

EXERCISES

CHAPTER 1

SEAN LI ¹

1. Redacted

Problem

(1.1) Simplify notation of the following terms

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

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

Solution.

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

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

Problem

(1.2) Find the alpha equivalent terms to

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

In

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

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

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

Solution. Only (a).

Problem

(1.3) Prove

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

Solution.

Proof. By definition of alpha equivalence

$$M \underset{\alpha}{=} N \iff \exists \varphi, M^\varphi \underset{\alpha}{\rightarrow} N \wedge \text{FR } M = \text{FR } N$$

The witness of φ is substituting bound variable x with z , and z is not a free variable in the term, thus the two terms are alpha equivalent.

$$\lambda x . x (\lambda z . y) \xrightarrow[\alpha]{x \rightarrow z} \lambda z . z (\lambda z . y)$$

■

Problem

(1.4) Consider the following term:

$$U := (\lambda z . z x z)((\lambda y . x y) x)$$

1. Find Sub U
2. Draw tree rep of U
3. Find FV U
4. Find alpha equivalent terms to U from below and point out which of those follows the Barendregt convention:

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

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

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

(d) $(\lambda v . (v x) v)((\lambda u . u v) x)$

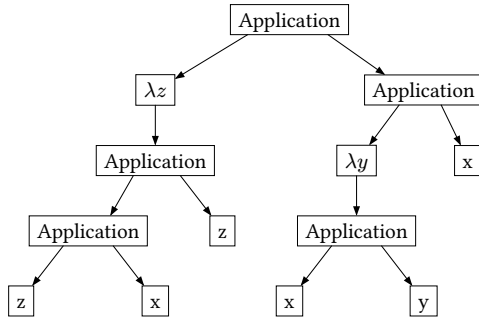
1. Find Sub U .

Solution.

$$\begin{aligned}
\text{Sub } U &= \\
&\{(\lambda z . z x z)((\lambda y . x y) x), (\lambda z . z x z), ((\lambda y . x y) x)\} \cup \\
&\{(\lambda y . x y), z x z, z x, x y\} \cup \\
&\{z, x, y\} \\
&= \{(\lambda z . z x z)((\lambda y . x y) x), (\lambda z . z x z), ((\lambda y . x y) x), \\
&\quad (\lambda y . x y), z x z, z x, x y, z, x, y\}
\end{aligned}$$

2. Draw a tree rep of U .

Solution.



3. Find $\text{FV } U$

Solution.

$$\begin{aligned}
\text{FV } U &= \text{FV } (\lambda y . y x y) \cup \text{FV } (\lambda z . x z) x \\
&= (\text{FV } y x y) \setminus \{y\} \cup (\text{FV } \lambda z . x z) \cup \{x\} \\
&= (\text{FV } y x) \setminus \{y\} \cup (\text{FV } x z) \setminus \{z\} \cup \{x\} \\
&= \{x\}
\end{aligned}$$

4. Find an alpha-equivalent term.

Solution.

$$(a) \stackrel{\alpha}{=} (c) \stackrel{\alpha}{=} U$$

Only (a) follows the Barendregt convention.

Problem

(1.5) Give the results of the following substitutions

- (a) $(\lambda x . y (\lambda y . x y))[y := \lambda z . z x]$
- (b) $((x y z)[x := y])[y := z]$
- (c) $((\lambda x . x y z)[x := y])[y := z]$
- (d) $(\lambda y . y y x)[x := y z]$

Solution.

- (a) $(\lambda v . (\lambda z . z x)(\lambda u . v u))$
- (b) $(y y z)[y := z] = z z z$
- (c) $(\lambda x . x y z)[y := z] = (\lambda x . x z z)$
- (d) $(\lambda u . u u (y z))$

Problem

(1.6)

$$\neg \left(\forall M L N \in \Lambda, M [x := N, y := L] \equiv_{\alpha} M [x := N][y := L] \right)$$

Solution.

Proof. Because $\text{RHS} = M [x := N][y := L] = M [x := N [y := L]][y := L]$, if $y \in \text{FV } N$, then what x gets substituted with will have y substituted for L , which is completely different with LHS. ■

Problem

(1.7) Find all available redexes in

$$U := (\lambda z . z x z)((\lambda y . x y) x)$$

And all reduction pathes to the β -normal form.

Solution. The first redex is the term as an application itself; another the second term in the application.



Problem

(1.8) Show that

$$(\lambda x . x x) y \not\equiv_{\beta} (\lambda x y . y x) x x$$

Solution. By Corollary 1.9.9, it suffices to prove the hypothesis with a proof of a common normal reduced form from LHS and RHS not existing.

Contradiction. By definition of \equiv_{β} , there exists The set of all terms attainable from β -reduction on $(\lambda x . x x) y$ and $(\lambda x y . y x) x x$ do not intersect. Therefore,

$$\neg \left(\exists L \in \Lambda, (\lambda x . x x) y \rightarrow_{\beta} L \wedge (\lambda x y . y x) x x \rightarrow_{\beta} L \right) \implies \neg \left((\lambda x . x x) y \equiv_{\beta} (\lambda x y . y x) x x \right)$$

■

Problem

(1.9) Define the combinators

$$K := \lambda x y . x$$

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

Prove that

$$\forall P Q \in \Lambda, K P Q \rightarrow_{\beta} P$$

$$\forall P Q R \in \Lambda, S P Q R \rightarrow_{\beta} P R (Q R)$$

Solution.

Proof.

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

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

■

Problem

(1.10) We define the church numerals

$$\begin{aligned}\text{zero} &:= \lambda f x . x \\ \text{one} &:= \lambda f x . f x \\ \text{two} &:= \lambda f x . f f x \\ &\dots \\ \text{num}_n &:= \lambda f x . f^n x\end{aligned}$$

And operations

$$\begin{aligned}\text{add} &:= \lambda n m f x . m f (n f x) \\ \text{mul} &:= \lambda n m f x . m (n f) x\end{aligned}$$

Show

- (a) $\text{add one one} \xrightarrow[\beta]{} \text{two}$
 (b) $\text{add one one} \not\xrightarrow[\beta]{} \text{mul one zero}$

Solution.

$$\begin{aligned}\text{(a) } \text{add one one} &= (\lambda n m f x . m f (n f x))(\lambda f x . f x)(\lambda f x . f x) \\ &\xrightarrow[\beta]{} (\lambda f x . (\lambda f x . f x) f ((\lambda f x . f x) f x)) \\ &\xrightarrow[\beta]{} (\lambda f x . (\lambda x . f x) f x) \\ &\xrightarrow[\beta]{} (\lambda f x . f f x) = \text{two} \\ \text{(b) } \text{mul one one} &= (\lambda n m f x . m (n f) x)(\lambda f x . f x)(\lambda f x . f x) \\ &\xrightarrow[\beta]{} \lambda f x . (\lambda f x . f x)((\lambda f x . f x) f) x \\ &\xrightarrow[\beta]{} \lambda f x . f x = \text{one}\end{aligned}$$

Because no intermediate form in the beta reduction process of the two terms are α -equivalent, by collorary 1.9.9 the two terms are not β -equivalent.

Problem

(1.11) We define

$$\text{succ} := \lambda m f x . f (m f x) \text{ s.t. } \forall \text{num}_n, \text{succ num}_n = \text{num}_{n+1}$$

Prove

$$\text{succ zero} \stackrel{\beta}{=} \text{one}$$

$$\text{succ one} \stackrel{\beta}{=} \text{two}$$

Solution. It suffices to provide a witness of a reduction chain from one side to the other to prove β -equivalence.

Proof.

$$\begin{aligned} \text{succ zero} &= (\lambda m f x . f (m f x))(\lambda f x . x) \\ &\rightarrow_{\beta} (\lambda f x . f ((\lambda f x . x) f x)) \\ &\rightarrow_{\beta} (\lambda f x . f x) = \text{one} \end{aligned}$$

The path $\text{succ zero} \rightarrow_{\beta} \text{one}$ derived above is the witness of a reduction chain from LHS to RHS.

$$\begin{aligned} \text{succ one} &= (\lambda m f x . f (m f x))(\lambda f x . f x) \\ &\rightarrow_{\beta} (\lambda f x . f ((\lambda f x . f x) f x)) \\ &\rightarrow_{\beta} (\lambda f x . f (f x)) = \text{two} \end{aligned}$$

The path $\text{succ one} \rightarrow_{\beta} \text{two}$ derived above is the witness of a reduction chain from LHS to RHS. ■

Problem

(1.12) We define the λ -terms \top_{λ} (true) and \perp_{λ} (false) and \neg_{λ} (not) by:

$$\begin{aligned} \top_{\lambda} &:= \lambda x y . x & \perp_{\lambda} &:= \lambda x y . y \\ \neg_{\lambda} &:= \lambda a . a \perp_{\lambda} \top_{\lambda} \end{aligned}$$

Show that

$$\neg_{\lambda}(\neg_{\lambda} \top_{\lambda}) \stackrel{\beta}{=} \top_{\lambda}$$

$$\neg_{\lambda}(\neg_{\lambda} \perp_{\lambda}) \stackrel{\beta}{=} \perp_{\lambda}$$

Solution. It suffices to provide a witness of a reduction chain from one side to the other to prove β -equivalence.

Proof.

$$\begin{aligned}
\neg_\lambda(\neg_\lambda \top_\lambda) &= \neg_\lambda((\lambda a . a \perp_\lambda \top_\lambda)(\lambda x y . x)) \\
&\xrightarrow[\beta]{\rightarrow} \neg_\lambda((\lambda x y . x) \perp_\lambda \top_\lambda) \\
&\xrightarrow[\beta]{\rightarrow} (\lambda a . a \perp_\lambda \top_\lambda) \perp_\lambda \\
&\xrightarrow[\beta]{\rightarrow} (\lambda x y . y) \perp_\lambda \top_\lambda \\
&\xrightarrow[\beta]{\rightarrow} \top_\lambda
\end{aligned}$$

$$\begin{aligned}
\neg_\lambda(\neg_\lambda \perp_\lambda) &= \neg_\lambda((\lambda a . a \perp_\lambda \top_\lambda)(\lambda x y . y)) \\
&\xrightarrow[\beta]{\rightarrow} \neg_\lambda((\lambda x y . y) \perp_\lambda \top_\lambda) \\
&\xrightarrow[\beta]{\rightarrow} \neg_\lambda \top_\lambda \\
&\xrightarrow[\beta]{\rightarrow} (\lambda a . a \perp_\lambda \top_\lambda)(\lambda x y . x) \\
&\xrightarrow[\beta]{\rightarrow} (\lambda x y . x) \perp_\lambda \top_\lambda \\
&\xrightarrow[\beta]{\rightarrow} \perp_\lambda
\end{aligned}$$

■

Problem

(1.13) Define

$$\text{iszero} := \lambda m . m (\lambda x . \perp_\lambda) \top_\lambda$$

Prove

$$\begin{aligned}
&\text{iszero zero} \xrightarrow[\beta]{\rightarrow} \top_\lambda \\
&\forall n \in \mathbb{N}^+, \text{iszero num}_n \xrightarrow[\beta]{\rightarrow} \perp_\lambda
\end{aligned}$$

Solution.

$$\begin{aligned}
\text{iszero zero} &= (\lambda m . m (\lambda x . \perp_\lambda) \top_\lambda)(\lambda f x . x) \\
&\xrightarrow[\beta]{\rightarrow} (\lambda f x . x)(\lambda x . \perp_\lambda) \top_\lambda \\
&\xrightarrow[\beta]{\rightarrow} \top_\lambda
\end{aligned}$$

$$\begin{aligned}
\text{iszero num}_n &= (\lambda m . m (\lambda x . \perp_\lambda) \top_\lambda) (\lambda f x . f^n x) \\
&\rightarrow_{\beta} (\lambda f x . f^n x) (\lambda x . \perp_\lambda) \top_\lambda \\
&\rightarrow_{\beta} (\lambda x . \perp_\lambda) ((\lambda x . \perp_\lambda)^{n-1} \top_\lambda) \rightarrow_{\beta} \perp_\lambda
\end{aligned}$$

Problem

(1.14) If-else can be modeled as

$$\text{ifelse} = \lambda x t f . x t f$$

Where when x , then t , else f . Prove correctness by applying \top_λ and \perp_λ on ifelse.

Solution.

$$\begin{aligned}
\text{ifelse } \top_\lambda &= (\lambda x t f . x t f) \top_\lambda \\
&\rightarrow_{\beta} (\lambda t f . (\lambda x y . x) t f) \rightarrow_{\beta} (\lambda t f . t) \\
\text{ifelse } \perp_\lambda &= (\lambda x t f . x t f) \perp_\lambda \\
&\rightarrow_{\beta} (\lambda t f . (\lambda x y . y) t f) \rightarrow_{\beta} (\lambda t f . f)
\end{aligned}$$

By applying the results to any two values, the correct corresponding value returns, ex, for ifelse \top_λ , t is always returned.

Problem

(1.15) Prove that $\Omega := (\lambda x . x x)(\lambda x . x x)$ does not have a β -nf.

Solution. Firstly let's prove Ω .

Proof. Induction on Ω 's only reduction path proves that every $\Omega \rightarrow_{\beta} \Omega_i = \Omega$. For the base case because Ω has one and only one redux, it could only reduce to Ω_1 which is equivalent to itself. For the inductive step, $\Omega_i = \Omega$, therefore $\Omega_i \rightarrow_{\beta} \Omega_{i+1}$ is still Ω .

By definition, a term having a β -nf requires the existence of a form in β -nf such that the term can reduce to. By induction, Ω only reduces to Ω , and Ω is not in β -nf because it contains β -redex. Therefore, Ω can never reduce to a β -nf, thus it does not have a β -nf. ■

Problem

(1.16) Let M be a λ -term with the following properties:

- M has a β -nf.
- There exists an infinite reduction path $M \equiv M_0 \xrightarrow{\beta} M_1 \xrightarrow{\beta} \dots$ on M .

Prove that every M_i has a β -nf, and give an example of M .

Solution. An example would be $(\lambda x y . y)\Omega$. Reduction can go on infinitely by reducing on Ω , but the β -nf of the term is $\lambda y . y$

Proof. Denote β -nf of M as M' . For any form in the reduction path, $M \xrightarrow{\beta} M_i$. In conjunction with $M \xrightarrow{\beta} M'$, by the Church-Rosser theorem, there exists L such that $M_i \xrightarrow{\beta} L$ and $M' \xrightarrow{\beta} L$. Because M' is in β -nf, L can only be M' , thus $M_i \xrightarrow{\beta} M'$, so M_i is capable of reducing to M' , a β -nf. Therefore, any form in the reduction path has a β -nf. ■

Problem

(1.17) If $M N$ is strongly normalizing, then both M and N are strongly normalizing.

Solution.

Proof. If M is not strongly normalizing, then there exists a reduction path $M_0 \xrightarrow{\beta} M_1 \xrightarrow{\beta} \dots$. Therefore, $M N$ would have had a reduction path $M N \xrightarrow{\beta} M_1 N \xrightarrow{\beta} \dots$ that is infinite, which contradicts with $M N$ being strongly normalizing. Vice versa for N . ■

Problem

(1.18) Let $L, M, N \in \Lambda$ such that $L \equiv_{\beta} M$ and $L \xrightarrow{\beta} N$. Moreover, N is in β -nf. Prove that $M \xrightarrow{\beta} N$.

Solution. Corollary 1.9.9.

Problem

(1.19) Define

$$U := \lambda z x . x (z z x) \quad \text{and} \quad Z := U U$$

Prove Z is a fixed point combinator.

Solution. Proving $\forall L \in \Lambda, L (Z L) \xrightarrow[\beta]{\rightarrow} Z L$.

Proof.

$$\begin{aligned} Z L &= (\lambda z x . x (z z x)) (\lambda z x . x (z z x)) L \\ &\xrightarrow[\beta]{\rightarrow} L ((\lambda z x . x (z z x)) (\lambda z x . x (z z x)) L) \xrightarrow[\beta]{\rightarrow} L (Z L) \end{aligned}$$

■

Problem

(1.20) Solve for $M \in \Lambda$ in each equation:

$$\begin{aligned} M &\stackrel{=}{\beta} \lambda x y . x M y \\ M x y z &\stackrel{=}{\beta} x y z M \end{aligned}$$

Solution. By the property of the Y combinator:

$$f (Y f) = Y f$$

The first equation can be remodeled as

$$M \stackrel{=}{\beta} L M \text{ where } L = \lambda m x y . x m y$$

Solving for fixed point of L :

$$\begin{aligned} M &\equiv Y L \equiv L (Y L) \\ &= (\lambda x . L (x x)) (\lambda x . L (x x)) \\ &= (\lambda x . (\lambda m u v . u m v) (x x)) (\lambda x . (\lambda m u v . u m v) (x x)) \\ &\xrightarrow[\beta]{\rightarrow} (\lambda x . (\lambda u v . u (x x) v)) (\lambda x . (\lambda u v . u (x x) v)) \end{aligned}$$

The second equation can be η -reduced on both sides:

$$\begin{aligned} M x y z &\stackrel{=}{\beta} x y z M \\ M &\stackrel{=}{\beta} \lambda x y z . x y z M \end{aligned}$$

Remodeling equation:

$$M = N M \text{ where } M = \lambda m x y z . x y z m$$

then

$$\begin{aligned}
M &\equiv Y\ N \equiv N\ (Y\ N) \\
&= (\lambda x . N\ (x\ x))(\lambda x . N\ (x\ x)) \\
&= (\lambda x . (\lambda m\ u\ y\ z . u\ y\ z\ m)(x\ x))(\lambda x . (\lambda m\ u\ y\ z . u\ y\ z\ m)(x\ x)) \\
&\xrightarrow[\beta]{} (\lambda x . (\lambda u\ y\ z . u\ y\ z\ (x\ x)))(\lambda x . (\lambda u\ y\ z . u\ y\ z\ (x\ x)))
\end{aligned}$$