Today λ calculus evaluation

β-reduction

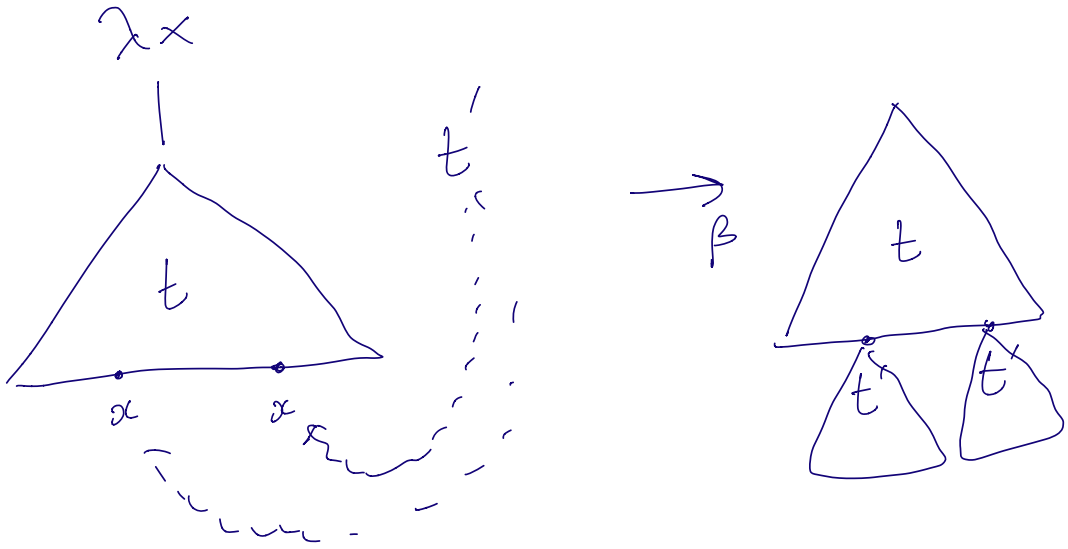
$(\lambda x.x)\ y \longrightarrow_\beta y$

identity combinator

$\longrightarrow_\beta$

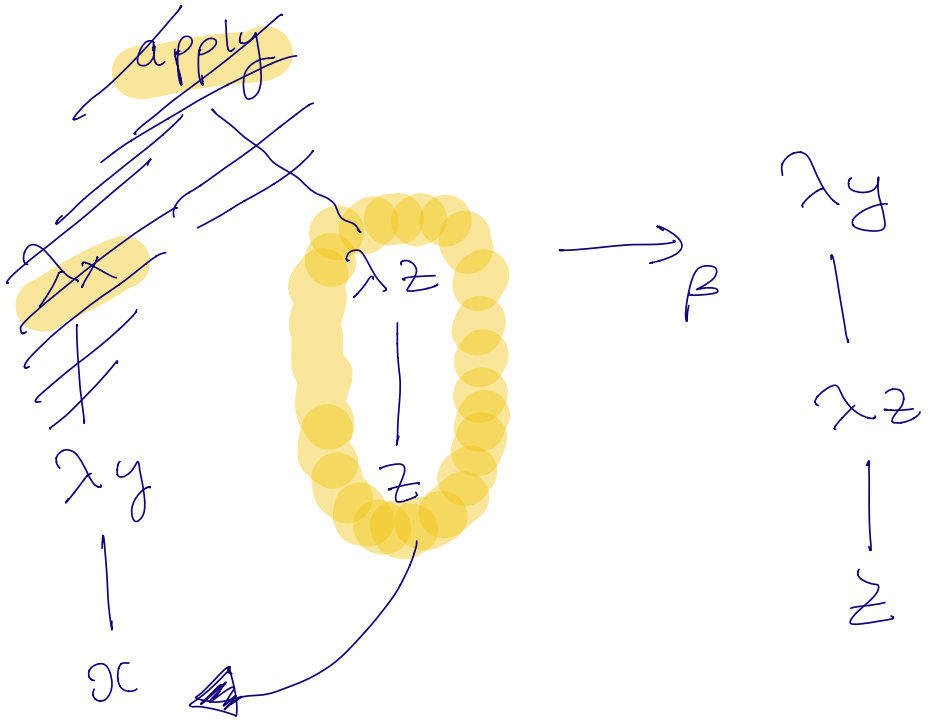
$(u\ r)\ (\lambda x.x)$

$(\lambda x . t)\ t'$



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

$\xrightarrow{\beta}\ \lambda y.(\lambda z.z)$



Reduction strategies

$id = (\lambda x \cdot x)$

① Full beta reduction

$id\ (id\ (\lambda z \cdot id\ z))$

$\xrightarrow{\beta}\ id\ (id\ (\lambda z \cdot z))$

$\xrightarrow{\beta}\ id\ (\lambda z \cdot z)$

$\xrightarrow{\beta}\ \lambda z \cdot z$ ← normal form

② Normal order reduction

NOR left-most, outer-most redex first

$$\underline{id}\ (\underline{id\ (\lambda z.\ id\ z)})$$

$$\xrightarrow{\beta}\ \underline{id}\ (\underline{\lambda z.\ id\ z})$$

$$\xrightarrow{\beta}\ \lambda z.\ \underline{id}\ z$$

$$\xrightarrow{\beta}\ \lambda z.\ z$$

Theorem: if t has a normal form t' following NOR, $t \xrightarrow[NOR]{} t'$

③ Call by name (CBN)

no reductions inside abstractions

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

$\rightarrow id\ (\lambda z.\ id\ z)$ ✗

$\rightarrow \lambda z.\ id\ z$

ALGOL 60

Haskell

call by need

lazy evaluation

④ Call by value

only outermost redexes reduced

redex reduced when its RHS has been reduced

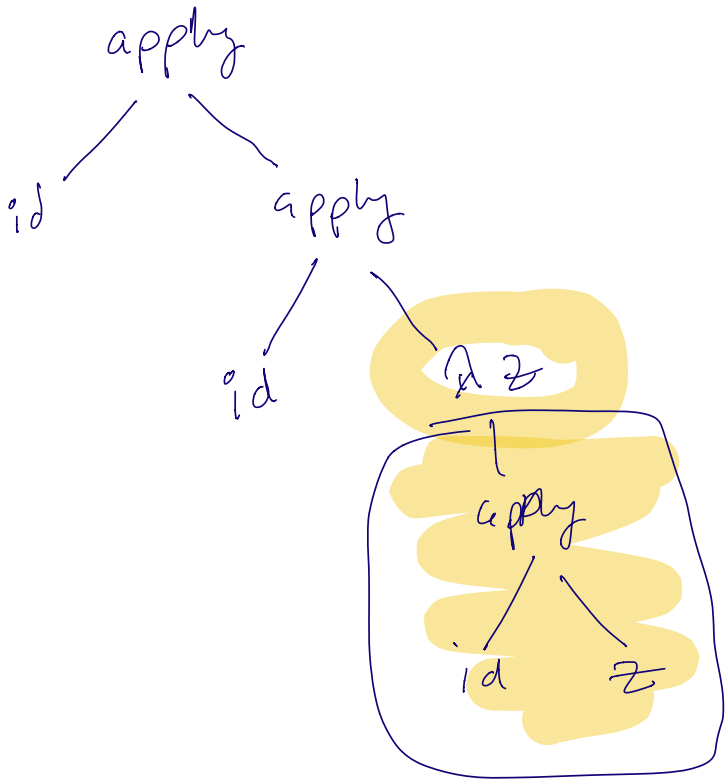
$id\ (id\ (\lambda z.\ id\ z))$

$\rightarrow_\beta id\ (\lambda z.\ id\ z)$

$\rightarrow_\beta \lambda z.\ id\ z$

opposite not true

if CBV converges, then CBN converges


apply
id
apply
id
λz
apply
id
z

$(\lambda x.\ x\ x)\ (\lambda x.\ x\ x)$

$\rightarrow_\beta\ (\lambda x.\ x\ x)\ (\lambda x.\ x\ x)$

$\rightarrow_\beta\ \ldots$

$(\lambda x.\ x\ x\ x)\ (\lambda x.\ x\ x\ x)$

$\rightarrow_\beta\ ((\lambda x.\ xxx)\ (\lambda x.\ xxx))\ (\lambda x.\ xxx)$

$\rightarrow_\beta\ (\lambda x.\ xxx)\ (\lambda x.\ xxx)\ (\lambda x.\ xxx)\ (\lambda x.\ xxx)$

$W = (\lambda x . x\ x\ x)$

$F = \lambda x . (\lambda y . y)$

$I = \lambda x . x$

$F\ (W\ W)\ I$

$\rightarrow F\ (W\ W\ W)\ I$

$\rightarrow F\ (W\ W\ W\ W)\ I$

$\rightarrow \ldots$

} doesn't terminate

Better Strategy

$(F\ (W\ W))\ I$

$\rightarrow (\lambda y . y)\ I$

$\rightarrow \lambda x . x$ done

} terminates

$\subset \beta_N$

β-reduction name clashes

$$(\lambda x.\ f\ x\ ((\lambda x.\ y\ x)\ v))\ z$$

$$(\lambda x.\ t)\ t' \longrightarrow [x \mapsto t']\ t$$

α-renaming

$$(\lambda x.\ f\ x\ ((\lambda l.\ y\ l))\ v)\ z$$

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

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

$\longrightarrow w$

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

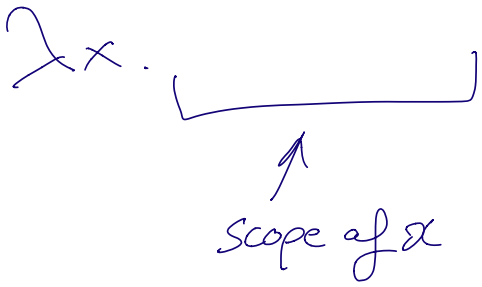
$(\lambda y.y)\ z$

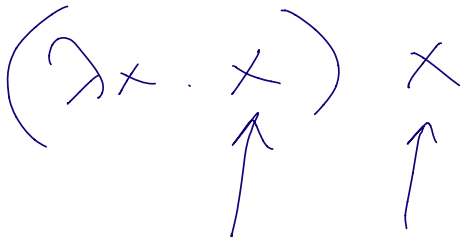
y got captured!

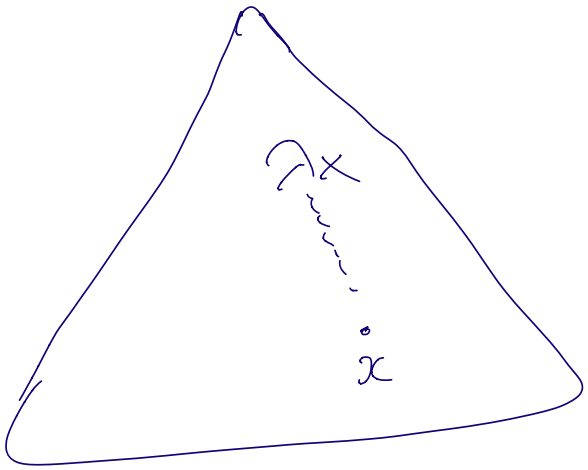
scoping

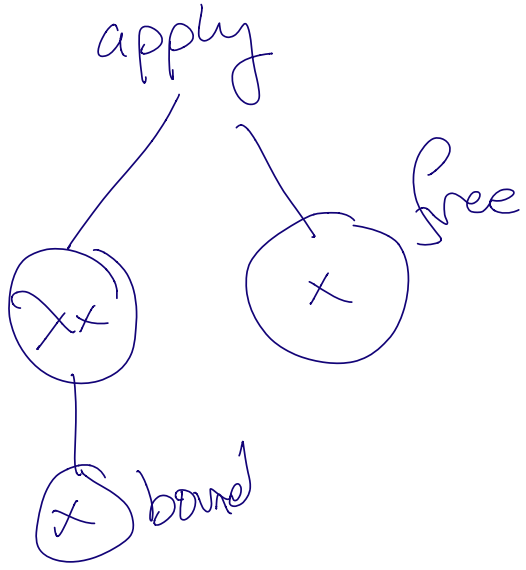
Bound variables

Free variables









Structural definition

① In x (ie a variable)

x is free

there are no bound variables

② $\lambda x . M$ ← arbitrary term

- Every x in M is bound
- for all $y \neq x$, if y is free in M, it is free in $\lambda x . M$

③ $M\ N$, similar to case $M\ N$

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

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

$$FV(M\ N) = FV(M) \cup FV(N)$$

$$t = x \mid \lambda x.t \mid t\ t$$