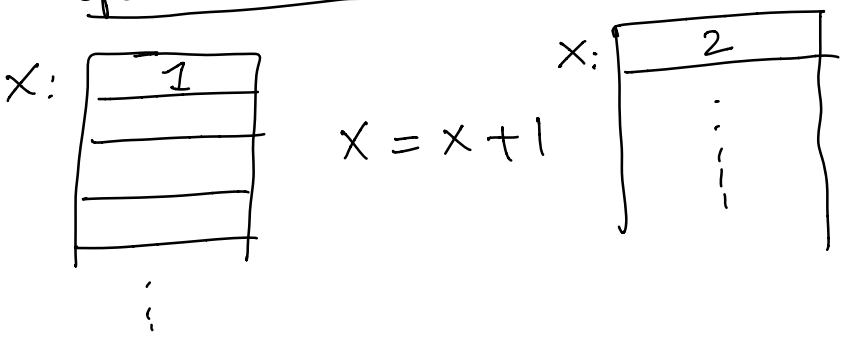
Semantics

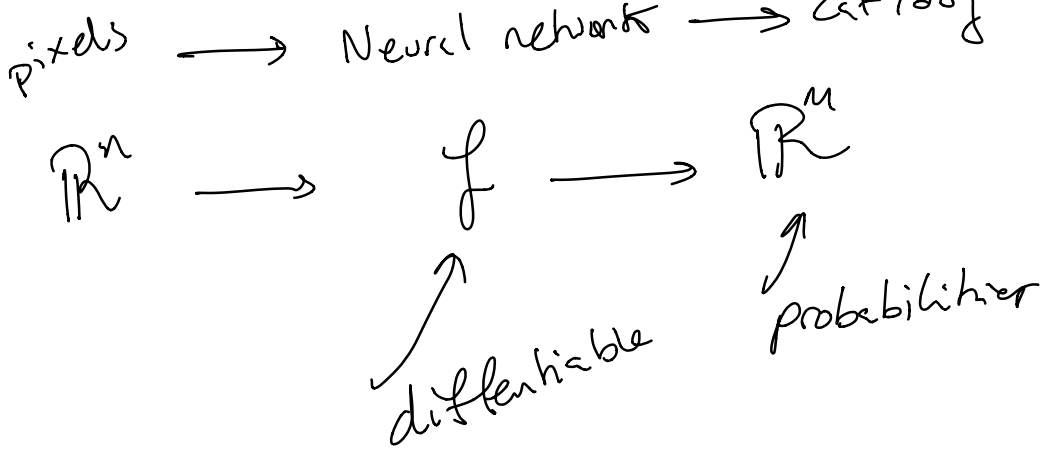
operational semantics



K - framework

In λ calculus, β reduction is an operational semantics

denotational semantics



axiomatic semantics

$$X = X + 1$$

if $x > 0$ then after executing $X = X + 1$

X is still > 0

Semantics with applications

arithmetic expressions

$$a := n \mid x \mid a_1 + a_2 \mid a_1 * a_2 \mid a_1 - a_2$$

Boolean expressions

$$b := \text{true} \mid \text{false} \mid a_1 = a_2 \mid a_1 \leq a_2 \mid \neg b_1 \mid b_1 \wedge b_2$$

$\neg$: NOT $\wedge$: AND

program

$$P := x := a \mid \text{skip} \mid P_1 ; P_2 \mid$$

if b then P_1 else P_2

| while b then P_1

Var is the set of all variables

a state $s : Var \to \mathbb{Z}$ (integers)

E.g. $s = \{ x \mapsto 10,\ y \mapsto 70,\ z \mapsto 200,\ \vdots \}$

$s[x \mapsto 0] = \{ x \mapsto 0,\ y \mapsto 70,\ z \mapsto 200 \}$

substitution

[x/0]

Semantics of expressions

$[\![a]\!] : State \to \mathbb{Z}$ (set of all possible states)

$[\![b]\!] : State \to \mathbb{B}$ ($\mathbb{B} = \{true, false\}$)

E.g. $[\![x+y]\!] = 0 + 70 = 70$

$[\![x = y]\!] = false$

Natural (or big-step) semantics

$\langle P, s \rangle \longrightarrow s'$

program — starting state — final state

skip

$\langle \text{skip}, s \rangle \longrightarrow s$

asn

$\langle x := a, s \rangle \longrightarrow s[x \to [\![a]\!](s)]$

sequential comp

$\langle P_1 ; P_2, s \rangle \longrightarrow s''$

where $\langle P_1, s \rangle \longrightarrow s'$

$\langle P_2, s' \rangle \longrightarrow s''$

alternative notation

$$\frac{\langle P_1, s\rangle \to s' \qquad \langle P_2, s'\rangle \to s''}{\langle P_1 ; P_2, s\rangle \to s''}$$

if (true) $\langle \text{if } b \text{ then } P_1 \text{ else } P_2, s\rangle \to s'$

if $[\![b]\!](s) = \text{true}$

and $\langle P_1, s\rangle \to s'$

$\langle \text{if } b \text{ then } P_1 \text{ else } P_2, s\rangle \to s'$

if $[\![b]\!](s) = \text{false}$

and $\langle P_2, s\rangle \to s'$

while (true) $\langle \text{while } b \text{ do } P, s \rangle \longrightarrow s''$

if $[\![b]\!](s) = \text{true}$

and $\langle P, s \rangle \longrightarrow s'$

and $\langle \text{while } b \text{ do } P, s' \rangle \longrightarrow s''$

while (false) $\langle \text{while } b \text{ do } P, s \rangle \longrightarrow s$

if $[\![b]\!](s) = \text{false}$

Eg $\langle z := x ; x := y ; y := z , s_0 \rangle$

$\langle z := x, s_0 \rangle \longrightarrow s_1 \qquad \langle x := y, s_1 \rangle \longrightarrow s_2$

$s_0 = \begin{cases} x \mapsto 5 \\ y \mapsto 7 \\ z \mapsto 0 \end{cases} \qquad s_1 = \begin{cases} x \mapsto 5 \\ y \mapsto 7 \\ z \mapsto 5 \end{cases} \qquad s_2 = \begin{cases} x \mapsto 7 \\ y \mapsto 7 \\ z \mapsto 5 \end{cases}$

Properties

two programs P_1 and P_2 are equivalent iff

forall s, s' $\langle P_1, s \rangle \rightarrow s'$ iff

$\langle P_2, s \rangle \rightarrow s'$

E.g. we can prove that

while b do P

is equivalent to

if b then

P ; while b do P

else

skip ;

Theorem : the semantics are deterministic

assume $\langle P, s \rangle \to s'$

if $\langle P, s \rangle \to s''$, then $s' = s''$

---

program semantics

$[\![P]\!] : State \to State$

this is a partial function

$$[\![P]\!](s) = \begin{cases} s' & \text{if } \langle P, s \rangle \to s' \\ \text{undef} & \text{o/w} \end{cases}$$

o/w ↑ non-termination

e.g. $[\![$ while true do $x := 1]\!](s) = \text{undef}$

$[\![a]\!] : State \to \mathbb{Z}$

$x + y + z$

Concurrent / shared memory setting

$\langle P, s \rangle \to \langle P', s' \rangle$