CS 520

Theory of Programming Language

05/05 – 05/12, 2021

Lambda Calculus ... Chap 10, Reynolds

1. Motivation / Background.

① Imperative computation. vs applicative computation.

fun. def. / fn appli

C, C++, Java, Python, Julia.

ho fns.

Ocaml, Haskell, Scala, Clojure.

(2) Lambda calculus.

(1) ..... core applicative/functional prog. lang.

(2) formal framework. .... eager-eval. lang. (Ocaml, Scala, Python)

for studying the design choices of PL.

lazy-eval. lang. (Haskell, Scala).
(call-by-name
call-by-need)

2. Syntax of Lambda calculus.

$\langle exp \rangle ::= \langle var \rangle \mid \langle exp \rangle \langle exp \rangle \mid \lambda \langle var \rangle . \langle exp \rangle$

lambda abstraction.

① examples.

(1) $(\lambda x. x)\ (\lambda z. z)$

$(\lambda x. (\lambda y. y\ x)\ z)\ (z\ w)$

$(\lambda z. z\ z)\ (\lambda z. z)$

(2) encoding.

: $\lambda x. \lambda y. x$ .... true

$\lambda x. \lambda y. y$ ... false.

:: $\lambda f. \lambda x. x$ .... 0 $f \mapsto f^0$.

$\lambda f. \lambda x. (f\ x)$ ... 1 $f \mapsto f^1$

$\lambda f. \lambda x. (f\ (f\ x))$ ... 2. $f \mapsto f^2$

(1)

(2) $FV : \langle exp \rangle \to 2^{\langle var \rangle}$.

(2) $\xi \in [\langle var \rangle \to \langle exp \rangle]$

$-/\xi : \langle exp \rangle \to \langle exp \rangle$

ex. define FV and $-/\xi$.

$FV(x) = \{x\}$

$FV(e_1\ e_2) = FV(e_1) \cup FV(e_2)$

$FV(\lambda x.e) = FV(e) - \{x\}$.

e/ξ

$x/\xi = \xi(x)$

$(e_1\ e_2)/\xi = (e_1/\xi)\ (e_2/\xi)$.

$(\lambda x.e)/\xi = (\lambda x_{new}.\ e/[\xi \mid x : x_{new}])$

$x_{new} \notin \bigcup_{w \in FV(e)-\{x\}} FV(\xi(w))$

(3) α-equivalence/renaming eq.

.... eq. rel. on $\langle exp \rangle$.

.... $(\lambda x.e)$ ... α-renaming ...

(3)

$e_1 \equiv_\alpha e_2$. or $e_1 \equiv e_2$ ... e_1 and e_2 are α-equivalent.

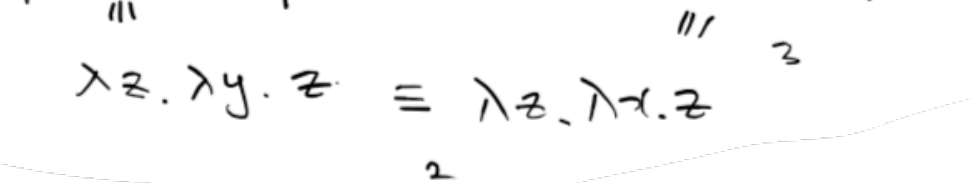
if we can obtain e_2 from e_1 by renaming bound variables in e_1

$\lambda x.e \longmapsto \lambda y.(e/x \to y)$ zero or multiple times.

$(y \notin FV(e) - \{x\})$

e.g. $(\lambda x.\, x)\ (\lambda z.\, z) \equiv (\lambda y.\, y)\ (\lambda v.\, v)$

$(\lambda x.\, \lambda y.\, x) \equiv (\lambda y.\, \lambda x.\, y)$



$\lambda z.\, \lambda y.\, z \equiv \lambda z.\, \lambda x.\, z$

(symbolic execution)

3. Contraction relation $\longrightarrow$ and reduction relation $\longrightarrow^*$ .... computation with expressions in lambda cal.

(evaluation relation $\Rightarrow$, $\overset{=}{\Rightarrow}$).

.... actual runtimes of PLs.

① $\longrightarrow \subseteq \langle exp \rangle \times \langle exp \rangle$ (instead of writing $(e_1, e_2) \in \longrightarrow$, we will write $e_1 \longrightarrow e_2$)

β-reduction

$$\frac{}{((\lambda x.\, e_1)\ e_2) \longrightarrow e_1/x \to e_2}$$

β-redex, redex.

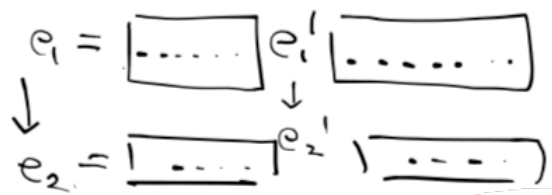
Renaming

$$\frac{e_1 \longrightarrow e_2 \quad e_2 \equiv e_2'}{e_1 \longrightarrow e_2'}$$

Contextual closure.

$$\frac{e_1' \longrightarrow e_2'}{e_1 \longrightarrow e_2}$$

where e_2 is obtained from e_1 by replacing e_1' inside e_1 by e_2'.



ex. Contract the LHS expressions as much as possible using $\rightarrow$.

(1) $(\lambda x.\, y)\ (\lambda z.\, z) \rightarrow y$

(2) $(\lambda x.\, x\ x)\ (\lambda z.\, z) \rightarrow (\lambda z.\, z)\ (\lambda z.\, z) \rightarrow (\lambda z.\, z)$

(3) $(\lambda x.\, (\lambda y.\, y\ z)\ z)\ (z\ w) \rightarrow (\lambda x.\, z\ z)\ (z\ w) \rightarrow (z\ z)$

$(\lambda x.\, (\lambda y.\, y\ z)\ z)\ (z\ w) \rightarrow (\lambda y.\, y\ z)\ z \rightarrow (z\ z)$

(4) $(\lambda u.\, \lambda v.\, v)\ ((\lambda x.\, x\ x)\ (\lambda x.\, x\ x))$

$\rightarrow \lambda v.\, v$

$(\lambda u.\, \lambda v.\, v)\ ((\lambda x.\, x\ x)\ (\lambda x.\, x\ x)) \rightarrow \ldots\ldots$

(2) e is a β-normal form. if e cannot be contracted.
($\nexists e'$ s.t. $e \longrightarrow e'$).

reduction.

(3) $\longrightarrow^*$ ... reflexive and transitive and λ-eq. closure of $\longrightarrow$.

$e \longrightarrow^* e'$ iff $e \equiv e'$ or $e \longrightarrow e_1 \longrightarrow e_2 \longrightarrow \cdots \longrightarrow e'$.

④ Q. Given e. (1) If $e \xrightarrow{*} e_1$ and e_1 is β-normal, and $e \xrightarrow{*} e_2$ and e_2 is β-normal. then are e_1 and e_2 α-equivalent?

Yes. ←

(2) Is it always possible to have a β-normal e_1 s.t. $e \xrightarrow{*} e_1$?

No. ← $(\lambda x.\ x\ x)\ (\lambda x.\ x\ x)$.

Yes. ← (3) What is a strategy of applying the contraction relation that gives us the β-normal form output of e, if such an output exists?

normal-order reduction:
outermost, leftmost red.