

Tutorial 1

1.

- a) x is function that takes x as a parameter and returns x .
- b) x is function that takes x as a parameter and returns x applied to x . $x(x)$
- c) x is a function that takes the function y that takes the parameter x and returns x as a parameter.
- d) x is the function that takes x and y and returns x .
- e) the function that takes x and y and returns x applied to y .
- e) is the function that takes x and returns x applied to x .

2.

- a) ~~$\lambda x. xy$~~
- b) $(\lambda y. y) (\lambda x. xy)$
- c) $(\lambda x. xy) (\lambda x. xy)$
- d) $\lambda x. (\lambda y. x) \rightarrow \lambda xy. x$
- e) $\lambda xyz. xy(yz)$

$$g) \forall x \lambda z. x \lambda (z)$$

$$h) \forall x. (\forall \lambda. x) \Rightarrow \forall x \lambda. x$$

$$i) (\forall x. x \lambda) (\forall x. x \lambda)$$

$$j) (\forall \lambda. \lambda) (\forall x. x \lambda)$$

$$k) \forall x. x \lambda$$

x obliges to x

g) if for function not possible x any other

x obliges to x

h) for function not possible x any other

any other x

i) x is not function not possible x any other

function not possible x is a function

j) x is a function not possible for function not possible

any other x obliges to x (x)

k) x is function not possible x is a function

g

any other x

g) x is function not possible for function not possible

$$(((h, x) : x \forall) ((z h)(x, A))) \beta$$

}

function

Foundations - substitution

To substitute, it must be applied to

$$V / (AB) / (\lambda v. A)$$

variable 1. $V [V := C] \equiv C$ (replace variable with C) $\equiv C$

variable 2. $V [v' := C] \equiv V$

application 3. $(AB) [v := C] \equiv A[v := C] B[v := C]$ (application)

same 4. $(\lambda v. A) [v := C] \equiv \lambda v. A$ (function)

5. $\lambda v. A [v' := C] \equiv$

$$FV(\lambda v. A) = FV(A) - \{v\}$$

$$v \neq v' \quad v \notin FV(C) \quad x \neq y$$

$$x \notin FV(\heartsuit) \text{ in}$$

$$(\lambda x. xy) [y := \heartsuit] \equiv$$

$$\boxed{\lambda x. (xy)} [y := \heartsuit] \equiv \lambda x. x [y := \heartsuit] y [y := \heartsuit] \equiv \lambda x. x \heartsuit$$

key

\equiv syntactically equal

\equiv computational equal

FV : free variable

$/$ is subtraction $(-)$

the replacement must be free

• Negation of terms:

$$\begin{aligned} & \neg \forall v' \neg \forall v \in FV(A) \\ & \neg \forall v' \neg \forall v \in FV(A) \end{aligned}$$

$$\lambda v''. A[v := v''] [v' := c]$$

$v'' = \text{new variable}$

$$(\lambda x. xy) [y := x]$$

$$\lambda x_{1000} (xy) [x = x_{1000}] [y := x] \equiv z$$

$$\lambda x_{1000} \cdot (\lambda [x := x_{1000}] y [x := x_{1000}]) [y := x] \equiv z$$

$$\lambda x_{1000} (x_{1000} y) [y := x] \equiv z$$

$$\lambda x_{1000} \cdot x_{1000} [y := x] y [y := x] \equiv$$

$$\lambda x_{1000} \cdot x_{1000} x$$

Rule 3

Substitute in the left and right parts

Higher order function: takes anything including a function as a parameter.

Tutorial 4, 5, 6

14

$$1. (\lambda y. (\lambda y. yx) (\lambda x. x))$$

$$2. (y (\lambda z. xz)) [x := (\lambda y. zy)]$$

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

$$1. (\lambda y. x (\lambda x. x)) [x := \lambda y. yx] \equiv$$

$$R5 \quad \lambda y. (x (\lambda x. x)) [x := \lambda y. yx] \equiv$$

$$R3 \quad \lambda y. [x := \lambda y. yx] (\lambda x. x) [x := \lambda y. yx] \equiv$$

$$R1 \quad \lambda y. (\lambda y. yx) (\lambda x. x) [x := \lambda y. yx] \equiv$$

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

$$R3 \quad 2. (y) [x := (\lambda y. zy)] (\lambda z. xz) [x := (\lambda y. zy)] \equiv$$

$$R1 \quad (y (\lambda z. (\lambda y. zy) z))$$

$$R6 \quad (y (\lambda z'. (\lambda y. zy) z'))$$

$$B7 \quad (\lambda (y z, (\lambda (y z) z)))$$

$$B8 \quad (\lambda (y z, (\lambda (y z) z)))$$

$$B9 \quad (\lambda (y) [x := (\lambda (y z) z)] (\lambda (y z) [z := (\lambda (y z) z)]))$$

$$y \cdot (\lambda (y z) (\lambda (y z) z))$$

$$B1 \quad y \cdot (\lambda (y z) (\lambda (y z) z)) [x := \lambda (y z) z] \equiv$$

$$B3 \quad y \cdot (\lambda (y z) [x := \lambda (y z) z]) (\lambda (y z) [x := \lambda (y z) z]) \equiv$$

$$B2 \quad y \cdot (\lambda (y z) (\lambda (y z) z)) [x := \lambda (y z) z] \equiv$$

$$y \cdot (\lambda (y z) (\lambda (y z) z)) [x := \lambda (y z) z] \equiv$$

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

$$y (\lambda (y z) [x := (\lambda (y z) z)]) \equiv$$

$$R2 \quad y (\lambda (y z) [x := (\lambda (y z) z)]) \equiv$$

$$2. \quad y (\lambda (y z) [x := (\lambda (y z) z)]) \equiv$$

Foundations ~

Alpha Rule

$$\lambda x. x \rightarrow_{\alpha} \lambda y. y$$

$$(\lambda x. x)z \rightarrow_{\alpha} (\lambda y. y)z$$

$$\lambda x. xy \rightarrow_{\alpha} (\lambda z. zy)$$

$$(\lambda y. \lambda x. xy) \rightarrow_{\alpha} (\lambda y. \lambda z. zy)$$

$$\lambda x. x \xrightarrow[0]{\text{number of steps (no steps)}}_{\text{reduces}} \alpha \lambda x. x$$

$$\lambda x. \lambda y. xy \rightarrow_{\alpha} \lambda z. \lambda x'. zx'$$

$$\lambda x. \lambda y. xy \xrightarrow{2} \lambda z. \lambda x'. zx'$$

Beta Rule

$$(\lambda v. A)B \rightarrow_{\beta} A[v := B]$$

$$(\lambda x. xx)(\lambda x. x) \rightarrow_{\beta} (xx)[x := \lambda x. x] =$$

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

$$x[x := \lambda x. x] = \lambda x. x$$

Compatibility

$$A \xrightarrow{p} B$$

$$AC \xrightarrow{p} BC$$

$$A \xrightarrow{\alpha} B$$

$$AC \xrightarrow{\alpha} BC$$

$$A \xrightarrow{p} B \text{ then } [C[x] := A] \rightsquigarrow [C[x] := B]$$

Extensionality

$$\boxed{\begin{array}{l} \forall x : f(x) = g(x) \\ f = g \end{array}}$$

They must be the same program!

$$\boxed{\text{Eta: } \eta}$$

- a freekey η .

β

η

Symmetry

$$A \xrightarrow{p} B \text{ then } B \rightarrow A$$

Generation

$$A \xrightarrow{p} B \text{ then } A =_{\beta} B$$

Reflexivity

$$A =_{\beta} A$$

$$A =_{\beta} B$$

$$B =_{\gamma} C$$

$$A =_{\gamma} C$$

$$\frac{A \rightarrow_{\beta} B}{A \rightarrow_{\beta \eta} B}$$

$$A \rightarrow_{\beta \eta} B$$

η allows the elimination of λ abstractions

5.

$$(((\lambda yz. xyz) (\lambda x. xx)) (\lambda x. x)) x$$

~~(((\lambda yz. xyz) (\lambda x. xx)) (\lambda x. x)) x~~

~~x (x x)~~

~~$$(((\lambda yz. (\lambda x. xx)) (\lambda x. x)) x)$$~~

~~$$(((\lambda yz. (\lambda x. xx)) yz) (\lambda x. x)) x$$~~

~~$$(\lambda yz. (\lambda x. (\lambda x. x) (\lambda x. x))) x$$~~

$$((\lambda yz. (\lambda x. xx) yz) (\lambda x. x)) x$$

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

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

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

$$\Omega (\lambda x. x x) (\lambda x. x x) \rightarrow \beta$$

$$[\lambda x. x] y \rightarrow_{\beta} y \text{ has } \beta\text{-normal form}$$

$$\lambda x. x \quad \beta\text{-normal form}$$

$$\lambda x. y x \rightarrow_{\eta} y$$

$$\lambda x x \rightarrow_{\alpha} \lambda y. y$$

$$(\lambda v. A) B \rightarrow_{\beta} A[v := B]$$

$(\lambda v. A) B$ apply B in place of v

$$\lambda v. A v =_{\beta \eta} B v$$

Compatibility

$$A \rightarrow_{\beta \eta} B \text{ and } n \text{ (if uses } B \text{ and } n \text{ there.)}$$

↓

$$A \rightarrow_{\beta} A_1 \rightarrow_{\eta} A_2 \rightarrow_{\beta} A_3 \rightarrow_{\beta} A_4 \rightarrow_{\eta} A_5$$

$$(xx) [x := \lambda x. xx] \equiv$$

$$(\lambda x. xx) (\lambda x. xx) \rightarrow \beta$$

$$(xx) [x := \lambda x. xx] \equiv$$

$$(\lambda x. xx) (\lambda x. xx) \rightarrow \beta$$

$$\lambda x. xx$$

$$\lambda x. x$$

$$\lambda x. xx$$

$$\lambda x. (\lambda y. z) x \rightarrow \beta \lambda x. z [y := x]$$

$$\lambda x. z$$

$$ABC \equiv (AB)C$$

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

$$(\lambda x. z [z := \lambda x. xx])$$

Strongly terminating = Weakly terminating
Weakly terminating \neq strongly terminating

Weakly normalising can be called β normalising.

performance of better and no significant change

performance of better = significant change
performance of better = significant change

$$[x \cdot x \cdot x] \cdot x = x \cdot x \cdot x$$

$$(x \cdot x \cdot x) (x \cdot x \cdot x) (x \cdot x \cdot x)$$

$$ABC \equiv (AB)C$$

$$x \cdot x \cdot x$$

$$[x \cdot x \cdot x] \cdot x \cdot x \cdot x \rightarrow x \cdot x \cdot x \cdot x \cdot x \cdot x$$

$$x \cdot x \cdot x$$

$$x \cdot x \cdot x$$

$$x \cdot x \cdot x$$

$$(x \cdot x \cdot x) (x \cdot x \cdot x) \rightarrow x \cdot x \cdot x$$

$$[x \cdot x \cdot x] \cdot x \cdot x \cdot x \equiv x \cdot x \cdot x$$

$$(x \cdot x \cdot x) (x \cdot x \cdot x) \rightarrow x \cdot x \cdot x$$

NOTE: Eta conversion removes all bound variables

$$6. (\lambda x y z. x z (y z)) (\lambda x. x) (\lambda x. x)$$

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

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

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

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

$$7. \lambda z x. (\lambda y. y) x \rightarrow_{\eta} \lambda z y. y$$

$\lambda z y. y$ because we take all ~~bound~~ ^{variable} lambda abstractions ($\lambda z, \lambda y$) and combine and remove all bound variables (x, y).

8.

8

$$(x \cdot y)$$

the add an added value
has (x, y) and
the same of the same
value

$$x \cdot y \rightarrow x \cdot (y \cdot x)$$

$$x \cdot y \rightarrow x \cdot (x \cdot y)$$

$$(x \cdot y) \rightarrow (x \cdot x) \cdot y$$

$$x \cdot x \cdot (x \cdot x)$$

$$(x \cdot x \cdot (x \cdot x)) \cdot (x \cdot x)$$

$$1. (x \cdot x \cdot x) \cdot (x \cdot x) \rightarrow B$$

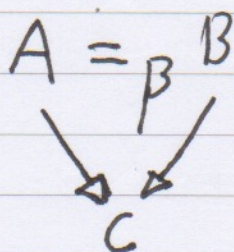
9.

Foundations notes

To be strongly β normalising, it must be also weakly β normalising.
 $A \rightarrow \rightarrow B$ does not mean $B \rightarrow \rightarrow A$

Example $(\lambda x. x) y \rightarrow y$ but $y \not\rightarrow (\lambda x. x) y$

$A =_{\beta} B$ means both programs terminate and give the same value.



Both programs A and B meet at some point in $A =_{\beta} B$ in a common C.

Ω is the looping program that never terminates.

Q Give a standard path that is not left most:

$$\lambda y. (((\lambda x. xx) (\lambda x. x) xy) y) \rightarrow_{\beta}$$

Sample questions

3. b) i)

$$A =_{\beta} B$$

B does not have to be in nf when A is

Proof $B \equiv (\lambda x. x) y$ $A \equiv y$

ii) Definition of has nf iff

$B =_{\beta} A$ and A in nf

iii) $\Omega \equiv (\lambda x. x x) (\lambda x. x x)$ has no beta nf since it is not weakly normalising. This is because even using the left most reduction so the program loops.

iv) $A \equiv (\lambda x. \neg(x x)) (\lambda y. \neg(y y))$

v) No, it doesn't normalise

vi) No, it is not weakly normalising.

3. a) i) $\lambda y z. (\lambda y. (\lambda x. x y z) z x) y$

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

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

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

ii)

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

iii) $\lambda y z. (\lambda a. (\lambda x. x p q) b c) y$

iv)

b) i) If $B \equiv (\lambda x. x) y$ and $A \equiv y$
then

$A =_{\beta} B$ therefore B does not have to be in nf

ii) If A has an nf then B must have/be in nf.

c) Ω is a program that never terminates.!!!

$$(\lambda x.xx)(\lambda x.xx) \rightarrow \beta$$

$$(\lambda x.xx)(\lambda x.xx) \rightarrow \beta$$

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

d) i) $\lambda z.xy \rightarrow \beta xy \rightarrow \eta \lambda z.xy \rightarrow \eta xy$

iii) $(\lambda z.zx)y \equiv \lambda z.z[x:=y] \rightarrow \beta yx$

iv) $(\lambda z.zx)y \rightarrow \eta (xy)$

v) $(\lambda x.xx)(\lambda x.xx) \rightarrow \beta (\lambda x.xx)(\lambda x.xx)$

So this program never terminates.

vi) $\lambda x.xy \neq (\lambda x.x)y \rightarrow \beta y$

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

3. a) i) $(\lambda y z. (\lambda y. ((\lambda x. x y z) z) x) y)$

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

iii) $\lambda y z. (\lambda y. (\lambda x. x y z) z x) y \rightarrow_{\alpha}$

$$\lambda y z. (\lambda p. (\lambda x. x j k) q r) y$$

iv) $\lambda y z. (\lambda y. (\lambda x. x y z) z x) y \rightarrow_{\beta}$

$$\lambda y z. (\lambda y. (\lambda x. x y z) [x := z] x) y \rightarrow_{\beta}$$

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

$$\lambda y z. ((z y z) x) y \rightarrow_{\beta} \quad \lambda y z. ((z y z) x) y$$

No

b) if false

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

$$B = y$$

$$A \rightarrow_{\beta} B \quad \text{but they are not the same!}$$

ii) The definition of β nf. has an af is that is in

c) Ω never terminates $(\lambda x. x x) (\lambda x. x x) \rightarrow_{\beta}$

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

and so on

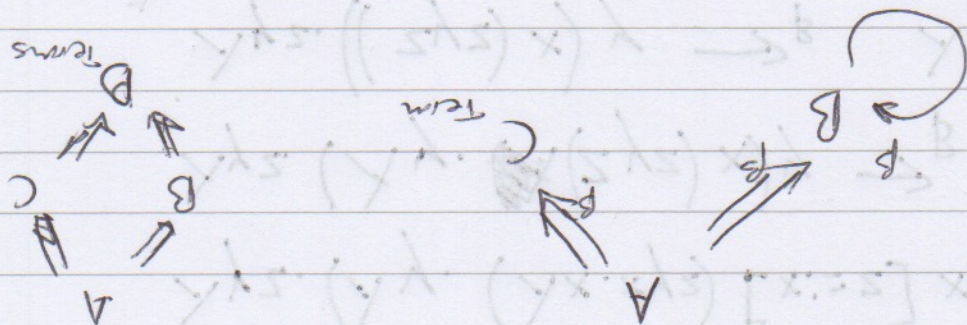
4. a) i) This is a lambda abstraction where x is applied to x and then negated.

ii) 4

iii) There are no free vars.

$$iv) A \equiv_a (\lambda x. \neg (xx)) (\lambda y. \neg (yy))$$

Heavily Normalising



$$(\lambda x. \neg (xx)) (\lambda y. \neg (yy))$$

$$\neg ((\lambda y. \neg (yy)) (\lambda y. \neg (yy)))$$

$$Y g \Rightarrow_g g (Y g)$$

$$Y = \lambda f. (\lambda x. f (xx)) (\lambda x. f (xx))$$

$$\Rightarrow_g g ((\lambda x. g (xx)) (\lambda x. g (xx))) \equiv g (Y g)$$

$$\left(\lambda a. \left(\lambda c. \left(\lambda j. c \ j \right) \right) \left(\lambda b. a \ c \ b \right) \right)$$

$$\lambda x y. \left(\lambda x y. y \right) \left(\left(\lambda x. x \ x \right) \left(\lambda x. x \ x \right) \right) P$$

Weakly normalising. Use normal order reduction to find weakly normalising terms.

$$\left(\left(\lambda x s. y \right) \left(\left(\lambda p t. p \right) \left(s \right) \right) \right) j$$

Normal order

Applicative order

$$(\lambda y. y) j$$

$$((\lambda x y. y) (L) j)$$

$$j$$

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

$$(\lambda x. x) y \quad k$$

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

$$y y$$

int mult (add1(3), ^{loop forever!} add2(x))
 return a x 3

$$1. A \equiv (\lambda x. x) y \rightarrow_{\beta} y.$$

$$\text{not } \text{true} =_{\beta} \text{false}$$

$$\begin{aligned} \text{not } \text{true} &\equiv (\lambda x. x \text{ false true}) \text{true} \rightarrow_{\beta} \\ & (x \text{ false true}) [x := \text{true}] \equiv \\ & \text{true false true} \end{aligned}$$

$$(\lambda x y. x) \text{false true} \rightarrow_{\beta}$$

$$(\lambda y. x) [x := \text{false}] \text{true} \equiv$$

$$(\lambda y. \text{false}) \text{true} \rightarrow_{\beta}$$

$$\text{false} [y := \text{true}] \equiv \underline{\text{false}}$$

$$\text{not} \equiv \lambda x. x \text{ false true}$$

$$\text{or } \text{true true} =_{\beta} \text{true}??$$

$$\text{or } \text{true true} \equiv$$

$$(\lambda x y. \text{cond } x \text{ true } y) \text{true true} \rightarrow_{\beta}$$

$(\lambda x. \text{cond } x \text{ true } y) \text{ true } \rightarrow y$

$\text{cond } \text{true} \text{ true } \equiv \text{true}$

$\text{cond } \text{true} \text{ false } \equiv \text{false}$

$\text{cond } \text{false} \text{ true } \equiv \text{true}$

$\text{cond } \text{false} \text{ false } \equiv \text{false}$

$(\lambda x. \text{cond } x \text{ true } y) \text{ true} \rightarrow y$

$(\lambda x. \text{cond } x \text{ true } y) \text{ false} \rightarrow \text{true}$

$(\lambda x. \text{cond } x \text{ true } y) \text{ true} \rightarrow y$

$\text{cond } \text{true} \text{ true } \equiv \text{true}$

$\text{cond } \text{true} \text{ false } \equiv \text{false}$

$\text{cond } \text{true} (\text{fst } (\text{pair } A \ B))$

$\text{cond } \text{true } \text{true} \text{ true } \equiv \text{true}$

$(\lambda y. \text{cond } \text{true } \text{true } y) \text{ true} \rightarrow y$

Exercise 3

1. not is defined as $\lambda x. x \text{ false true} \rightarrow_{\beta}$

$$(x \text{ false true}) [x := \text{false}] \rightarrow_{\beta} \equiv$$

$$(\text{false false true}) \equiv$$

$$((\lambda x y. y) \text{ false}) \text{ true} \rightarrow_{\beta}$$

$$(\lambda y. y) \text{ true} \rightarrow_{\beta} \text{true}$$

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

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

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

Ex 1

1. $A = (\lambda x. x) y$ $A' = (\lambda 1) 2$

$$(\lambda z (\lambda g. g (\lambda x. yx (\lambda z. zj))))$$

2. $A \equiv (\lambda x y. yx) (\lambda x. x)$ $(\lambda 1) (\lambda 2)$
 $(\lambda x. (\lambda y. yx))$ $(\lambda \lambda 1 2) (\lambda 1)$
 $\quad \quad \quad 2 \quad 1$

$\{x_1, s, z\}$

$\lambda y. y x \quad \lambda t. t y$

~~members of~~

$$(\lambda x y. y x) (\lambda x. x) (\lambda \lambda 12) (\lambda 2) (\lambda 2)$$

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

$$3. (\lambda x y. x y) (\lambda z. z x) (\lambda \lambda 21) (\lambda 1)$$

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

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

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

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

$$2. ((\lambda x y z. x y z) \text{ true}) A B$$

$$\lambda (\lambda y z. x y z) [x : \text{true}] \rightarrow \text{true} \equiv$$

$$(\lambda y z. \text{true } y z) \equiv$$

$$(\lambda z. \text{true } y z) [y := A] \rightarrow \text{true}$$

$$\lambda z. \text{true } A B$$

$$((\lambda x y. x) A) B$$

~~4. $\lambda x y. \text{cond}$~~

$$3(((\lambda x y z. x y z) \text{false}) A) B \equiv$$

$$((\lambda y z. x y z) [x := \text{false}] A) B \rightarrow_{\beta}$$

$$((\lambda y z. \text{false } y \ z) A) B \equiv ((\lambda z. \text{false } y \ z) [y := A]) B \rightarrow_{\beta}$$

$$((\lambda z. \text{false } A) B) \equiv (\text{false } A) [z := B] \rightarrow_{\beta}$$

$$\text{false } A \ B \equiv (\lambda x y. y) A B \rightarrow_{\beta} B$$

4. and true false

$$\text{and} \equiv \lambda x y. \text{cond } x \ y \ \text{false}$$

$$\lambda x y. (\text{cond } x \ y) \ \text{false} \ \text{true}$$

~~$$\lambda x y. (((\lambda x y z. x y z) x) y) (\text{true} \ \text{false}) \rightarrow_{\beta}$$~~

~~$$\lambda x y. (x \ (\text{true} \ \text{false})) \rightarrow_{\beta} \lambda x y. ((x) y) \ \text{true} \ \text{false} \rightarrow_{\beta}$$~~

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

$$\& (\text{cond } \text{false} \ \text{true}) \equiv ((\lambda x y z. x y z) \text{false}) \ \text{true}$$

λ

4. (1) $x \cdot y$ and $\neg(x \cdot y)$

~~Step~~ cond true false false $\equiv (\lambda x y z. x y z)$ true false

$$\text{false} \wedge \text{false} = (\text{A} \wedge \text{B}) \wedge \text{false} = (\text{A} \cdot \text{false}) \cdot \text{false}$$

S. and True

$$\neg x y \cdot ((\text{and } x \text{ y false}) \text{ true}) = \text{true}$$

$$29 \times 10^3 (26 \times 24 \times 9)$$

End true true false $\rightarrow B$

$$\begin{array}{l} \text{true} \\ \text{true} \\ \text{false} \end{array} \quad \begin{array}{l} ((\neg x) \vee y) \\ ((\neg x) \vee y) \\ ((\neg x) \vee y) \end{array} \quad \begin{array}{l} \text{true} \\ \text{true} \\ \text{false} \end{array} \rightarrow \begin{array}{l} \text{true} \\ \text{true} \\ \text{false} \end{array}$$

true

6. and false false

$$((\lambda xy. \text{cond } xy \text{ false}) \text{ false})$$

cond. false false false $\rightarrow p$ false false false

$$((\forall x. y) \text{ false}) \text{ false} \rightarrow \text{false} \quad (\forall y. \text{ false}) \text{ false} \rightarrow \text{false}$$

$$\text{or} \equiv \lambda xy. \text{cond } x \text{ true } y$$

$$7. \text{ and false true } \equiv_{\beta} \text{ false}$$

$$(\lambda xy. \text{cond } xy \text{ false}) \text{ false true } \rightarrow_{\beta}$$

$$(\text{cond false true false}) \rightarrow_{\beta} \text{ false true false}$$

$$(((\lambda xy. y) \text{ true}) \text{ false}) \rightarrow_{\beta}^2 \text{ false}$$

$$8. \text{ or true false } \equiv_{\beta} \text{ true}$$

$$((\lambda xy. \text{cond } x \text{ true } y) \text{ true}) \text{ false } \rightarrow_{\beta}^2$$

$$\text{cond true true false} \equiv (((\lambda xyz. xyz) \text{ true}) \text{ true}) \text{ false}$$

$$\text{true true false} \equiv (\lambda xy. x) \text{ true false} \rightarrow_{\beta} (\lambda y. \text{true}) \text{ false} \rightarrow_{\beta} \text{true}$$

$$9. \text{ or true true } \equiv_{\beta} \text{ true}$$

$$((\lambda xy. \text{cond } x \text{ true } y) \text{ true}) \text{ true } \rightarrow_{\beta}$$

$$\text{cond true true true} \rightarrow_{\beta} \text{true true true} \equiv$$

$$((\lambda xy. x) \text{ true}) \text{ true} \rightarrow_{\beta}^2 \text{true}$$

10. or false false = false

$$((\lambda xy. \text{cond } x \text{ true } y) \text{ false}) \text{ false} \Rightarrow \text{false}$$

$$\text{cond false true false} \Rightarrow \text{false}$$

$$((\lambda xy. y) \text{ true}) \text{ false} \Rightarrow \text{false}$$

11. or false true = true

$$((\lambda xy. (\text{cond } x \text{ true } y) \text{ false}) \text{ true}) \Rightarrow \text{true}$$

$$\text{cond false true true} \Rightarrow \text{true}$$

$$((\lambda xy. y) \text{ true}) \text{ true} \Rightarrow \text{true}$$

Ex 1

$$1. \lambda x. x \ y$$

$$2. (\lambda xy. xy) \ y$$

$$3. (\lambda x \lambda y. x) \ y$$

$$(\lambda xy. xy) \cdot (\lambda z. z) : (\lambda 1 \ 2) \ (\lambda 1 \ 2)$$

$$(\lambda x \lambda y. xy) (\lambda 1 \ 2) (\lambda 1 \ 2) \Rightarrow (\lambda y. y) (\lambda 1 \ 2) \Rightarrow (\lambda 1 \ 2)$$

$$(\lambda x y. xy) y \equiv (\lambda y. xy) [x:=y] \rightarrow_{\beta} \lambda y. yy$$

$$\begin{array}{ccc} (\lambda x y. xy) & (\lambda z. zz) & \rightarrow_{\alpha} \lambda z. zz \\ \substack{2 \quad 1} & \substack{1 \quad 3} & \\ (\lambda \lambda 21) & (\lambda 13) & \end{array} \neq \lambda y. xy$$

Exercise 4

$$\text{posin} \equiv \lambda x. \text{fst} (\text{snd} (\dots (\text{snd } x) \dots)) \text{ for } i < n$$

Exercise 5

$$1. \text{succ } \lambda zyx. zy(yx)$$

$$\lambda zyx. (zy(yx))^n \equiv_{\beta} \lambda zyx. (zy(yx)) [z:=n] \rightarrow_{\beta}$$

$$xyx. (ny(yx)) \rightarrow_{\beta}$$

$$\lambda yx. ((\lambda x. y^n x) y (yx)) \rightarrow_{\beta}$$

$$\lambda yx. ((\lambda x. y^n x) (yx)) \rightarrow_{\beta} \lambda yx. y^n xy$$

$$\text{succ } n+1 \equiv \lambda yx. y^{n+1} x$$

$$2. \text{iszero } 0 \equiv_{\beta} \text{true}$$

$$(\lambda z. z (\lambda x. \text{false}) \text{true}) 0$$

$$(\lambda z. z (\text{false}) [x:\text{true}]) 0 \rightarrow_{\beta} \text{false} (\lambda z. z (\text{false}) 0)$$

$$\lambda z. z (false) 0 \equiv \lambda z. z (\lambda xy. y) 0$$

not false = true

$$\lambda x. x \begin{cases} true \\ false \end{cases} \text{ false = true}$$

$$(\lambda x. x \lambda xy. y \lambda xy. x) \lambda xy. y = \text{true}$$

$$\lambda xy. y \text{ false} = \text{true}$$

Things that cannot be defined in λ calculus

$$\text{null } A =_p \begin{cases} \text{true} & \text{if } A =_p \perp \\ \text{false} & \text{otherwise} \end{cases}$$

Notes

$$\lambda x. xx \equiv \lambda y. yy \quad \lambda z. yz \equiv \lambda x. zx$$

$$\lambda x. xx \not\equiv \lambda y. yz$$

$$\lambda x. xx =_{\alpha} \lambda y. yy$$

paper cut
1/11/2014

Normal order reduction
(Left most reduction)

$$\begin{aligned} ((\lambda a. a) (\lambda b. b) (\lambda c. c)) &\rightarrow_p \\ ((\lambda b. b) (\lambda c. c)) &\rightarrow_p \\ (\lambda c. c) \end{aligned}$$

Applicative order
reduction (right most
reduction)

$$\begin{aligned} ((\lambda a. a) (\lambda b. b) (\lambda c. c)) &\rightarrow_p \\ (\lambda a. a) (\lambda c. c) &\rightarrow_p \\ (\lambda c. c) \end{aligned}$$

$$((\lambda a. b) ((\lambda x. xx) (\lambda x. xx)))$$

Normal

$$(\lambda a. b) [a := ((\lambda x. xx) (\lambda x. xx))] \rightarrow_B$$

b

Applicative order

$$(\lambda a. b) ((\lambda x. xx) (\lambda x. xx)) \rightarrow_B (\lambda a. b) ((\lambda x. xx) (\lambda x. xx))$$

Eth

$$(\lambda x. xy \rightarrow y) \rightarrow_{\lambda y} (\lambda x. xy \rightarrow y)$$

$$\lambda x. xy \rightarrow y$$

y

$$\text{IFThenElse } (\lambda a. \lambda s. \lambda c. a \ s \ c) ((\lambda B) C) ((\lambda A) B) C$$

$$((\lambda A) B) C$$

$$((\lambda x. xy) (\lambda a. \lambda s. \lambda c. a \ s \ c)) T$$

$$y = \lambda a. \lambda s. a$$

$$\lambda x. xy \quad \lambda xy. x \quad \lambda x. \lambda y. x$$

$$\begin{array}{c|c} (\lambda z x. (\lambda y. y) x) \quad T \quad F & (\lambda z y. y) \quad T \quad F \\ \hline \Rightarrow_{\beta} (\lambda x. (\lambda y. y) x) \quad F & \Rightarrow_{\beta} (\lambda y. y) \quad F \\ \Rightarrow_{\beta} (\lambda y. y) \quad F & \Rightarrow_{\beta} \underline{F} \\ \Rightarrow_{\beta} \underline{F} & \end{array}$$

$$\begin{array}{c|c|c} (\lambda z x. (\lambda y. y) x z) \quad T \quad F & & \\ \hline \Rightarrow (\lambda y. y) \quad F \quad T & (\lambda x. (\lambda y. y) x) \quad T \quad F & (\lambda z. (\lambda y. y) z) \quad T \quad F \\ & \Rightarrow (\lambda y. y) \quad T \quad F & (\lambda y. y) \quad T \quad F \\ & \xleftrightarrow{\alpha \text{ convertible}} & \end{array}$$

$$(\lambda z x. (\lambda y. y) x z) \quad T \quad F \rightarrow_{\eta} (\lambda y. y) \quad T \quad F \quad ?$$

We cannot have something with a λ at the start.
 at start reduce to variable with the start.

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

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

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

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

$$(\lambda x. xy) y \rightarrow_{\beta} yy$$

$$f \equiv \lambda x. yz$$

$$(\lambda x f. qxf) f \rightarrow_{\beta} \lambda f. qff$$

$$M = \lambda x. xy$$

Eta conversion can only happen if $\lambda xy. My$ if x and y are ~~bound~~ in M .
not free

$$(\lambda p. \lambda q. (\lambda r. pqr)) \rightarrow_{\eta} (\lambda x. x) \quad \text{TK}$$

$$/ (x. y)$$

$$\lambda x. \lambda y. My$$

$$\lambda x. \lambda y. My$$

$$\lambda x. \lambda y. My$$

(y occurs free in M)

(y occurs bound in M)

(y does not occur in M)

$$\lambda y. y$$

$$(\lambda j. \lambda i. (\lambda p. \lambda q. qp) i (\lambda a. \lambda b. \lambda j. jab) j)$$

$$(\lambda i. (\lambda p. \lambda q. qp) i (\lambda a. \lambda b. \lambda j. jab))$$

Its reduction only works if the arguments are not redexes and if they are at the end of the expression and not ~~for~~ in any subterms. Each variable may only occur once.

$$\lambda x y. x y (\lambda y. y) \rightarrow_n \lambda x y. x y \rightarrow_n \text{nothing}$$

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

$$e) \lambda z. \left(\lambda y. \left((\lambda x. x) (\lambda x. x z) (\lambda y. y) (y) \right) \right) z$$

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

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

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

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

$$\lambda y. y \Rightarrow_\beta y (y)$$

$$\cancel{\lambda y. y} \quad \lambda y. (\lambda y. \text{is zero } y \text{ true } (a \text{ pred } y))$$

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

$$\Rightarrow y (y) z$$

$$c) \quad \lambda y. \underbrace{y(x \ x \ y)}_{\uparrow} \ (\lambda x y. y \ (x \ x \ y))$$

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

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

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

$$\top$$

Exercises

5.

$$1) \text{ succ } n =_{\beta} n + 1$$

$$(\lambda z y x. z y (y x)) \lambda y x. y^n x \equiv_{\beta} \cancel{(\lambda y x. y^n x)} \cancel{(\lambda y x. y x)}$$

$$\lambda y x. (\lambda y x. y^n x) y (y x) \equiv_{\beta} \lambda y x. y^n x (y x) \equiv_{\beta} \lambda y x. (y^n (y x))$$

$$2) \text{ iszero } 0 =_{\beta} \text{ true} \quad (\lambda z. z (\lambda x. \text{false}) \text{true}) 0$$

$$(\lambda y x. x (\lambda x. \text{false}) \text{true}) \rightarrow_{\beta} (\lambda x. x) \text{true} \equiv_{\beta} \text{true}$$

$$3) \text{ iszero } (\text{succ } n) \equiv_{\beta} \text{false}$$

$$(\lambda z. z (\lambda x. \text{false}) \text{true})$$

$$(\lambda x. \lambda y. (y x) \text{ true})$$

3) isos (true) \equiv^b true

$$(\lambda x. \lambda y. (y x) \text{ true}) \rightarrow^b (\lambda x. x) \text{ true} \equiv^b \text{true}$$

$$\text{is } 1) \text{ isos } 0 \equiv^b \text{true} \quad (\lambda x. \lambda y. (y x) \text{ true}) 0$$

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

$$\lambda x. (\lambda y. (\lambda z. (y z) x)) \equiv^b (\lambda x. \lambda y. (\lambda z. (y z) x)) \equiv^b$$

$$(\lambda x. \lambda y. (\lambda z. (y z) x)) \cdot \lambda x. \lambda y. x \equiv^b (\lambda x. \lambda y. x) \cdot (\lambda x. \lambda y. x)$$

$$1) \text{ true } u \equiv^b u + 1$$

P.

Exercises

2010 Foundations Exam

$$4. \text{a)} \left((\lambda x y z. xz(yz)) (\lambda x. xx) (\lambda x. xx) x (\lambda x. xx) (\lambda x. xx) \right) \rightarrow_{\beta}$$

$$(\lambda y z. (\lambda x. xx) z(yz)) (\lambda x. xx) x (\lambda x. xx) (\lambda x. xx) \rightarrow_{\beta}$$

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

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

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

$$(xx (xx)) (\lambda x. xx) (\lambda x. xx) \rightarrow NF.$$

Weakly normalising because it cannot right reduce as the right is Ω

2013

A Ω

$$\overset{A}{(\lambda x. y)} (\overset{B}{(\lambda x. x)}) (\lambda x. xx) (\lambda x. xx) \text{ -weak}$$

$$3. \text{a)} (\lambda y z. (\lambda x. xx) yz) (\lambda x. x) x \rightarrow$$

$$\cancel{(\lambda z. (\lambda x. xx) (\lambda x. x) z)} x \rightarrow \lambda x. xx$$

$$(\lambda z. (\lambda x. xx) \cancel{\lambda x. x} z) x \rightarrow$$

$$\cancel{(\lambda z. (\lambda x. xx) (\lambda x. x) z)} x \rightarrow$$

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

3. a) $\lambda xy z. x y z$

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

b) Applies x to y then applies z to that

$$c) (\lambda x. xxx) ((\lambda x. xxx) ((\lambda x. xxx) (\lambda x. x))) \rightarrow \beta$$

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

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

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

$$(\lambda x. x)$$

$$2) (\lambda y z. (\lambda x. xx) y z) (\lambda x. x) x \rightarrow \beta$$

$$(\lambda z. (\lambda x. xx) (\lambda x. x) z) x \rightarrow \beta$$

$$(\lambda z. (\lambda x. x) (\lambda x. x) z) x \rightarrow \beta$$

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

$$\begin{aligned}
 &g) \quad (\lambda x. \lambda y. y (\lambda x. x x) x) P Q \\
 &\quad (\lambda x. \lambda y. y \overline{x x}) P Q \\
 &\quad (\lambda y. y P P) Q \\
 &\quad Q P P
 \end{aligned}$$

$$\begin{aligned}
 &(\lambda x. x (\lambda z. z) x) (\lambda y. y) \\
 &\quad (\lambda x. x) (\lambda y. y) \\
 &\quad (\lambda y. y)
 \end{aligned}$$

$$4) \quad ((\lambda x y z. x z (y z)) (\lambda x. x x) (\lambda x. x x) x) (\lambda x. x x) (\lambda x. x x)$$

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

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

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

$$\cdot \quad (\lambda x. x x) (\lambda x. x x)$$

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

$$\lambda A \rightarrow_B A (Y A) \quad Y \equiv (\lambda xy. y (xxy)) (\lambda xy. y (xxy))$$

$$(\lambda xy. y (xxy)) (\lambda xy. y (xxy)) A \rightarrow_B$$

$$(\lambda y. y ((\lambda xy. y (xxy)) (\lambda xy. y (xxy)))) A \rightarrow_B$$

$$A ((\lambda xy. y ((\lambda xy. y (xxy)) (\lambda xy. y (xxy))))$$

$$\begin{array}{c} X \\ X \\ \hline (\lambda x. x x) (\lambda x. x x) \\ \hline (\lambda x. x x) (\lambda x. x x) \\ X \\ Y (\lambda x. x x) \\ X \equiv Y (\lambda x. x x) \end{array} \quad \text{g)}$$

$$2. a) \lambda x y z. ((xy)z \ (\lambda x. x)) (\lambda x. xz \ (yz)) \ \lambda x. ((\lambda x) y)$$

$$\lambda x y z. xyz \ (\lambda x. x) (\lambda x. xz \ (yz)) (\lambda x. xxy)$$

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

$$c) \lambda y z. y = A \quad \lambda x. x x = B \\ \lambda x. x x = C$$

$$d) A = \lambda y. \text{cond (fst } y) \text{ I } \Omega \\ B = \text{pair true } \Omega$$

e) No, everything that is computable is computable in the Lambda calculus

Church's thesis states that anything that is computable is representable in λ calculus. Church's thesis has not yet been disproven.

f) • You need to define functions such as numbers even

$$3. a) i) ((x \cdot y \cdot x) A) B \equiv_B A$$

$$ii) A) \lambda x y. x$$

b) No. The λ calculus is the language of computability, so everything is computable. is representable in λ calculus

c) No. Everything in λ calculus is computable.

$$d) v \in FV(A) \quad A_v = B_v \quad B_v \quad A = B_n \quad B$$

$$\text{Because } v \in FV(A) \quad \text{Because } v \in FV(B)$$

$$\lambda v. A_v \rightarrow_n A \quad \lambda v. B_v \rightarrow_n B$$

$$A = \lambda x. v x$$

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

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

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

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

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

Does not normalize

2. a) $w(((\lambda x y z. x z (y z)) u) v)$

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

b) $w(\lambda x y z. x z (y z)) u v, w, x, y, z, u, v, (y z), x z (y z),$
 $x z, \lambda x y z. x z (y z), \lambda y z. x z (y z), \lambda z. x z (y z),$

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

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

$y = A A$

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

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

d)

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

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

