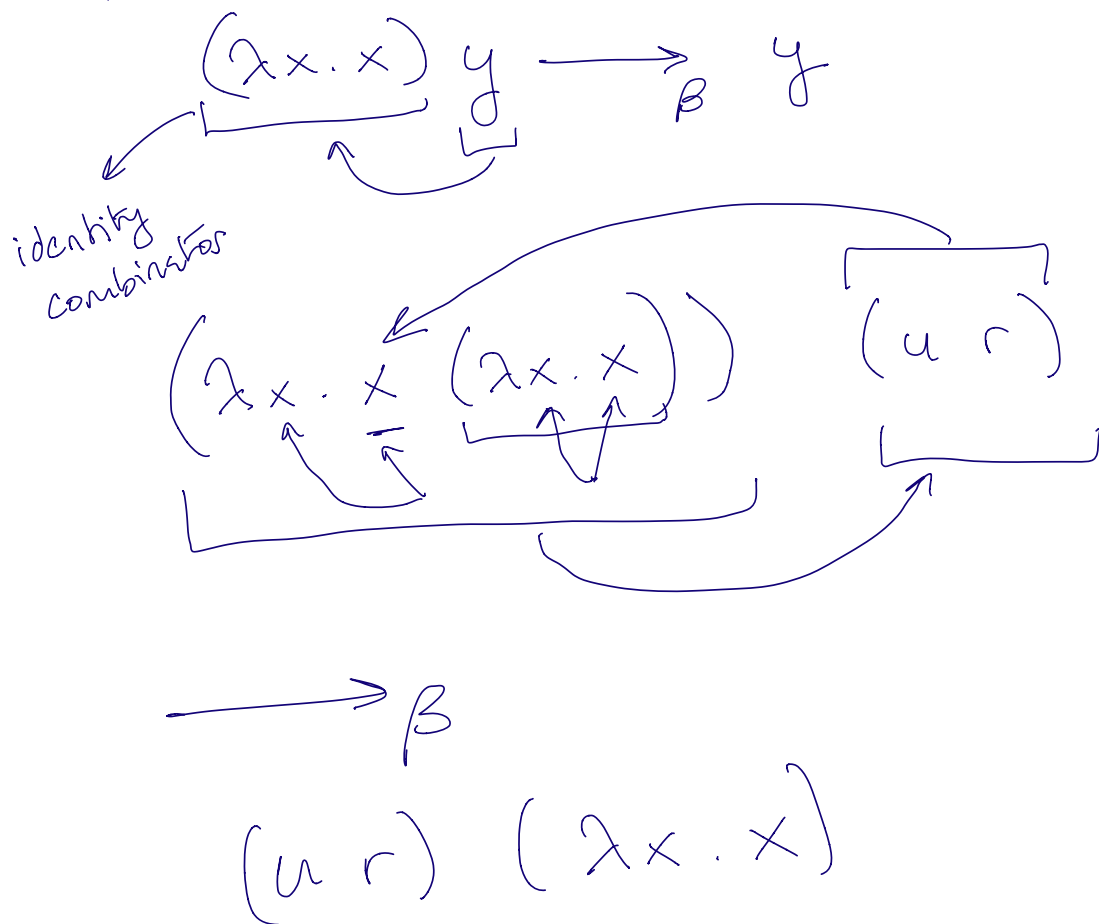
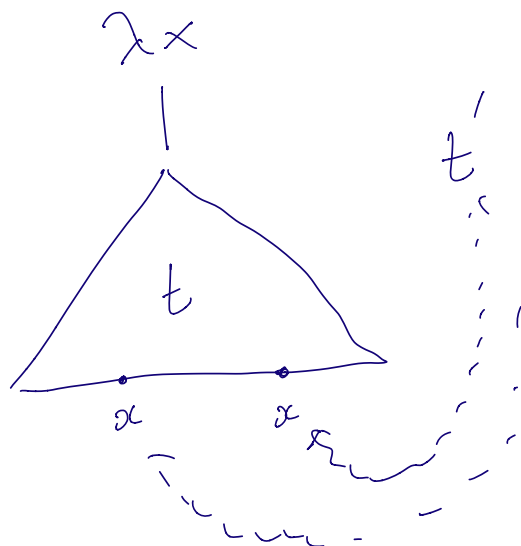


Today λ calculus evaluation

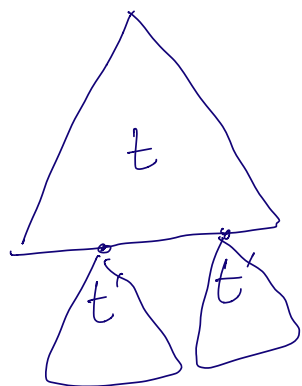
β -reduction



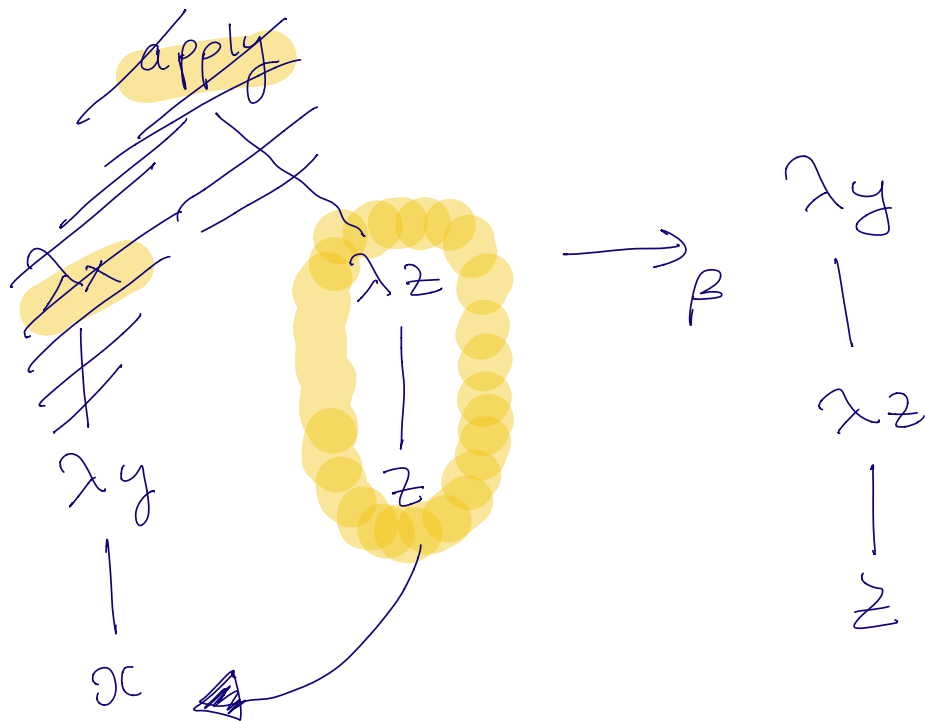
$$\underbrace{(\lambda x . t)}_{\beta} t'$$



\rightarrow_{β}



$$\begin{array}{c}
 (\lambda x. (\lambda y. x)) (\lambda z. z) \\
 \xrightarrow{\beta} \lambda y. (\lambda z. z)
 \end{array}$$



Reduction strategies

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

① Full beta reduction

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

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

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

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

normal form

② Normal order reduction

NOR left-most, outer-most redex first

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

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

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

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

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

③ Call by name (CBN)

no reductions inside abstractions

\swarrow
 \downarrow
 \downarrow
 $\text{id} (\text{id} (\lambda z. \text{id } z))$

\rightarrow
 $\text{id} (\lambda z. \text{id } z)$

\rightarrow
 $\lambda z. \text{id } z$

ALGOL 60

Haskell

call by need

lazy evaluation

④ Call by value

only outermost redexes reduced

redex reduced when its RHS has been reduced

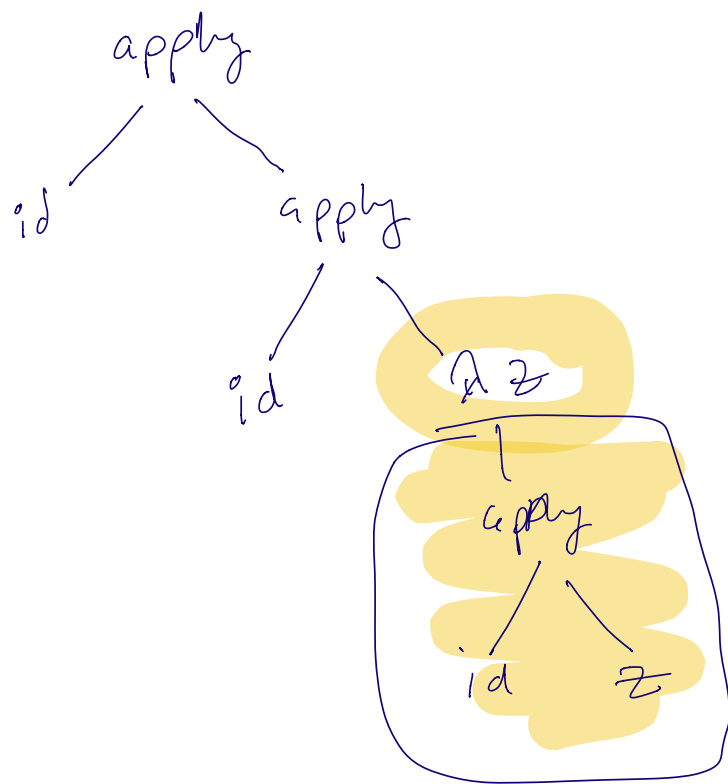
$\text{id} (\text{id} (\lambda z. \text{id } z))$

\rightarrow_{β}
 $\text{id} (\lambda z. \text{id } z)$

\rightarrow_{β}
 $\lambda z. \text{id } z$

opposite
not true

if CBV converges,
 then CBN converges



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

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

$$\rightarrow_{\beta} \dots$$

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

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

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

$$W = \underline{(\lambda x. x x x)}$$

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

$$I = \lambda x. x$$

$$F (\underline{W W}) I$$

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

$$\rightarrow \underline{F} (\underline{W W W W}) I$$

$\rightarrow \dots$

doesn't
terminate

Better strategy

$$(\underline{F (W W)}) I$$

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

$$\rightarrow \lambda x. x \quad \underline{\text{done}}$$

terminates

CBN

β -reduction name clashes

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


$$(\lambda x. t) \ t' \rightarrow \underline{[x \mapsto t'] \ t}$$

α -renaming

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


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

$$\rightarrow (\lambda y. \omega) z$$

$$\rightarrow \omega$$

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

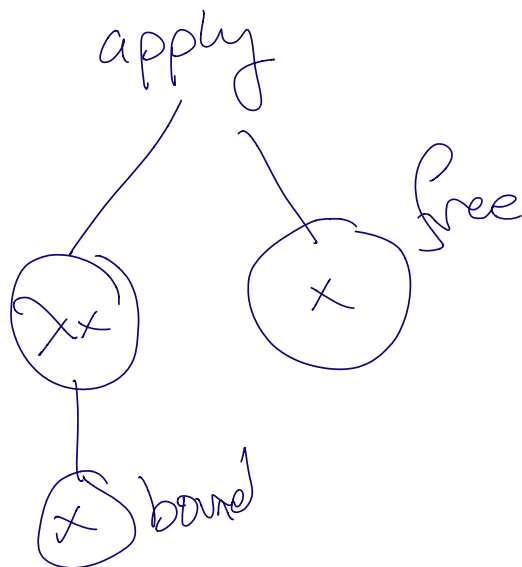
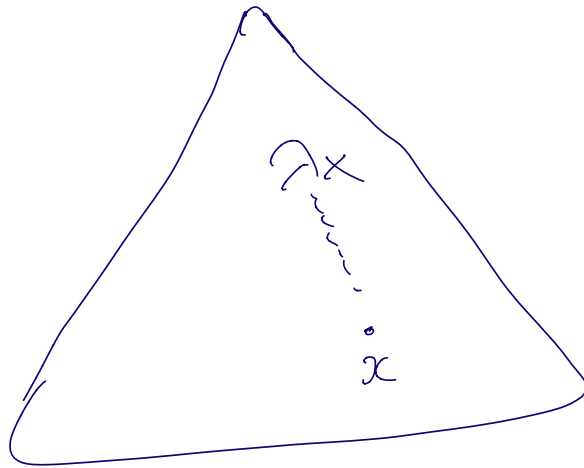
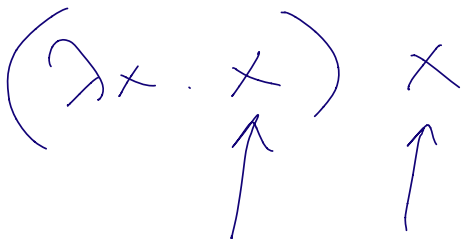
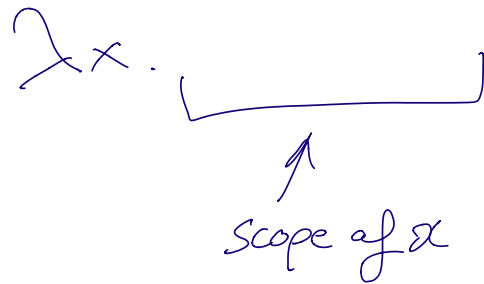
$$(\lambda y. y) z$$

y got
captured!

scoping

Bound variable

Free variables



Structural definition

① $\ln x$ (ie a variable)

x is free

there are no bound variables

② $\lambda x. M$ \swarrow 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 $M N$

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

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

$$FV(\underline{M N}) = FV(M) \cup FV(N)$$

$$t = x$$

$$| \lambda x. t$$

$$| t t$$