$[\![\textbf{while } B \textbf{ do } C]\!]$: States → States

Operationally

while B do C ≡ if B then (C; while B do C) else skip.

So we want

$[\![\text{while } B \text{ do } C]\!] = [\![\text{if } B \text{ then } (C; \text{while } B \text{ do } C) \text{ else skip}]\!]$

$= \lambda s.\ \text{if}\ ([\![B]\!]s, [\![C; \text{while } B \text{ do } C]\!]s, [\![\text{skip}]\!]s)$

$\circledast\ = \lambda s.\ \text{if}\ ([\![B]\!]s, [\![\text{while } B \text{ do } C]\!]([\![C]\!]s), s)$

Consider

$f_{[\![B]\!],[\![C]\!]} : (\text{State} \to \text{State}) \to (\text{State} \to \text{State})$

$$f_{[\![B]\!],[\![C]\!]} =^{df} \lambda w : State \to State$$

$$\lambda s : State .$$

$$if\,([\![B]\!]s, w([\![C]\!]s), s)$$

NB: We want $[\![\text{while } B \text{ do } C]\!]$ satisfies the equation

$$[\![\text{while } B \text{ do } C]\!] \overset{(*)}{=} f_{[\![B]\!],[\![C]\!]}([\![\text{while } B \text{ do } C]\!])$$

$\boxed{?}$ Does there always exist a unique state transformer satisfying ⊛ ?

$\boxed{!}$ If so then we define $[\![\text{while } B \text{ do } C]\!]$ as the unique fixed point of $f_{[\![B]\!],[\![C]\!]}$!

$\boxed{\text{ans}}$ No! $[\![\text{while true do skip}]\!]$

$$f_{[\![\text{true}]\!],[\![\text{skip}]\!]} = \lambda w.\lambda s.\ \text{if}(\text{true}, w(s), s)$$
$$= \lambda w.\lambda s.\ w(s)$$
$$= \text{the identity}$$

NB: A fixed of a function $h : A \to A$ is $a \in A$ s.t. $h(a) = a$

Every state transformer is a fixed point of $f_{[\![\text{true}]\!],[\![\text{skip}]\!]}$, so how do we single out the fixpoint $[\![\text{while true do skip}]\!]$?

$: \text{State} \to \text{State}$

intuitively

$[\![\text{while true do skip}]\!](s) \uparrow \quad \forall s \in \text{State}$

Notation for undefined.

$[\![\text{while true do skip}]\!] = \bot$ — the totally undefined function.

Amongst all fixed points of $f_{[\![B]\!],[\![C]\!]}$ we want to calculate the fixed point corresponding to $[\![\text{while } B \text{ do } C]\!]$!

(by approximation

$$[\![\text{while } B \text{ do } C]\!]_0 = \bot$$

$$[\![\text{while } B \text{ do } C]\!]_1 = f_{[\![B]\!],[\![C]\!]}\left([\![\text{while } B \text{ do } C]\!]_0\right)$$

$$= \lambda s.\ \text{if}\left([\![B]\!]s, \uparrow, s\right)$$

$$= \lambda s. \begin{cases} s & [\![B]\!](s) = \text{false} \\ \uparrow & \text{otw} \end{cases}$$

$$[\![\text{while } B \text{ do } C]\!]_2 = f_{[\![B]\!],[\![C]\!]}\left([\![\text{while } B \text{ do } C]\!]_1\right)$$

$$= \lambda s.\ \text{if}\left([\![B]\!](s),\ [\![\text{while } B \text{ do } C]\!]_1([\![C]\!]s),\ s\right)$$

$$= \lambda s. \begin{cases} s & [\![B]\!](s) = \text{false} \\ [\![\text{while } B \text{ do } C]\!]_1([\![C]\!]s) & \text{otw} \end{cases}$$

$$= \lambda s. \begin{cases} s & \text{if } [\![B]\!](s) = \text{false} \\ [\![C]\!](s) & \text{if } [\![B]\!](s) = \text{true and } [\![B]\!]([\![C]\!]s) = \text{false} \\ \uparrow & \text{otw} \end{cases}$$

TWO KEY INGREDIENTS OF THE THEORY

Recap

$$[\![\text{while } B \text{ do } C]\!]_0 \stackrel{\text{def}}{=} \perp$$

$$\sqsubseteq [\![\text{while } B \text{ do } C]\!]_1 \stackrel{\text{def}}{=} f_{[\![B]\!],[\![C]\!]}(\perp)$$

$$\sqsubseteq [\![\text{while } B \text{ do } C]\!]_2 \stackrel{\text{def}}{=} f^2_{[\![B]\!],[\![C]\!]}(\perp)$$

$$\sqsubseteq \cdots [\![\text{while } B \text{ do } C]\!]_n \sqsubseteq \cdots$$

$$[\![\text{while } B \text{ do } C]\!]_n \stackrel{\text{def}}{=} f^n_{[\![B]\!],[\![C]\!]}(\perp)$$

less information than

the join of all the information

$$[\![\text{while } B \text{ do } C]\!] \stackrel{\text{def}}{=} \bigsqcup_{n \geq 0} f^n_{[\![B]\!],[\![C]\!]}(\perp)$$

$\boxed{?}$ How come $\bigsqcup_{n \geq 0} f_{\langle\!\langle B\rangle\!\rangle, \llbracket C\rrbracket}^{n}(\perp)$

is a fixed point of $f_{\langle\!\langle B\rangle\!\rangle, \llbracket C\rrbracket}$?

That is,

$$f_{\langle\!\langle B\rangle\!\rangle, \llbracket C\rrbracket}\left(\bigsqcup_{n \geq 0} f_{\langle\!\langle B\rangle\!\rangle, \llbracket C\rrbracket}^{n}(\perp)\right)$$

$$= \bigsqcup_{n \geq 0} f_{\langle\!\langle B\rangle\!\rangle, \llbracket C\rrbracket}^{n}(\perp) .$$

Fixed point property of $[\![\mathbf{while}\ B\ \mathbf{do}\ C]\!]$

$$[\![\mathbf{while}\ B\ \mathbf{do}\ C]\!] = f_{[\![B]\!],[\![C]\!]}([\![\mathbf{while}\ B\ \mathbf{do}\ C]\!])$$

where, for each $b : State \to \{true, false\}$ and $c : State \rightharpoonup State$, we define

$$f_{b,c} : (State \rightharpoonup State) \to (State \rightharpoonup State)$$

as

$$f_{b,c} = \lambda w \in (State \rightharpoonup State).\, \lambda s \in State.\, if\big(b(s), w(c(s)), s\big).$$

- Why does $w = f_{[\![B]\!],[\![C]\!]}(w)$ have a solution?
- What if it has several solutions—which one do we take to be $[\![\mathbf{while}\ B\ \mathbf{do}\ C]\!]$?

Approximating $[\![\textbf{while}\ B\ \textbf{do}\ C]\!]$

$$f_{[\![B]\!],[\![C]\!]}{}^{n}(\bot)$$

$$= \lambda s \in State. \begin{cases} [\![C]\!]^k(s) & \text{if } \exists\, 0 \leq k < n.\, [\![B]\!]([\![C]\!]^k(s)) = \mathit{false} \\ & \text{and } \forall\, 0 \leq i < k.\, [\![B]\!]([\![C]\!]^i(s)) = \mathit{true} \\ \uparrow & \text{if } \forall\, 0 \leq i < n.\, [\![B]\!]([\![C]\!]^i(s)) = \mathit{true} \end{cases}$$

Partial order $\leq$ on a set is a binary relation

That is: (reflexive) $x \leq x$, (transitive) $x \leq y \wedge y \leq z \Rightarrow x \leq z$

$$D \stackrel{\text{def}}{=} (State \rightharpoonup State)$$

(antisymmetric) $x \leq y \wedge y \leq x \Rightarrow x = y$

- **Partial order $\sqsubseteq$ on D:**

$w \sqsubseteq w'$ iff for all $s \in State$, if w is defined at s then so is w' and moreover $w(s) = w'(s)$.

iff the graph of w is included in the graph of w'.

- **Least element $\perp \in D$ w.r.t. $\sqsubseteq$:**

$\perp$ = totally undefined partial function

= partial function with empty graph

(satisfies $\perp \sqsubseteq w$, for all $w \in D$).

Topic 2

Least Fixed Points

The way to single out the computationally meaningful fixpoints.

E.g.

$D = (State \to State)_\bot$

Thesis

All domains of computation are partial orders with a least element.

E.g. $(A \to B)_\bot$

for A, B sets.

Thesis

All domains of computation are partial orders with a least element.

f monotonic

All computable functions are mononotic.

$$x \sqsubseteq y \Rightarrow f(x) \sqsubseteq f(y)$$

more information on the input gives more information on the output