Reminder of syntax:

$P ::= C$ (C for command)

$C ::= C_1 ; C_2 \mid I = E \mid$ if E then C_1 else $C_2 \mid$
while E do C

$E ::= I \mid N \mid E_1 + E_2$

Corresponding to each non-terminal, there is a function with the same name

Let $\mathcal{E}$ be function that maps expression syntax to semantics. Each alternative has an equation involving the function $\mathcal{E}$ and the corresponding syntax.

$$\mathcal{E}[\![I]\!] = \lambda s . s([\![I]\!])$$

↑ "store"

the store, s is a function that maps $I \to \mathbb{N}$ (natural numbers)

$\mathbb{N}$ is set $\{$zero, one, two...$\}$

Note the difference between $\emptyset$ (syntax) and zero (semantics)

$$E[\![N]\!] = \lambda s.\, N[\![N]\!]$$

e.g. $E[\![1]\!] = \lambda s.\, N[\![1]\!] = \lambda s.\, one$

i.e. N maps numerals to numbers

Arithmetic

$$E[\![E_1 + E_2]\!] = \lambda s.\, (E[\![E_1]\!]\, s)\ plus\ (E[\![E_2]\!]\, s)$$

e.g. $E[\![x + 2]\!]\, s_0$ where $s_0 = \{([\![x]\!], one)\}$

$= (\lambda s.\, (E[\![x]\!]\, s)\ plus\ (E[\![2]\!]\, s))\ s_0$

$= E[\![x]\!]\, s_0\ plus\ E[\![2]\!]\, s_0$

$= (\lambda s.\, s([\![x]\!]))\, s_0\ plus\ (\lambda s.\, N[\![2]\!])\, s_0$

$= s_0[\![x]\!]\ plus\ two$

$= one\ plus\ two$

$= three$

Commands (Statements)

$$C[\![I = E]\!] = \lambda s.\, s[[\![I]\!] \mapsto E[\![E]\!] s]$$

e.g. $s_0 = \{([\![x]\!], zero)\}$

$$C[\![y = x+1]\!]\, s_0$$

$$= (\lambda s.\, s[[\![y]\!] \mapsto E[\![x+1]\!] s])\, s_0$$

$$= s_0[[\![y]\!] \mapsto E[\![x+1]\!] s_0]$$

$(E[\![x]\!] s_0)$ plus $(E[\![1]\!] s_0)$

$s_0([\![y]\!])$ plus one

zero plus one

one

$$= s_1 = \{([\![x]\!], zero), ([\![y]\!], one)\}$$

Therefore assignment is store update.

$$C[\![c_1 ; c_2]\!] = \lambda s.\ C[\![c_2]\!](C[\![c_1]\!]s)$$

e.g. $C[\![x := 0 ; y := 1]\!]s_0$ where $s_0 = \{\}$

$= (\lambda s.\ C[\![y := 1]\!](C[\![x := \phi]\!]s))\ s_0$

$= C[\![y := 1]\!](C[\![x := \phi]\!]s_0)$

$C[\![x := \phi]\!]s_0$

$s_0[[\![x]\!] \mapsto \in[\![\phi]\!]s_0]$

$N[\![\phi]\!]s_0$

zero

$= s_1$ where $s_1 = \{([\![x]\!], zero)\}$

$= s_1[[\![y]\!] \mapsto \in[\![1]\!]s_1]$

$N[\![1]\!]s_1$

one

$= s_2$ where $s_2 = \{([\![x]\!], zero), ([\![y]\!], one)\}$

$C[\![\text{if } E \text{ then } C_1 \text{ else } C_2]\!] = \lambda s.$?

we need a conditional form in the λ calculus

$e_1 \rightarrow e_2 \,[]\, e_3$ where the e's are any expression

e_1 — test, e_2 — then, e_3 — else

$C[\![\text{if } E \text{ then } C_1 \text{ else } C_2]\!] =$

$\lambda s.((E[\![E]\!]s) = \text{zero}) \rightarrow C[\![C_2]\!]s \,[]\, C[\![C_1]\!]s$

i.e. zero corresponds to false, any other number corresponds to true

$$C[\![\text{while } E \text{ do } C]\!] =$$

$$\lambda s.\ (E[\![E]\!]s) = zero \rightarrow s\ [\!]\ C[\![C; \text{while } E \text{ do } C]\!]s$$

Lastly the program:

$$P[\![C]\!] = \lambda s.\ C[\![C]\!]s$$

$= C[\![\text{if } x+y \text{ then } z=3 \text{ else } z=4]\!]\, s_2$

$= (E[\![x+y]\!]\, s_2) = zero \rightarrow C[\![z=4]\!]s_2 \,[]\, C[\![z=3]\!]\, s_2$

$= (E[\![x]\!]s_2 \text{ plus } E[\![y]\!]s_2) = zero \rightarrow \ldots$

$= (zero \text{ plus } one) = zero \rightarrow \ldots$

$= (one = zero) \rightarrow C[\![z=4]\!]\, s_2 \,[]\, C[\![z=3]\!]\, s_2$

$= C[\![z=3]\!]\, s_2$

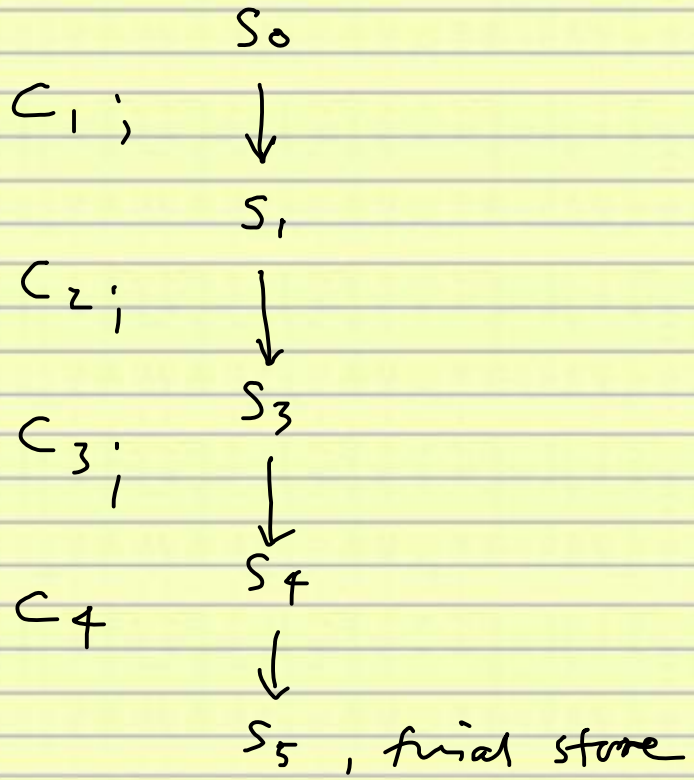
$= s_2[[\![z]\!] \mapsto three]$

$= s_3 \quad \text{where } s_3 = \{([\![x]\!], zero), ([\![y]\!], one), ([\![z]\!], three)\}$

What have we done?

We have shown that the semantics of a program is a sequence of updated stores. The final store is called the denotation of the program.

In summary, the method produces a denotation for a whole program (the final store), but also explains the sequence of updating the initial empty store



The advantages of this method:

1. λ calculus is very simple and well-defined
2. there is no real machine (λ calculus has a simple VM)