X}] = \sigma \mapsto \sigma[[\![a]\!]\sigma/\mathsf{X}]$.

With the last one, you probably see why I preferred the "substitution-like" notation for update.

SEQUENCING

Denotation of sequencing/composition is of course defined by composition of partial functions we defined before:

$$[\![c_1 \ ; \ c_2]\!] := [\![c_2]\!] \circ [\![c_1]\!]$$

CONDITIONALS

$$[\![\texttt{IF}\ b\ \texttt{THEN}\ c_1\ \texttt{ELSE}\ c_2\ \texttt{ENDIF}]\!]\sigma := \begin{cases} [\![c_1]\!]\sigma & \text{if } [\![b]\!]\sigma = \top \\ [\![c_2]\!]\sigma & \text{if } [\![b]\!]\sigma = \bot \end{cases}$$

What do we need for WHILE?

Consider a functional (a.k.a. higher-order function)

or functional form

$\Phi_{b,c} : (\mathsf{States}_{\mathsf{IMP}} \rightharpoonup \mathsf{States}_{\mathsf{IMP}}) \to (\mathsf{States}_{\mathsf{IMP}} \rightharpoonup \mathsf{States}_{\mathsf{IMP}})$

defined as follows:

$$\Phi_{b,c}(f)(\sigma) := \begin{cases} \sigma & \text{if } [\![b]\!]\sigma = \bot \\ f([\![c]\!]\sigma) & \text{if } [\![b]\!]\sigma = \top \end{cases}$$

Clearly, the denotation of `WHILE` we are looking for must be a fixed point of this functional, i.e., it must satisfy

$$\Phi_{b,c}(f) = f$$

We already have some background on this ...

- We remember that monotone functions on complete lattices have fixpoints (Knaster-Tarski)
- But $\mathsf{States}_{\mathsf{IMP}} \rightharpoonup \mathsf{States}_{\mathsf{IMP}}$ is not a complete lattice
- Recall: $f \sqsubseteq g$ iff $f \subseteq g$ when f and g are identified with their graphs
- This means: $dom(f) \subseteq dom(g)$ and whenever $f(\sigma) \neq \perp\!\!\!\perp$, $f(\sigma) = g(\sigma)$
- This does yield a poset

- Also, recalling that

$$\Phi_{b,c}(f)(\sigma) := \begin{cases} \sigma & \text{if } [\![b]\!]\sigma = \bot \\ f([\![c]\!]\sigma) & \text{if } [\![b]\!]\sigma = \top \end{cases}$$

  pretty obvious to see $f \sqsubseteq g$ implies $\Phi_{b,c}(f) \sqsubseteq \Phi_{b,c}(g)$
- So $\Phi_{b,c}$ is a monotone function on a poset $\mathsf{States}_{\mathsf{IMP}} \rightharpoonup \mathsf{States}_{\mathsf{IMP}}$
- But this poset fails to be a lattice, much less a complete one
- Consider $f_1 = \{\langle \lambda \mathsf{X}.0, \lambda \mathsf{X}.0 \rangle\}$ and $f_2 = \{\langle \lambda \mathsf{X}.0, \lambda \mathsf{X}.1 \rangle\}$
- $f_1 \cup f_1$ is not a partial function and there cannot be any playing the role of $f_1 \sqcup f_2$

- ω-chain in a poset $(W, \sqsubseteq)$: $w_0 \sqsubseteq w_1 \sqsubseteq w_2 \sqsubseteq \ldots$
- More formally, we can identify such a chain with a poset morphism (other names: a function which is monotone or order-preserving) $w : (\mathbb{N}, \leq) \longrightarrow (W, \sqsubseteq)$
- $(W, \sqsubseteq)$ is an ω-cpo if suprema of all ω-chains exist, and so does the least element $\bot$

  sometimes called instead $\omega\text{-cpos}_\bot$, but we won't be so pedantic
- $\mathsf{States}_{\mathsf{IMP}} \rightharpoonup \mathsf{States}_{\mathsf{IMP}}$ is an ω-cpo (on blackboard)
- (in fact, with a suitable generalization of the notion of a chain, we could do better than that)

- Does every monotone function on an ω-cpo have a fixpoint?

  Actually, the best counterexamples I was able to come up with are

  set-theoretical and I don't assume you all know set theory

- Anyway, here's a mild (?) strengthening of monotonicity
- Let $(W, \sqsubseteq)$ be an ω-cpo. Say that $\Psi : (W, \sqsubseteq) \longrightarrow (W, \sqsubseteq)$ is ω-continuous if for any ω-chain $w_0 \sqsubseteq w_1 \sqsubseteq w_2 \sqsubseteq \ldots$, $\Psi(\bigsqcup w_n) = \bigsqcup \Psi(w_n)$
- Note when Ψ is monotone, we have that $\Psi(w_0) \sqsubseteq \Psi(w_1) \sqsubseteq \Psi(w_2) \sqsubseteq \ldots$
- On the other hand, does ω-continuity imply monotonicity?

  This uses the following reading of this equation: *for any ω-chain, $\bigsqcup \Psi(w_n)$ has to exist and is equal to $\Psi(\bigsqcup w_n)$*

Kleene Theorem and WHILE

Theorem (Kleene)
If $(W, \sqsubseteq)$ *is an* ω*-cpo and* $\Psi : (W, \sqsubseteq) \longrightarrow (W, \sqsubseteq)$ *is* ω*-continuous, the least fixpoint of* Ψ *is given by* $\bigsqcup_{n \in \mathbb{N}} \Psi^n(\bot)$

Proof.
On the blackboard. □

- Let us return to
  $\Phi_{b,c} : (\mathsf{States}_{\mathsf{IMP}} \rightharpoonup \mathsf{States}_{\mathsf{IMP}}) \to (\mathsf{States}_{\mathsf{IMP}} \rightharpoonup \mathsf{States}_{\mathsf{IMP}})$
  defined as follows:

$$\Phi_{b,c}(f)(\sigma) := \begin{cases} \sigma & \text{if } [\![b]\!]\sigma = \bot \\ f([\![c]\!]\sigma) & \text{if } [\![b]\!]\sigma = \top. \end{cases}$$

  Is it ω-continuous? Split cases:
- either $[\![b]\!]\sigma = \bot$ or ...
- ...or $\bigsqcup f_n([\![c]\!]\sigma) = \perp\!\!\!\perp$, in which case $\forall n. f_n([\![c]\!]\sigma) = \perp\!\!\!\perp$ ...
- ...or $\bigsqcup f_n([\![c]\!]\sigma) \neq \perp\!\!\!\perp$, in which case $\exists n. f_n([\![c]\!]\sigma) \neq \perp\!\!\!\perp$
- ...finish as HA

Finally!

$$[\![\texttt{WHILE}\ b\ \texttt{DO}\ c\ \texttt{END}]\!] := \bigcup_{n \in \omega} \Phi^n_{b,c}(\varnothing)$$

where, recall,

$$\Phi_{b,c}(f)(\sigma) := \begin{cases} \sigma & \text{if } [\![b]\!]\sigma = \bot \\ f([\![c]\!]\sigma) & \text{if } [\![b]\!]\sigma = \top \end{cases}$$

We need a more transparent description ...

Lemma (WHILE helper)
for any n', σ, b, c *if* $\Phi_{b,c}^{n'+1}(\varnothing)(\sigma)$ *is defined, then there exists* $n \leq n'$ *s.t.*

(a) $[\![c]\!]^n\sigma = [\![$ `WHILE` b `DO` c `END` $]\!]\sigma$ *and*

(b) $[\![c]\!]^n\sigma \vDash^I \neg b$ *and*

(c) *for all* $k < n$, $[\![c]\!]^k\sigma \vDash^I b$

- We turned IMP programs into mathematical objects

- We turned IMP programs into mathematical objects
- (would be much harder for "real-life" languages!)

- We turned IMP programs into mathematical objects
- (would be much harder for "real-life" languages!)
- Now we can return to Hoare triples

  we know now what to evaluate soundness and completeness claims against

reminder 2: semantics of Hoare assertions

Reminder

- Given $\mathsf{b} \in \mathsf{BExp}$ and $\sigma \in \mathsf{States}_{\mathsf{IMP}}$, the denotation $[\![\ \mathsf{b}\ ]\!]\sigma$ is an element of the obvious object of truth values $2 := \{\top, \bot\}$
- Similarly, $[\![B]\!]\sigma I \in 2$. Denote the obvious boolean operations on 2 as $\wedge$, $\vee$, $\neg$ ...
- $[\![\ \texttt{true}\ ]\!]\sigma I := \top$
- $$[\![a_1 \texttt{ == } a_2]\!]\sigma I := \begin{cases} \top & \text{if } [\![a_1]\!]\sigma I = [\![a_2]\!]\sigma I \\ \bot & \text{otherwise} \end{cases}$$
- $$[\![a_1 \texttt{ < } a_2]\!]\sigma I := \begin{cases} \top & \text{if } [\![a_1]\!]\sigma I < [\![a_2]\!]\sigma I \\ \bot & \text{otherwise} \end{cases}$$
- $[\![\ \texttt{not}\ B]\!]\sigma I := \neg [\![B]\!]\sigma I$
- $$[\![\bigwedge_{z \in \mathbb{Z}} B_z]\!]\sigma I = \begin{cases} \top & \text{if } [\![B_z]\!]\sigma I = \top \text{ for all } z \\ \bot & \text{otherwise} \end{cases}$$

Reminder

- Recall that $\sigma \Vdash^I B$ was another notation for $[\![B]\!]\sigma I = \top$
- Exercise reminder: Show that $[\![B[\mathsf{z}/i]]\!]\sigma I = [\![B]\!]\sigma I[z/i]$ and use this to show that forcing for defined quantifiers agrees with the standard definition
- Furthermore, write $\Vdash B$ for "for all σ and I, $\sigma \Vdash^I B$"
- Finally, let us introduce the PCA $\perp\!\!\!\perp$-convention:

$$\text{for any } B \text{ and } I,\ \perp\!\!\!\perp \Vdash^I B$$

Soundness of Hoare rules

Reminder

$$\vdash \{A\}\ \texttt{SKIP}\ \{A\}$$

$$\vdash \{B[a/\ \texttt{X}\ ]\}\ \texttt{X}\ \texttt{:=}\ a\{B\}$$

$$\frac{\vdash \{A_1\}c_1\{A_2\} \qquad \vdash \{A_2\}c_2\{A_3\}}{\vdash \{A_1\}c_1\ \texttt{;}\ c_2\{A_3\}}$$

$$\frac{\vdash \{A_1 \wedge b\}c_1\{A_2\} \qquad \vdash \{A_1 \wedge \neg b\}c_2\{A_2\}}{\vdash \{A_1\}\ \texttt{IF}\ b\ \texttt{THEN}\ c_1\ \texttt{ELSE}\ c_2\ \texttt{ENDIF}\ \{A_2\}}$$

$$\frac{\vdash \{A \wedge b\}c\{A\}}{\vdash \{A\}\ \texttt{WHILE}\ b\ \texttt{DO}\ c\ \texttt{END}\ \{A \wedge \neg b\}}$$

$$\frac{\vDash A' \to A, \quad \vdash \{A\}c\{B\}, \quad \vDash B \to B'}{\vdash \{A'\}c\{B'\}}$$

- We can formally define (PCA)-validity of Hoare triples:

$$\begin{aligned} \sigma \vDash^I \{A\}c\{B\} \quad & \sigma \vDash^I A \text{ implies } [\![c]\!]\sigma \vDash^I B \\ \sigma \vDash \{A\}c\{B\} \quad & \text{for any } I,\ \sigma \vDash^I \{A\}c\{B\} \\ \vDash \{A\}c\{B\} \quad & \text{for any } \sigma,\ \sigma \vDash \{A\}c\{B\} \end{aligned}$$

- We can formally define (PCA)-validity of Hoare triples:

$$\begin{aligned} \sigma \vDash^I \{A\}c\{B\} \quad & \sigma \vDash^I A \text{ implies } [\![c]\!]\sigma \vDash^I B \\ \sigma \vDash \{A\}c\{B\} \quad & \text{for any } I,\ \sigma \vDash^I \{A\}c\{B\} \\ \vDash \{A\}c\{B\} \quad & \text{for any } \sigma,\ \sigma \vDash \{A\}c\{B\} \end{aligned}$$

- Do we have that $\sigma \vDash \{A\}c\{B\}$ when c executed in σ does not terminate?

- We can formally define (PCA)-validity of Hoare triples:

$$\begin{aligned} \sigma \vDash^I \{A\}c\{B\} \quad & \sigma \vDash^I A \text{ implies } [\![c]\!]\sigma \vDash^I B \\ \sigma \vDash \{A\}c\{B\} \quad & \text{for any } I,\ \sigma \vDash^I \{A\}c\{B\} \\ \vDash \{A\}c\{B\} \quad & \text{for any } \sigma,\ \sigma \vDash \{A\}c\{B\} \end{aligned}$$

- Do we have that $\sigma \vDash \{A\}c\{B\}$ when c executed in σ does not terminate?

- Consider the opposite total $\lightning$-convention:

$$\text{for any } B \text{ and } I,\ \lightning \not\vDash^I B \quad ?$$

- Recall $[\![$ SKIP $]\!]\sigma = \sigma$
- Recall our proposed axiom: $\vdash \{A\}$ SKIP $\{A\}$
- Recall our definition of $\vDash \{A\}c\{B\}$: for any σ and I, $\sigma \vDash^I A$ implies $[\![c]\!]\sigma \vDash^I B$
- Clearly, we have $\vDash \{A\}$ SKIP $\{A\}$

- For assignment, we proposed $\vdash \{B[a/\ \mathsf{X}\ ]\}\ \mathsf{X}\ \mathsf{:=}\ a\{B\}$

- For assignment, we proposed $\vdash \{B[a/\ \mathsf{X}\ ]\}\ \mathsf{X}\ :=\ a\{B\}$
- Recall $[\![\ \mathsf{X}\ :=\ a]\!]\sigma = \sigma[[\![a]\!]\sigma/\mathsf{X}]$

- For assignment, we proposed $\vdash \{B[a/\ \mathsf{X}\ ]\}\ \mathsf{X}\ :=\ a\{B\}$
- Recall $[\![\ \mathsf{X}\ :=\ a]\!]\sigma = \sigma[[\![a]\!]\sigma/\mathsf{X}]$
- Assume $\sigma \models^I B[a/\ \mathsf{X}\ ]$. Is it true that $\sigma[[\![a]\!]\sigma/\mathsf{X}] \models^I B$?

- For assignment, we proposed $\vdash \{B[a/\ \mathsf{X}\ ]\}\ \mathsf{X}\ :=\ a\{B\}$
- Recall $[\![\ \mathsf{X}\ :=\ a]\!]\sigma = \sigma[[\![a]\!]\sigma/\mathsf{X}]$
- Assume $\sigma \models^I B[a/\ \mathsf{X}\ ]$. Is it true that $\sigma[[\![a]\!]\sigma/\mathsf{X}] \models^I B$?
- In fact, it is an equivalent way of saying the same thing

- For assignment, we proposed $\vdash \{B[a/\ \mathsf{X}\ ]\}\ \mathsf{X}\ :=\ a\{B\}$
- Recall $[\![\ \mathsf{X}\ :=\ a]\!]\sigma = \sigma[[\![a]\!]\sigma/\mathsf{X}]$
- Assume $\sigma \models^I B[a/\ \mathsf{X}\ ]$. Is it true that $\sigma[[\![a]\!]\sigma/\mathsf{X}] \models^I B$?
- In fact, it is an equivalent way of saying the same thing
- Proof: induction on the complexity of B, we can guess?

- For assignment, we proposed $\vdash \{B[a/\ \mathsf{X}\ ]\}\ \mathsf{X}\ :=\ a\{B\}$
- Recall $[\![\ \mathsf{X}\ :=\ a]\!]\sigma = \sigma[[\![a]\!]\sigma/\mathsf{X}]$
- Assume $\sigma \models^I B[a/\ \mathsf{X}\ ]$. Is it true that $\sigma[[\![a]\!]\sigma/\mathsf{X}] \models^I B$?
- In fact, it is an equivalent way of saying the same thing
- Proof: induction on the complexity of B, we can guess?
- $\sigma \models^I (a_1\ ==\ a_2)[a/\ \mathsf{X}\ ]$ iff $[\![a_1[a/\ \mathsf{X}\ ]]\!]\sigma I = [\![a_2[a/\ \mathsf{X}\ ]]\!]\sigma I$

- For assignment, we proposed $\vdash \{B[a/\ \mathsf{X}\ ]\}\ \mathsf{X}\ :=\ a\{B\}$
- Recall $[\![\ \mathsf{X}\ :=\ a]\!]\sigma = \sigma[[\![a]\!]\sigma/\mathsf{X}]$
- Assume $\sigma \models^I B[a/\ \mathsf{X}\ ]$. Is it true that $\sigma[[\![a]\!]\sigma/\mathsf{X}] \models^I B$?
- In fact, it is an equivalent way of saying the same thing
- Proof: induction on the complexity of B, we can guess?
- $\sigma \models^I (a_1\ ==\ a_2)[a/\ \mathsf{X}\ ]$ iff $[\![a_1[a/\ \mathsf{X}\ ]]\!]\sigma I = [\![a_2[a/\ \mathsf{X}\ ]]\!]\sigma I$
- On the other hand, $\sigma[[\![a]\!]\sigma/\mathsf{X}] \models^I (a_1\ ==\ a_2)$ iff $[\![a_1]\!](\sigma[[\![a]\!]\sigma/\mathsf{X}])I = [\![a_2]\!](\sigma[[\![a]\!]\sigma/\mathsf{X}])I$.

- For assignment, we proposed $\vdash \{B[a/\ \mathsf{X}\ ]\}\ \mathsf{X}\ :=\ a\{B\}$
- Recall $[\![\ \mathsf{X}\ :=\ a]\!]\sigma = \sigma[[\![a]\!]\sigma/\mathsf{X}]$
- Assume $\sigma \models^I B[a/\ \mathsf{X}\ ]$. Is it true that $\sigma[[\![a]\!]\sigma/\mathsf{X}] \models^I B$?
- In fact, it is an equivalent way of saying the same thing
- Proof: induction on the complexity of B, we can guess?
- $\sigma \models^I (a_1\ ==\ a_2)[a/\ \mathsf{X}\ ]$ iff $[\![a_1[a/\ \mathsf{X}\ ]]\!]\sigma I = [\![a_2[a/\ \mathsf{X}\ ]]\!]\sigma I$
- On the other hand, $\sigma[[\![a]\!]\sigma/\mathsf{X}] \models^I (a_1\ ==\ a_2)$ iff $[\![a_1]\!](\sigma[[\![a]\!]\sigma/\mathsf{X}])I = [\![a_2]\!](\sigma[[\![a]\!]\sigma/\mathsf{X}])I$.
- In order to show our equivalence, we need

- For assignment, we proposed $\vdash \{B[a/\ \mathsf{X}\ ]\}\ \mathsf{X}\ :=\ a\{B\}$
- Recall $[\![\ \mathsf{X}\ :=\ a]\!]\sigma = \sigma[[\![a]\!]\sigma/\mathsf{X}]$
- Assume $\sigma \models^I B[a/\ \mathsf{X}\ ]$. Is it true that $\sigma[[\![a]\!]\sigma/\mathsf{X}] \models^I B$?
- In fact, it is an equivalent way of saying the same thing
- Proof: induction on the complexity of B, we can guess?
- $\sigma \models^I (a_1\ ==\ a_2)[a/\ \mathsf{X}\ ]$ iff $[\![a_1[a/\ \mathsf{X}\ ]]\!]\sigma I = [\![a_2[a/\ \mathsf{X}\ ]]\!]\sigma I$
- On the other hand, $\sigma[[\![a]\!]\sigma/\mathsf{X}] \models^I (a_1\ ==\ a_2)$ iff $[\![a_1]\!](\sigma[[\![a]\!]\sigma/\mathsf{X}])I = [\![a_2]\!](\sigma[[\![a]\!]\sigma/\mathsf{X}])I$.
- In order to show our equivalence, we need

Lemma
For any a, a', σ, I *and* X, $[\![a']\!](\sigma[[\![a]\!]\sigma/\mathsf{X}])I = [\![a'[a/\ \mathsf{X}\ ]]\!]\sigma I$

- For assignment, we proposed $\vdash \{B[a/\,\mathsf{X}\,]\}\ \mathsf{X}\ :=\ a\{B\}$
- Recall $[\![\ \mathsf{X}\ :=\ a]\!]\sigma = \sigma[[\![a]\!]\sigma/\mathsf{X}]$
- Assume $\sigma \models^I B[a/\,\mathsf{X}\,]$. Is it true that $\sigma[[\![a]\!]\sigma/\mathsf{X}] \models^I B$?
- In fact, it is an equivalent way of saying the same thing
- Proof: induction on the complexity of B, we can guess?
- $\sigma \models^I (a_1 == a_2)[a/\,\mathsf{X}\,]$ iff $[\![a_1[a/\,\mathsf{X}\,]]\!]\sigma I = [\![a_2[a/\,\mathsf{X}\,]]\!]\sigma I$
- On the other hand, $\sigma[[\![a]\!]\sigma/\mathsf{X}] \models^I (a_1 == a_2)$ iff $[\![a_1]\!](\sigma[[\![a]\!]\sigma/\mathsf{X}])I = [\![a_2]\!](\sigma[[\![a]\!]\sigma/\mathsf{X}])I$.
- In order to show our equivalence, we need

Lemma
For any a, a', σ, I *and* X, $[\![a']\!](\sigma[[\![a]\!]\sigma/\mathsf{X}])I = [\![a'[a/\,\mathsf{X}\,]]\!]\sigma I$

- Exercise: prove this lemma and use it to finish the proof of soundness of the assignment rule

  All conceptually trivial, but a lot of writing involved!

- Recall our earlier failed attempts at assignment rule

- Recall our earlier failed attempts at assignment rule
- Exercise: show formally that $\not\vdash \{\top\}\ \mathtt{X}\ :=\ a\{\ \mathtt{X}\ ==\ a\}$ and $\not\vdash \{A\}\ \mathtt{X}\ :=\ a\{A[a/\ \mathtt{X}\ ]\}$

- Recall our earlier failed attempts at assignment rule
- Exercise: show formally that $\not\vdash \{\top\}\ \mathtt{X}\ :=\ a\{\ \mathtt{X}\ ==\ a\}$ and $\not\vdash \{A\}\ \mathtt{X}\ :=\ a\{A[a/\ \mathtt{X}\ ]\}$
- In fact, another rule I was suggesting is no better!

- Recall our earlier failed attempts at assignment rule
- Exercise: show formally that $\not\vdash \{\top\}\ \mathtt{X}\ \mathtt{:=}\ a\{\ \mathtt{X}\ \mathtt{==}\ a\}$ and $\not\vdash \{A\}\ \mathtt{X}\ \mathtt{:=}\ a\{A[a/\ \mathtt{X}\ ]\}$
- In fact, another rule I was suggesting is no better!
- Exercise: show formally that $\not\vdash \{A\}\ \mathtt{X}\ \mathtt{:=}\ a\{\bigvee_{z\in\mathbb{Z}} (A[\mathtt{z}/\mathtt{X}]\ \mathtt{and}\ \mathtt{z}\ \mathtt{==}\ a)\}$

- Recall our earlier failed attempts at assignment rule
- Exercise: show formally that $\not\models \{\top\}\ \mathtt{X}\ \mathtt{:=}\ a\{\ \mathtt{X}\ \mathtt{==}\ a\}$ and $\not\models \{A\}\ \mathtt{X}\ \mathtt{:=}\ a\{A[a/\ \mathtt{X}\ ]\}$
- In fact, another rule I was suggesting is no better!
- Exercise: show formally that $\not\models \{A\}\ \mathtt{X}\ \mathtt{:=}\ a\{\bigvee_{z\in\mathbb{Z}} (A[\mathtt{z}/\mathtt{X}]\ \mathtt{and}\ \mathtt{z}\ \mathtt{==}\ a)\}$
- Can we save this idea?

- Recall our earlier failed attempts at assignment rule
- Exercise: show formally that $\not\models \{\top\}\ \mathtt{X}\ \mathtt{:=}\ a\{\ \mathtt{X}\ \mathtt{==}\ a\}$ and $\not\models \{A\}\ \mathtt{X}\ \mathtt{:=}\ a\{A[a/\ \mathtt{X}\ ]\}$
- In fact, another rule I was suggesting is no better!
- Exercise: show formally that $\not\models \{A\}\ \mathtt{X}\ \mathtt{:=}\ a\{\bigvee_{z\in\mathbb{Z}}(A[\mathtt{z}/\mathtt{X}]\ \mathtt{and}\ \mathtt{z}\ \mathtt{==}\ a)\}$
- Can we save this idea?
- Exercise: show that $\models \{A\}\ \mathtt{X}\ \mathtt{:=}\ a\{\bigvee_{z\in\mathbb{Z}}(A[\mathtt{z}/\mathtt{X}]\ \mathtt{and}\ \mathtt{X}\ \mathtt{==}\ a[\mathtt{z}/\mathtt{X}])\}$.

- Recall our earlier failed attempts at assignment rule
- Exercise: show formally that $\not\models \{\top\}\ \mathtt{X}\ \mathtt{:=}\ a\{\ \mathtt{X}\ \mathtt{==}\ a\}$ and $\not\models \{A\}\ \mathtt{X}\ \mathtt{:=}\ a\{A[a/\ \mathtt{X}\ ]\}$
- In fact, another rule I was suggesting is no better!
- Exercise: show formally that $\not\models \{A\}\ \mathtt{X}\ \mathtt{:=}\ a\{\bigvee_{z\in\mathbb{Z}}(A[\mathtt{z}/\mathtt{X}]\ \mathtt{and}\ \mathtt{z}\ \mathtt{==}\ a)\}$
- Can we save this idea?
- Exercise: show that $\models \{A\}\ \mathtt{X}\ \mathtt{:=}\ a\{\bigvee_{z\in\mathbb{Z}}(A[\mathtt{z}/\mathtt{X}]\ \mathtt{and}\ \mathtt{X}\ \mathtt{==}\ a[\mathtt{z}/\mathtt{X}])\}$.
- Let us also try using LogVar to store the original value.

  Exercise: show that $\models \{A\ \mathtt{and}\ \mathtt{X}\ \mathtt{==}\ i\}\ \mathtt{X}\ \mathtt{:=}\ a\{A[i/\mathtt{X}]\ \mathtt{and}\ \mathtt{X}\ \mathtt{==}\ a[i/\mathtt{X}]\}$

- Recall our earlier failed attempts at assignment rule
- Exercise: show formally that $\not\models \{\top\}\ \mathtt{X}\ \mathtt{:=}\ a\{\ \mathtt{X}\ \mathtt{==}\ a\}$ and $\not\models \{A\}\ \mathtt{X}\ \mathtt{:=}\ a\{A[a/\ \mathtt{X}\ ]\}$
- In fact, another rule I was suggesting is no better!
- Exercise: show formally that $\not\models \{A\}\ \mathtt{X}\ \mathtt{:=}\ a\{\bigvee_{z\in\mathbb{Z}}(A[\mathtt{z}/\mathtt{X}]\ \mathtt{and}\ \mathtt{z}\ \mathtt{==}\ a)\}$
- Can we save this idea?
- Exercise: show that $\models \{A\}\ \mathtt{X}\ \mathtt{:=}\ a\{\bigvee_{z\in\mathbb{Z}}(A[\mathtt{z}/\mathtt{X}]\ \mathtt{and}\ \mathtt{X}\ \mathtt{==}\ a[\mathtt{z}/\mathtt{X}])\}$.
- Let us also try using LogVar to store the original value.

  Exercise: show that $\models \{A\ \mathtt{and}\ \mathtt{X}\ \mathtt{==}\ i\}\ \mathtt{X}\ \mathtt{:=}\ a\{A[i/\mathtt{X}]\ \mathtt{and}\ \mathtt{X}\ \mathtt{==}\ a[i/\mathtt{X}]\}$
- We have a variant using substitution, another one using infinitary operators and now this. Does it matter which one we choose? Which one is *most general*?

- For sequencing, we have seen a proposal too:

$$\frac{\vdash \{A_1\}c_1\{A_2\} \qquad \vdash \{A_2\}c_2\{A_3\}}{\vdash \{A_1\}c_1 \,;\, c_2\{A_3\}}$$

- For sequencing, we have seen a proposal too:

$$\frac{\vdash \{A_1\}c_1\{A_2\} \qquad \vdash \{A_2\}c_2\{A_3\}}{\vdash \{A_1\}c_1 \,;\, c_2\{A_3\}}$$

- Recall: $[\![c_1 \;;\; c_2]\!] := [\![c_2]\!] \circ [\![c_1]\!]$

- For sequencing, we have seen a proposal too:

$$\frac{\vdash \{A_1\}c_1\{A_2\} \qquad \vdash \{A_2\}c_2\{A_3\}}{\vdash \{A_1\}c_1 \,;\, c_2\{A_3\}}$$

- Recall: $[\![c_1 \;;\; c_2]\!] := [\![c_2]\!] \circ [\![c_1]\!]$
- This rule preserves validity (is sound)

- For sequencing, we have seen a proposal too:

$$\frac{\vdash \{A_1\}c_1\{A_2\} \qquad \vdash \{A_2\}c_2\{A_3\}}{\vdash \{A_1\}c_1 \,;\, c_2\{A_3\}}$$

- Recall: $[\![c_1 \;;\; c_2]\!] := [\![c_2]\!] \circ [\![c_1]\!]$
- This rule preserves validity (is sound)
- Assume

  (1) $\models \{A_1\}c_1\{A_2\}$, (2) $\models \{A_2\}c_2\{A_3\}$ and (3) $\sigma \models^I A_1$

- For sequencing, we have seen a proposal too:

$$\frac{\vdash \{A_1\}c_1\{A_2\} \qquad \vdash \{A_2\}c_2\{A_3\}}{\vdash \{A_1\}c_1 \,;\, c_2\{A_3\}}$$

- Recall: $[\![c_1 \;;\; c_2]\!] := [\![c_2]\!] \circ [\![c_1]\!]$
- This rule preserves validity (is sound)
- Assume

  (1) $\models \{A_1\}c_1\{A_2\}$, (2) $\models \{A_2\}c_2\{A_3\}$ and (3) $\sigma \models^I A_1$
- Then (3) and (1) yield (4) $[\![c_1]\!]\sigma \models^I A_2$

- For sequencing, we have seen a proposal too:

$$\frac{\vdash \{A_1\}c_1\{A_2\} \qquad \vdash \{A_2\}c_2\{A_3\}}{\vdash \{A_1\}c_1 \,;\, c_2\{A_3\}}$$

- Recall: $[\![c_1 \;;\; c_2]\!] := [\![c_2]\!] \circ [\![c_1]\!]$
- This rule preserves validity (is sound)
- Assume

  (1) $\models \{A_1\}c_1\{A_2\}$, (2) $\models \{A_2\}c_2\{A_3\}$ and (3) $\sigma \models^I A_1$
- Then (3) and (1) yield (4) $[\![c_1]\!]\sigma \models^I A_2$
- Finally, (4) and (2) yield $[\![c_2]\!]([\![c_1]\!]\sigma) \models^I A_3$

- For conditionals, we proposed

$$\frac{\vdash \{A_1 \wedge b\} c_1 \{A_2\} \qquad \vdash \{A_1 \wedge \neg b\} c_2 \{A_2\}}{\vdash \{A_1\}\ \texttt{IF}\ b\ \texttt{THEN}\ c_1\ \texttt{ELSE}\ c_2\ \texttt{ENDIF}\ \{A_2\}}$$

  Note how we promote here b to an assertion, using the fact that BExp is a subset of AssertHo!

- For conditionals, we proposed

$$\frac{\vdash \{A_1 \wedge b\} c_1 \{A_2\} \qquad \vdash \{A_1 \wedge \neg b\} c_2 \{A_2\}}{\vdash \{A_1\}\ \texttt{IF}\ b\ \texttt{THEN}\ c_1\ \texttt{ELSE}\ c_2\ \texttt{ENDIF}\ \{A_2\}}$$

  Note how we promote here b to an assertion, using the fact that BExp is a subset of AssertHo!

- Exercise: prove soundness

- For conditionals, we proposed

$$\frac{\vdash \{A_1 \wedge b\} c_1 \{A_2\} \qquad \vdash \{A_1 \wedge \neg b\} c_2 \{A_2\}}{\vdash \{A_1\}\ \texttt{IF}\ b\ \texttt{THEN}\ c_1\ \texttt{ELSE}\ c_2\ \texttt{ENDIF}\ \{A_2\}}$$

  Note how we promote here b to an assertion, using the fact that BExp is a subset of AssertHo!

- Exercise: prove soundness
- We would like to be able to show that we can derive the trivial rule from this

- For conditionals, we proposed

$$\frac{\vdash \{A_1 \wedge b\} c_1 \{A_2\} \qquad \vdash \{A_1 \wedge \neg b\} c_2 \{A_2\}}{\vdash \{A_1\}\ \texttt{IF}\ b\ \texttt{THEN}\ c_1\ \texttt{ELSE}\ c_2\ \texttt{ENDIF}\ \{A_2\}}$$

  Note how we promote here b to an assertion, using the fact that BExp is a subset of AssertHo!

- Exercise: prove soundness
- We would like to be able to show that we can derive the trivial rule from this
- And we still have the problem of, e.g., comparing various forms of assignment rule...

- For WHILE we proposed $$\frac{\vdash \{A \wedge b\} c \{A\}}{\vdash \{A\}\ \texttt{WHILE}\ b\ \texttt{DO}\ c\ \texttt{END}\ \{A \wedge \neg b\}}$$

  Note again we promote here b to an assertion, using the fact that BExp is a subset of AssertHo

- For WHILE we proposed $$\frac{\vdash \{A \wedge b\} c \{A\}}{\vdash \{A\}\ \texttt{WHILE}\ b\ \texttt{DO}\ c\ \texttt{END}\ \{A \wedge \neg b\}}$$

  Note again we promote here b to an assertion, using the fact that BExp is a subset of AssertHo

- As you know, such an A is called invariant of the loop

- For WHILE we proposed $$\frac{\vdash \{A \wedge b\} c \{A\}}{\vdash \{A\}\ \texttt{WHILE}\ b\ \texttt{DO}\ c\ \texttt{END}\ \{A \wedge \neg b\}}$$

  Note again we promote here b to an assertion, using the fact that BExp is a subset of AssertHo

- As you know, such an A is called invariant of the loop
- Exercise: prove soundness ...

- For WHILE we proposed $$\frac{\vdash \{A \wedge b\} c \{A\}}{\vdash \{A\}\ \texttt{WHILE}\ b\ \texttt{DO}\ c\ \texttt{END}\ \{A \wedge \neg b\}}$$

  Note again we promote here b to an assertion, using the fact that BExp is a subset of AssertHo
- As you know, such an A is called invariant of the loop
- Exercise: prove soundness ...
- ...actually no, this isn't just an exercise. We need to do it now

- We have to show that whenever $\vDash \{A \wedge b\} c \{A\}$, it's true that $\vDash \{A\}$`WHILE` b `DO` c `END`$\{A \wedge \neg b\}$

- We have to show that whenever $\vDash \{A \wedge b\} c \{A\}$, it's true that $\vDash \{A\}$`WHILE` b `DO` c `END`$\{A \wedge \neg b\}$
- That is, assume

  $$(*) \text{ for any } \sigma, I,\ \sigma \vDash^I A \wedge b \text{ implies } [\![c]\!]\sigma \vDash^I A.$$

  Is it true that whenever $\sigma \vDash^I A$, it holds that

  $$\bigcup_{n \in \mathbb{N}} \Phi^n_{b,c}(\varnothing)(\sigma) \vDash^I A \wedge \neg b, \text{ where, recall,}$$

  $$\Phi_{b,c}(f)(\sigma) := \begin{cases} \sigma & \text{if } [\![b]\!]\sigma = \bot \\ f([\![c]\!]\sigma) & \text{if } [\![b]\!]\sigma = \top \end{cases} \qquad ?$$

Recall ...

Lemma (WHILE helper)
for any n', σ, b, c if $\Phi_{b,c}^{n'+1}(\varnothing)(\sigma)$ is defined, then there exists $n \leq n'$ s.t.

(a) $[\![c]\!]^n\sigma = [\![$ `WHILE` b `DO` c `END` $]\!]\sigma$ *and*

(b) $[\![c]\!]^n\sigma \models^I \neg b$ *and*

(c) *for all* $k < n$, $[\![c]\!]^k\sigma \models^I b$

- Consequence was formalized by a single rule

$$\frac{\vDash A' \to A, \quad \vdash \{A\}c\{B\}, \quad \vDash B \to B'}{\vdash \{A'\}c\{B'\}}$$

- Consequence was formalized by a single rule

$$\frac{\vDash A' \to A, \quad \vdash \{A\}c\{B\}, \quad \vDash B \to B'}{\vdash \{A'\}c\{B'\}}$$

- Proof of soundness: trivial.

- Consequence was formalized by a single rule

$$\frac{\vDash A' \to A, \quad \vdash \{A\}c\{B\}, \quad \vDash B \to B'}{\vdash \{A'\}c\{B'\}}$$

- Proof of soundness: trivial.
- But the rule has semantic judgements hardwired into it!

- Can we (or should we) do any better? Is there an inference system for assertions?

- Can we (or should we) do any better? Is there an inference system for assertions?
- And, for that matter ...can we hope for a complete set of rules for PCAs?

- Can we (or should we) do any better? Is there an inference system for assertions?
- And, for that matter ...can we hope for a complete set of rules for PCAs?
- Can we have a finitary/decidable set of axioms and rules for both assertions and PCAs s.t. $\vdash \{A\}c\{B\}$ iff $\vDash \{A\}c\{B\}$?

- Can we (or should we) do any better? Is there an inference system for assertions?
- And, for that matter ...can we hope for a complete set of rules for PCAs?
- Can we have a finitary/decidable set of axioms and rules for both assertions and PCAs s.t. $\vdash \{A\}c\{B\}$ iff $\vDash \{A\}c\{B\}$?
- That would imply valid PCAs are recursively enumerable

- Can we (or should we) do any better? Is there an inference system for assertions?
- And, for that matter ...can we hope for a complete set of rules for PCAs?
- Can we have a finitary/decidable set of axioms and rules for both assertions and PCAs s.t. $\vdash \{A\}c\{B\}$ iff $\vDash \{A\}c\{B\}$?
- That would imply valid PCAs are recursively enumerable
- Now consider $\{\texttt{true}\}c\{\texttt{false}\}$ ...

- Can we (or should we) do any better? Is there an inference system for assertions?
- And, for that matter ...can we hope for a complete set of rules for PCAs?
- Can we have a finitary/decidable set of axioms and rules for both assertions and PCAs s.t. $\vdash \{A\}c\{B\}$ iff $\vDash \{A\}c\{B\}$?
- That would imply valid PCAs are recursively enumerable
- Now consider $\{\texttt{true}\}c\{\texttt{false}\}$ ...
- It is valid if c diverges on all states: the halting problem!

- Can we (or should we) do any better? Is there an inference system for assertions?
- And, for that matter ...can we hope for a complete set of rules for PCAs?
- Can we have a finitary/decidable set of axioms and rules for both assertions and PCAs s.t. $\vdash \{A\}c\{B\}$ iff $\vDash \{A\}c\{B\}$?
- That would imply valid PCAs are recursively enumerable
- Now consider $\{\texttt{true}\}c\{\texttt{false}\}$ ...
- It is valid if c diverges on all states: the halting problem!
- Similarly, our assertion language includes quantified arithmetic $\Rightarrow$ Gödel's Incompleteness Theorem!

- With the consequence rule involving semantic validity, we can only talk about relative completeness

- With the consequence rule involving semantic validity, we can only talk about relative completeness
- That is, to check all valid applications of the consequence rule, we would need an oracle for all valid arithmetical statements

- With the consequence rule involving semantic validity, we can only talk about relative completeness
- That is, to check all valid applications of the consequence rule, we would need an oracle for all valid arithmetical statements
- Would such a completeness result be useless?

- With the consequence rule involving semantic validity, we can only talk about relative completeness
- That is, to check all valid applications of the consequence rule, we would need an oracle for all valid arithmetical statements
- Would such a completeness result be useless?
- Sensibly large fragments of arithmetic are decidable

  Examples: Skolem's primitive recursive arithmetic (PRA) or a chosen fragment of Presburger's arithmetic

- With the consequence rule involving semantic validity, we can only talk about relative completeness
- That is, to check all valid applications of the consequence rule, we would need an oracle for all valid arithmetical statements
- Would such a completeness result be useless?
- Sensibly large fragments of arithmetic are decidable

  Examples: Skolem's primitive recursive arithmetic (PRA) or a chosen fragment of Presburger's arithmetic
- Good for useful heuristics to use in our automated tools

  And still better if/when we use proof assistants, where humans guide machines

weakest liberal preconditions

- We used rules for assignment, sequencing, and of course consequence and while loops

- We used rules for assignment, sequencing, and of course consequence and while loops
- The applications of consequence rule are very easy to verify to whichever tractable subsystem of arithmetic we pick

  the only possible exception being this infinitary definition of factorial

- We used rules for assignment, sequencing, and of course consequence and while loops
- The applications of consequence rule are very easy to verify to whichever tractable subsystem of arithmetic we pick

  the only possible exception being this infinitary definition of factorial
- We see that we have a sound system which can be used for annotating programs

- We used rules for assignment, sequencing, and of course consequence and while loops
- The applications of consequence rule are very easy to verify to whichever tractable subsystem of arithmetic we pick

  the only possible exception being this infinitary definition of factorial
- We see that we have a sound system which can be used for annotating programs
- We can return now to question of (relative) completeness: are our rules as general as possible?

  again, assuming somebody gives us an oracle for arithmetic ...

- Here come weakest (liberal) preconditions

- Here come weakest (liberal) preconditions
- Define $WLP_I(c, B) := \{\sigma \in \mathsf{States}_{\mathsf{IMP}} \mid [\![c]\!]\sigma \models^I B\}$

  This is the semantic version of weakest liberal preconditions

  Ordinary (total) weakest preconditions are obtained with total $\perp\!\!\!\perp$-convention for $\models^I$

- Here come weakest (liberal) preconditions
- Define $WLP_I(c, B) := \{\sigma \in \mathsf{States}_{\mathsf{IMP}} \mid [\![c]\!]\sigma \models^I B\}$

  This is the semantic version of weakest liberal preconditions

  Ordinary (total) weakest preconditions are obtained with total $\lightning$-convention for $\models^I$
- Note: $\models^I \{A\}c\{B\}$ iff $[\![A]\!]_I \subseteq WLP_I(c, B)$

  In this way, we switch perspective to predicate transformer semantics

- For any pair c, B, can we possibly find $wlp(c, B)$ s.t. for any I, $[\![wlp(c, B)]\!]_I = WLP_I(c, B)$?

- For any pair c, B, can we possibly find $wlp(c, B)$ s.t. for any I, $[\![wlp(c, B)]\!]_I = WLP_I(c, B)$?
- Further, can we show that $\vdash \{wlp(c, B)\}c\{B\}$?

- For any pair c, B, can we possibly find $wlp(c, B)$ s.t. for any I, $[\![wlp(c, B)]\!]_I = WLP_I(c, B)$?
- Further, can we show that $\vdash \{wlp(c, B)\}c\{B\}$?
- Assertion languages allowing such a function are expressive

- For any pair c, B, can we possibly find $wlp(c, B)$ s.t. for any I, $[\![wlp(c, B)]\!]_I = WLP_I(c, B)$?
- Further, can we show that $\vdash \{wlp(c, B)\}c\{B\}$?
- Assertion languages allowing such a function are expressive
- Note that in the presence of the consequence rule, expressivity trivially implies (relative) completeness: WHY?