

EXERCISES

CHAPTER 3

SEAN LI ¹

1. Reducted

Definition Some rules for reference.

$$\frac{x : \sigma \in \Gamma \quad \Gamma \text{ is a } \lambda 2 \text{ context}}{\Gamma \vdash x : \sigma} \text{ (T-Var)} \quad \frac{\Gamma \vdash M : \sigma \rightarrow \tau \quad \Gamma \vdash N : \sigma}{\Gamma \vdash M N : \tau} \text{ (T-App)}$$
$$\frac{\Gamma, x : \sigma \vdash M : \tau}{\Gamma \vdash \lambda x : \sigma. M : \sigma \rightarrow \tau} \text{ (T-Abst)} \quad \frac{\alpha \in \mathbb{T}_2 \quad \text{FV } \alpha \subseteq \text{dom } \Gamma}{\alpha : * \in \Gamma} \text{ (T-Form)}$$
$$\frac{\Gamma \vdash M : \Pi_{\alpha : *} A \quad \Gamma \vdash B : *}{\Gamma \vdash M B : A [\alpha := B]} \text{ (T2-App)} \quad \frac{\Gamma, \alpha : * \vdash M : A}{\Gamma \vdash \lambda \alpha : *. M : \Pi_{\alpha : *} A} \text{ (T2-Abst)}$$

In this document, convention is that all type judgements in a proof tree, unless stated otherwise, is derived from a single and unique $\lambda 2$ context per tree. Multiple conclusions might be drawn on a single line from usage of the same inference rule for compactness. Eg:

ex 1. $\alpha, \beta : *$ **T-Form**

Is shorthand for

ex 1. $\Gamma \vdash \alpha : *$ **T-Form**

ex 2. $\Gamma \vdash \beta : *$ **T-Form**

Problem

(3.1) How many $\lambda 2$ contexts consisting of four and only four declarations

- (1) $\Gamma \vdash \alpha : *$
- (2) $\Gamma \vdash \beta : *$
- (3) $\Gamma \vdash f : \alpha \rightarrow \beta$
- (4) $\Gamma \vdash x : \alpha$

Solution. The last two declarations depende on the first two. Therefore this is an easy combinatorics problem: $2! \times 2! = 4$ contexts:

$$\begin{array}{ll} 1 - 2 - 3 - 4 & 1 - 2 - 4 - 3 \\ 2 - 1 - 3 - 4 & 2 - 1 - 4 - 3 \end{array}$$

Problem

(3.2) Give a full derivation in $\lambda 2$ to show the following type term is legal:

$$M \equiv \lambda \alpha, \beta, \gamma : * . \lambda f : \alpha \rightarrow \beta. \lambda g : \beta \rightarrow \gamma. \lambda x : \alpha. g(f x)$$

Solution.

		Bound
1.	$\alpha : *$	
2.	$\beta : *$	Bound
3.	$\gamma : *$	Bound
4.	$f : \alpha \rightarrow \beta$	Bound
5.	$g : \beta \rightarrow \gamma$	Bound
6.	$x : \alpha$	Bound
7.	$\alpha, \beta, \gamma : *$	T-Form
8.	$\alpha \rightarrow \beta, \beta \rightarrow \gamma : *$	T-Form
9.	$f : \alpha \rightarrow \beta, x : \alpha$	T-Var
10.	$f x : \beta$	8,8 T-App
11.	$g : \beta \rightarrow \gamma$	T-Var
12.	$g(f x) : \gamma$	11,10 T-App
13.	$\lambda x : \alpha. g(f x) : \alpha \rightarrow \gamma$	12 T-Abst
14.	$\lambda g : \beta \rightarrow \gamma. \lambda x : \alpha. g(f x) : (\beta \rightarrow \gamma) \rightarrow \alpha \rightarrow \gamma$	13 T-Abst
15.	$\lambda f : \alpha \rightarrow \beta. \lambda g : \beta \rightarrow \gamma. \lambda x : \alpha. g(f x) : (\alpha \rightarrow \beta) \rightarrow (\beta \rightarrow \gamma) \rightarrow \alpha \rightarrow \gamma$	14 T-Abst

16.	$\boxed{\begin{array}{c} \lambda\gamma : * . \lambda f : \alpha \rightarrow \beta. \lambda g : \beta \rightarrow \gamma. \lambda x : \alpha. g(f x) \\ : \Pi\gamma : * . (\alpha \rightarrow \beta) \rightarrow (\beta \rightarrow \gamma) \rightarrow \alpha \rightarrow \gamma \end{array}}$	15 T2-Abst
17.	$\boxed{\begin{array}{c} \lambda\beta, \gamma : * . \lambda f : \alpha \rightarrow \beta. \lambda g : \beta \rightarrow \gamma. \lambda x : \alpha. g(f x) \\ : \Pi\beta, \gamma : * . (\alpha \rightarrow \beta) \rightarrow (\beta \rightarrow \gamma) \rightarrow \alpha \rightarrow \gamma \end{array}}$	16 T2-Abst
18.	$\begin{array}{c} \lambda\alpha, \beta, \gamma : * . \lambda f : \alpha \rightarrow \beta. \lambda g : \beta \rightarrow \gamma. \lambda x : \alpha. g(f x) \\ : \Pi\alpha, \beta, \gamma : * . (\alpha \rightarrow \beta) \rightarrow (\beta \rightarrow \gamma) \rightarrow \alpha \rightarrow \gamma \end{array}$	17 T2-Abst

Problem

(3.3 a) Given M in 3.2, and a context Γ such that

$$\Gamma \vdash \text{nat} : *$$

$$\Gamma \vdash \text{bool} : *$$

$$\Gamma \vdash \text{succ} : \text{nat} \rightarrow \text{nat}$$

$$\Gamma \vdash \text{even} : \text{nat} \rightarrow \text{bool}$$

Prove $M \text{ nat } \text{nat } \text{bool } \text{succ } \text{even}$ is legal.

Solution. Proof by deriving the term's type.

Proof.

1. $M : \Pi\alpha, \beta, \gamma : * . (\alpha \rightarrow \beta) \rightarrow (\beta \rightarrow \gamma) \rightarrow \alpha \rightarrow \gamma$ **T-Var**
 2. $\text{nat}, \text{bool} : *$ **T-Form**
 3. $M \text{ nat} : \Pi\beta, \gamma : * . (\text{nat} \rightarrow \beta) \rightarrow (\beta \rightarrow \gamma) \rightarrow \text{nat} \rightarrow \gamma$ **2,1 T2-App**
 4. $M \text{ nat } \text{nat} : \Pi\gamma : * . (\text{nat} \rightarrow \text{nat}) \rightarrow (\text{nat} \rightarrow \gamma) \rightarrow \text{nat} \rightarrow \gamma$ **2,3 T2-App**
 - 5.
- $M \text{ nat } \text{nat } \text{bool} : (\text{nat} \rightarrow \text{nat}) \rightarrow (\text{nat} \rightarrow \text{bool}) \rightarrow \text{nat} \rightarrow \text{bool}$ **2,3 T2-App**
6. $\text{succ} : \text{nat} \rightarrow \text{nat}, \text{even} : \text{nat} \rightarrow \text{bool}$ **T-Var**
 7. $M \text{ nat } \text{nat } \text{bool } \text{succ} : (\text{nat} \rightarrow \text{bool}) \rightarrow \text{nat} \rightarrow \text{bool}$ **6,5 T-App**
 8. $M \text{ nat } \text{nat } \text{bool } \text{succ } \text{even} : \text{nat} \rightarrow \text{bool}$ **6,7 T-App**

■

Problem

(3.3 b.i) Prove $\lambda x : \text{nat}. \text{even}(\text{succ } x)$ via 3.3 a.

Solution. The result of beta reduction on the term in 3.3 a is what we are proving.

Proof.

$$\begin{aligned} M &\equiv \text{nat nat bool succ even} \\ &\equiv (\lambda \alpha, \beta, \gamma, f, g. \lambda x : \alpha. g(f x)) \text{ nat nat bool succ even} \\ &\xrightarrow{\beta} (\lambda f : \text{nat} \rightarrow \text{nat}. \lambda g : \text{nat} \rightarrow \text{bool}. \lambda x : \text{nat}. g(f(x))) \text{ succ even} \\ &\xrightarrow{\beta} (\lambda x : \text{nat}. \text{even}(\text{succ } x)) \end{aligned}$$

By the subject reduction lemma, $\lambda x : \text{nat}. \text{even}(\text{succ } x) : \text{nat} \rightarrow \text{bool}$, thus is legal. ■

Problem

(3.3 b.ii) Prove $\lambda x : \text{nat}. \text{even}(\text{succ } x)$ via derivation in the context provided in 3.3 a.

Solution.

Proof.

1.	$\text{nat}, \text{bool} : *$	T-Form
2.	$x : \text{nat}$	Bound
3.	$\text{succ} : \text{nat} \rightarrow \text{nat}$	T-Var
4.	$x : \text{nat}$	T-Var
5.	$\text{succ } x : \text{nat}$	3,4 T-App
6.	$\text{even} : \text{nat} \rightarrow \text{bool}$	T-Var
7.	$\text{even}(\text{succ } x) : \text{bool}$	6,5 T-App
8.	$\lambda x : \text{nat}. \text{even}(\text{succ } x) : \text{nat} \rightarrow \text{bool}$	7 T-Abst

Problem

(3.4) Give a shorthanded (omit T-Var and T-Form) derivation in $\lambda 2$ to show the following term is valid in $\Gamma \equiv \text{nat} : *, \text{bool} : *$

$$(\lambda\alpha, \beta : * . \lambda f : \alpha \rightarrow \alpha. \lambda g : \alpha \rightarrow \beta. \lambda x : \alpha. g(f(fx))) \text{ nat bool}$$

Solution.

Proof.

1.	$\alpha, \beta : *$	Bound
2.	$f : \alpha \rightarrow \alpha$	Bound
3.	$g : \alpha \rightarrow \beta$	Bound
4.	$x : \alpha$	Bound
5.	$f x : \alpha$	$^{*,*} \text{T-App}$
6.	$f(fx) : \alpha$	$^{*,5} \text{T-App}$
7.	$g(f(fx)) : \beta$	$^{*,6} \text{T-App}$
8.	$\lambda x : \alpha. g(f(fx)) : \alpha \rightarrow \beta$	7 T-Abst
9.	$\lambda g : \alpha \rightarrow \beta. \lambda x : \alpha. g(f(fx)) : (\alpha \rightarrow \beta) \rightarrow \alpha \rightarrow \beta$	8 T-Abst
10.	$\lambda f : \alpha \rightarrow \alpha. \lambda g : \alpha \rightarrow \beta. \lambda x : \alpha. g(f(fx))$	
	$: (\alpha \rightarrow \alpha) \rightarrow (\alpha \rightarrow \beta) \rightarrow \alpha \rightarrow \beta$	9 T-Abst
11.		
	$\lambda\alpha, \beta : * . \lambda f : \alpha \rightarrow \alpha. \lambda g : \alpha \rightarrow \beta. \lambda x : \alpha. g(f(fx))$	
	$: \Pi\alpha, \beta : * . (\alpha \rightarrow \alpha) \rightarrow (\alpha \rightarrow \beta) \rightarrow \alpha \rightarrow \beta$	10 T2-Abst
12.		
	$(\lambda\alpha, \beta : * . \lambda f : \alpha \rightarrow \alpha. \lambda g : \alpha \rightarrow \beta. \lambda x : \alpha. g(f(fx))) \text{ nat}$	
	$: \Pi\beta : * . (\text{nat} \rightarrow \text{nat}) \rightarrow (\text{nat} \rightarrow \beta) \rightarrow \text{nat} \rightarrow \beta$	$^{*,11} \text{T2-App}$
13.		
	$(\lambda\alpha, \beta : * . \lambda f : \alpha \rightarrow \alpha. \lambda g : \alpha \rightarrow \beta. \lambda x : \alpha. g(f(fx))) \text{ nat bool}$	
	$: \Pi\beta : * . (\text{nat} \rightarrow \text{nat}) \rightarrow (\text{nat} \rightarrow \text{bool}) \rightarrow \text{nat} \rightarrow \text{bool}$	$^{*,12} \text{T2-App}$

Problem

(3.5 a) Let $\perp \equiv \Pi\alpha : * . \alpha$. Prove \perp is legal.

Solution. Here a notion called kind checking is introduced. This has not yet been discussed in this book (?)

Proof.

- | | |
|--|----------------|
| 1. $\alpha : *$ | Bound |
| 2. $\boxed{\alpha : *}$ | T-Form |
| 3. $\Pi\alpha : *. \alpha : * \rightarrow *$ | T2-Abst |

■

Problem

(3.5 b) Consider the context $\Gamma \equiv \beta : *, x : \perp$. Find an inhabitant of type β under Γ .

Solution. $x \beta$ is. Because x is of second-order type, it must be parametric to a type, thus x is of form $\lambda\alpha : *. M$ where $\Gamma, \alpha : * \vdash M : \alpha$.

Proof.

- | | |
|--------------------------------|-------------------|
| 1. $x : \Pi\alpha : *. \alpha$ | T-Var |
| 2. $\beta : *$ | T-Form |
| 3. $x \beta : \beta$ | 1,2 T2-App |

■

Problem

(3.5 c) Give three inhabitants of $\beta \rightarrow \beta$ in β -nf under Γ in 3.5 b.

Solution.

1. $\lambda y : \beta. y$.

Proof.

- | | |
|---|-----------------|
| 1. $y : \beta$ | Bound |
| 2. $\boxed{y : \beta}$ | T-Var |
| 3. $\lambda y : \beta. y : \beta \rightarrow \beta$ | 2 T-Abst |

■

2. $\lambda y : \beta. x \beta$.

Proof.

1.	$y : \beta$	Bound
2.	$x : \Pi\alpha : * . \alpha$	T-Var
3.	$\beta : *$	T-Form
4.	$x \beta : \beta$	2,3 T2-App
5.	$\lambda y : \beta. x \beta : \beta \rightarrow \beta$	4 T-Abst

■

$$3. \lambda y : \beta. x (\beta \rightarrow \beta) y.$$

Proof.

1.	$y : \beta$	Bound
2.	$x : \Pi\alpha : * . \alpha$	T-Var
3.	$\beta \rightarrow \beta : *$	T-Form
4.	$x (\beta \rightarrow \beta) : \beta \rightarrow \beta$	2,3 T2-App
5.	$y : \beta$	T-Var
6.	$x (\beta \rightarrow \beta) y : \beta$	4,5 T-App
7.	$\lambda y : \beta. x (\beta \rightarrow \beta) y : \beta \rightarrow \beta$	5 T-Abst

■

Problem

(3.5 d) Prove that the following terms inhabit the same type in Γ :

$$\lambda f : \beta \rightarrow \beta \rightarrow \beta. f (x \beta)(x \beta)$$

$$x ((\beta \rightarrow \beta \rightarrow \beta) \rightarrow \beta)$$

Solution. We simply derive the types.

First Term.

1.	$f : \beta \rightarrow \beta \rightarrow \beta$	Bound
2.	$f : \beta \rightarrow \beta \rightarrow \beta$	T-Var
3.	$x : \Pi\alpha : * . \alpha$	T-Var
4.	$\beta : *$	T-Form
5.	$x \beta : \beta$	3,4 T2-App
6.	$f (x \beta) : \beta \rightarrow \beta$	2,5 T-App

7. $\boxed{f(x\beta)(x\beta) : \beta}$ **6,5 T-App**
8. $\lambda f : \beta \rightarrow \beta \rightarrow \beta. f(x\beta)(x\beta) : (\beta \rightarrow \beta \rightarrow \beta) \rightarrow \beta$ **6 T-Abst**

■

Second Term.

1. $(\beta \rightarrow \beta \rightarrow \beta) \rightarrow \beta : *$ **T-Form**
2. $x : \Pi \alpha : *. \alpha$ **T-Var**
3. $x((\beta \rightarrow \beta \rightarrow \beta) \rightarrow \beta) : (\beta \rightarrow \beta \rightarrow \beta) \rightarrow \beta$ **2,1 T2-App**

■

The two terms were shown to both inhabit $(\beta \rightarrow \beta \rightarrow \beta) \rightarrow \beta$.

Problem

(3.6 a) Find inhabitant of type

$$\Pi \alpha, \beta : *. (\text{nat} \rightarrow \alpha) \rightarrow (\alpha \rightarrow \text{nat} \rightarrow \beta) \rightarrow \text{nat} \rightarrow \beta$$

In context $\Gamma \equiv \text{nat} : *$.

Solution.

$$\lambda \alpha, \beta : *. \lambda x : \text{nat} \rightarrow \alpha. \lambda y : (\alpha \rightarrow \text{nat} \rightarrow \beta). \lambda z : \text{nat}. y(xz)z$$

Proof.

- | | |
|--|--------------------|
| 1. $\alpha, \beta : *$ | Bound |
| 2. $\text{nat} \rightarrow \alpha : *$ | T-Form |
| 3. $x : \text{nat} \rightarrow \alpha$ | Bound |
| 4. $\alpha \rightarrow \text{nat} \rightarrow \beta : *$ | T-Form |
| 5. $y : \alpha \rightarrow \text{nat} \rightarrow \beta$ | Bound |
| 6. $\text{nat} : *$ | Bound |
| 7. $z : \text{nat}$ | Bound |
| 8. $y : \alpha \rightarrow \text{nat} \rightarrow \beta$ | T-Var |
| 9. $x : \text{nat} \rightarrow \alpha$ | T-Var |
| 10. $z : \text{nat}$ | T-Var |
| 11. $xz : \alpha$ | 9,10 T-App |
| 12. $y(xz) : \text{nat} \rightarrow \beta$ | 8,11 T-App |
| 13. $y(xz)z : \beta$ | 12,10 T-App |

14.	$\boxed{\lambda z : \text{nat}. y(xz) z : \text{nat} \rightarrow \beta}$	13 T-Abst
15.	$\boxed{\lambda y : \alpha \rightarrow \text{nat} \rightarrow \beta. \lambda z : \text{nat}. y(xz) z : (\alpha \rightarrow \text{nat} \rightarrow \beta) \rightarrow \text{nat} \rightarrow \beta}$	14 T-Abst
16.	$\boxed{\lambda x : \text{nat} \rightarrow \alpha. y : \alpha \rightarrow \text{nat} \rightarrow \beta. \lambda z : \text{nat}. y(xz) z : (\text{nat} \rightarrow \alpha) \rightarrow (\alpha \rightarrow \text{nat} \rightarrow \beta) \rightarrow \text{nat} \rightarrow \beta}$	15 T2-Abst
17.	$\lambda \alpha, \beta : *. x : \text{nat} \rightarrow \alpha. y : \alpha \rightarrow \text{nat} \rightarrow \beta. \lambda z : \text{nat}. y(xz) z : \Pi \alpha, \beta : *. (\text{nat} \rightarrow \alpha) \rightarrow (\alpha \rightarrow \text{nat} \rightarrow \beta) \rightarrow \text{nat} \rightarrow \beta$	16 T2-Abst

■

Problem

(3.6 b) Find inhabitant of type

$$\Pi \delta : *. ((\alpha \rightarrow \gamma) \rightarrow \delta) \rightarrow (\alpha \rightarrow \beta) \rightarrow (\beta \rightarrow \gamma) \rightarrow \delta$$

In context $\Gamma \equiv \alpha : *, \beta : *, \gamma : *$

Solution.

$$\lambda \delta : *. \lambda x : (\alpha \rightarrow \gamma) \rightarrow \delta. \lambda y : \alpha \rightarrow \beta. \lambda z : \beta \rightarrow \gamma. x(\lambda u : \alpha. z(yu))$$

A derivation in shorthand will be given (omitting T-Form / T-Var)

Proof.

1.	$\delta : *$	Bound
2.	$x : (\alpha \rightarrow \gamma) \rightarrow \delta$	Bound
3.	$y : \alpha \rightarrow \beta$	Bound
4.	$z : \beta \rightarrow \gamma$	Bound
5.	$u : \alpha$	Bound
6.	$y u : \beta$	*,* T-App
7.	$z(yu) : \gamma$	
*₆ T-App		
8.	$\lambda u : \alpha. z(yu) : \alpha \rightarrow \gamma$	7 T-Abst
9.	$x(\lambda u : \alpha. z(yu)) : \delta$	8 T-Abst
10.	$\lambda z : \beta \rightarrow \gamma. x(\lambda u : \alpha. z(yu)) : (\beta \rightarrow \gamma) \rightarrow \delta$	9 T-Abst

$$11. \quad \boxed{\begin{array}{l} \lambda y : \alpha \rightarrow \beta. \lambda z : \beta \rightarrow \gamma. x (\lambda u : \alpha. z (y u)) \\ : (\alpha \rightarrow \beta) \rightarrow (\beta \rightarrow \gamma) \rightarrow \delta \end{array}}$$

10 T-Abst

$$12. \quad \boxed{\begin{array}{l} \lambda x : (\alpha \rightarrow \gamma) \rightarrow \delta. \lambda y : \alpha \rightarrow \beta. \lambda z : \beta \rightarrow \gamma. x (\lambda u : \alpha. z (y u)) \\ : ((\alpha \rightarrow \gamma) \rightarrow \beta) \rightarrow (\alpha \rightarrow \beta) \rightarrow (\beta \rightarrow \gamma) \rightarrow \delta \end{array}}$$

11 T-Abst

13.

$$\begin{aligned} \lambda \delta : * . \lambda x : (\alpha \rightarrow \gamma) \rightarrow \delta. \lambda y : \alpha \rightarrow \beta. \lambda z : \beta \rightarrow \gamma. x (\lambda u : \alpha. z (y u)) \\ : \Pi \delta : * . ((\alpha \rightarrow \gamma) \rightarrow \delta) \rightarrow (\alpha \rightarrow \beta) \rightarrow (\beta \rightarrow \gamma) \rightarrow \delta \end{aligned}$$

12 T2-Abst

■

Problem

(3.6 c) Find inhabitant of type

$$\Pi \alpha, \beta, \gamma : * . (\alpha \rightarrow (\beta \rightarrow \alpha) \rightarrow \gamma) \rightarrow \alpha \rightarrow \gamma$$

In the empty context

Solution.

$$\lambda \alpha, \beta, \gamma : * . \lambda f : (\alpha \rightarrow (\beta \rightarrow \alpha) \rightarrow \gamma). \lambda x : \alpha. f x (\lambda u : \beta. x)$$

Proof.

1.	α, β, γ	Bound
2.	$f : \alpha \rightarrow (\beta \rightarrow \alpha) \rightarrow \gamma$	Bound
3.	$x : \alpha$	Bound
4.	$f x : (\beta \rightarrow \alpha) \rightarrow \gamma$	*,* T-App
5.	$u : \beta$	Bound
6.	$x : \alpha$	T-Var
7.	$\lambda u : \beta. x : \beta \rightarrow \alpha$	6 T-Abst
8.	$f x (\lambda u : \beta. x) : \gamma$	4,7 T-App
9.	$\lambda x : \alpha. f x (\lambda u : \beta. x) : \alpha \rightarrow \gamma$	8 T-Abst
10.	$\lambda f : \alpha \rightarrow (\beta \rightarrow \alpha) \rightarrow \gamma. \lambda x : \alpha. f x (\lambda u : \beta. x)$	
	$: (\alpha \rightarrow (\beta \rightarrow \alpha) \rightarrow \gamma) \rightarrow \alpha \rightarrow \gamma$	9 T-Abst

11.

$$\lambda\alpha, \beta, \gamma : * . \lambda f : \alpha \rightarrow (\beta \rightarrow \alpha) \rightarrow \gamma . \lambda x : \alpha . f x (\lambda u : \beta . x)$$

$$: \Pi\alpha, \beta, \gamma : * . (\alpha \rightarrow (\beta \rightarrow \alpha) \rightarrow \gamma) \rightarrow \alpha \rightarrow \gamma$$

10 T2-Abst

■

Problem

(3.7) Let $\perp \equiv \Pi\alpha : * . \alpha$ and context $\Gamma \equiv \alpha : *, \beta : *, x : \alpha \rightarrow \perp, f : (\alpha \rightarrow \alpha) \rightarrow \alpha$. Give a derivation that successively calculate an inhabitant of α and β , both in context Γ .

Solution. Have $M : \alpha := f (\lambda n : \alpha . n)$. Then $\Gamma \vdash x M \beta : \beta$.

Typing M.

1. $f : (\alpha \rightarrow \alpha) \rightarrow \alpha$ **T-Var**
2. $n : \alpha$ **Bound**
3. $\boxed{n : \alpha}$ **T-Var**
4. $\lambda n : \alpha . n : \alpha \rightarrow \alpha$ **3 T-Abst**
5. $f (\lambda n : \alpha . n) : \alpha$ **1,4 T-App**

■

Typing $x M \beta$.

1. $M : \alpha$ **T-Var**
2. $x : \alpha \rightarrow \Pi\alpha : * . \alpha$ **T-Var**
3. $M x : \Pi\alpha : * . \alpha$ **2,1 T-App**
4. $M x \beta : \beta$ **3,* T2-App**

■

Problem

(3.8) Recall $K \equiv \lambda x y . x \in \Lambda$ from untyped lambda calculus. Consider the following types

$$T_1 \equiv \Pi \alpha, \beta : * . \alpha \rightarrow \beta \rightarrow \alpha \quad T_2 \equiv \Pi \alpha : * . \alpha \rightarrow (\Pi \beta : * . \beta \rightarrow \alpha)$$

Find inhabitants of both type $t_1 : T_1$ and $t_2 : T_2$ under the empty context, which may be considered the closed $\lambda 2$ form of $K \in \Lambda_{T_2}$.

Solution.

$$\lambda \alpha, \beta : * . \lambda x : \alpha. \lambda y : \beta. x$$

$$\lambda \alpha : * . \lambda x : \alpha. \lambda \beta : * . \lambda y : \beta. x$$

First Form.

1.	$\alpha, \beta : *$	Bound
2.	$x : \alpha$	Bound
3.	$y : \beta$	Bound
4.	$x : \alpha$	T-Var
5.	$\lambda y : \beta. x : \beta \rightarrow \alpha$	4 T-Abst
6.	$\lambda x : \alpha. \lambda y : \beta. x : \alpha \rightarrow \beta \rightarrow \alpha$	5 T-Abst
7.	$\lambda \alpha, \beta : * . \lambda x : \alpha. \lambda y : \beta. x : \Pi \alpha, \beta : * . \alpha \rightarrow \beta \rightarrow \alpha$	5 T2-Abst

■

Second Form.

1.	$\alpha : *$	Bound
2.	$x : \alpha$	Bound
3.	$\beta : *$	Bound
4.	$y : \beta$	Bound
5.	$x : \alpha$	T-Var
6.	$\lambda y : \beta. x : \beta \rightarrow \alpha$	5 T-Abst
7.	$\lambda \beta : * . \lambda y : \beta. x : \Pi \beta : * . \beta \rightarrow \alpha$	6 T2-Abst
8.	$\lambda x : \alpha. \lambda \beta : * . \lambda y : \beta. x : \alpha \rightarrow (\Pi \beta : * . \beta \rightarrow \alpha)$	7 T-Abst
9.	$\lambda \alpha : * . \lambda x : \alpha. \lambda \beta : * . \lambda y : \beta. x : \Pi \alpha : * . \alpha \rightarrow (\Pi \beta : * . \beta \rightarrow \alpha)$	8 T2-Abst

■

Problem

(3.9) Pretype the combinator

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

In closed form (typable in an empty context) in Λ_{T2} .

Solution.

$$S \equiv \lambda \alpha, \beta, \gamma : * . \lambda x : \alpha \rightarrow \beta \rightarrow \gamma. \lambda y : \alpha \rightarrow \beta. \lambda z : \alpha. x z (y z)$$

Proof.

1.	$\alpha, \beta, \gamma : *$	Bound
2.	$x : \alpha \rightarrow \beta \rightarrow \gamma$	Bound
3.	$y : \alpha \rightarrow \beta$	Bound
4.	$z : \alpha$	Bound
5.	$x z : \beta \rightarrow \gamma$	*,* T-App
6.	$y x : \beta$	*,* T-App
7.	$x z (y x) : \gamma$	5,6 T-App
8.	$\lambda z : \alpha. x z (y x) : \alpha \rightarrow \gamma$	7 T-Abst
9.	$\lambda y : \alpha \rightarrow \beta. \lambda z : \alpha. x z (y x) : (\alpha \rightarrow \beta) \rightarrow \alpha \rightarrow \gamma$	8 T-Abst
10.	$\lambda x : \alpha \rightarrow \beta \rightarrow \gamma. \lambda y : \alpha \rightarrow \beta. \lambda z : \alpha. x z (y x)$	9 T-Abst
11.	$: (\alpha \rightarrow \beta \rightarrow \gamma) \rightarrow (\alpha \rightarrow \beta) \rightarrow \alpha \rightarrow \gamma$	
		10 T2-Abst

Problem

(3.10 a) Consider the term

$$M \equiv \lambda x : \Pi \alpha : * . \alpha \rightarrow \alpha. x (\sigma \rightarrow \sigma)(x \sigma)$$

Prove that M is legal.

Solution. For a term to be legal there must exist a context so that the term could be typed. Here, a witness context is $\Gamma \equiv \sigma : *$.

Proof.

1. $x : \Pi\alpha : * . \alpha \rightarrow \alpha$ 2. $x (\sigma \rightarrow \sigma) : (\sigma \rightarrow \sigma) \rightarrow (\sigma \rightarrow \sigma)$ 3. $x \sigma : \sigma \rightarrow \sigma$ 4. $x (\sigma \rightarrow \sigma)(x \sigma) : \sigma \rightarrow \sigma$ 5.	Bound $\ast, \ast \text{ T2-App}$ $\ast, \ast \text{ T2-App}$ 2,3 T-App 4 T-Abst
	■

Problem

(3.10 b) Find a term N such that $M N$ is legal and may be considered to be a way to add type information to $(\lambda x . x x)(\lambda y . y)$

Solution.

$$M \sigma N \equiv (\lambda x : \Pi\alpha : * . \alpha \rightarrow \alpha. x (\sigma \rightarrow \sigma)(x \sigma))\sigma(\lambda y : \sigma. y)$$

Is the same as $(\lambda x . x x)(\lambda y . y)$ modulo type annotations.

Proof.

1. $M : (\Pi\alpha : * . \alpha \rightarrow \alpha) \rightarrow \sigma \rightarrow \sigma$ 2. $M \sigma : (\sigma \rightarrow \sigma) \rightarrow \sigma \rightarrow \sigma$ 3. $y : \sigma$ 4. $y : \sigma$ 5. have $N := \lambda y : \sigma. y : \sigma \rightarrow \sigma$ 6. $M \sigma N : \sigma \rightarrow \sigma$	T-Var 1, * T2-App Bound T-Var 4 T-Abst 2,5 T-Abst
	■

Problem

(3.11) Recall $\perp \equiv \Pi\alpha : * . \alpha$ from 3.5. Type and prove the following term legal:

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

Solution.

Proof. The type $\perp \rightarrow \perp$ is closed and well formed. Therefore, the term is legal.

1.	$\perp : * \equiv \Pi\alpha : *. \alpha$	T-Form
2.	$x : \perp$	Bound
3.	$x(\perp \rightarrow \perp \rightarrow \perp) : \perp \rightarrow \perp \rightarrow \perp$	$^{**} \text{ T2-App}$
4.	$x(\perp \rightarrow \perp) : \perp \rightarrow \perp$	$^{**} \text{ T2-App}$
5.	$x(\perp \rightarrow \perp)x : \perp$	$4,* \text{ T-App}$
6.	$x(\perp \rightarrow \perp \rightarrow \perp)(x(\perp \rightarrow \perp)x) : \perp \rightarrow \perp$	$3,5 \text{ T-App}$
7.	$x(\perp \rightarrow \perp \rightarrow \perp)x : \perp \rightarrow \perp$	$3,* \text{ T-App}$
8.	$x(\perp \rightarrow \perp \rightarrow \perp)x x : \perp$	$7,* \text{ T-App}$
9.	$x(\perp \rightarrow \perp \rightarrow \perp)(x(\perp \rightarrow \perp)x)(x(\perp \rightarrow \perp \rightarrow \perp)x x) : \perp$	$6,8 \text{ T-App}$
10.	$\lambda x : \perp. x(\perp \rightarrow \perp \rightarrow \perp)(x(\perp \rightarrow \perp)x)$ $(x(\perp \rightarrow \perp \rightarrow \perp)x x) : \perp \rightarrow \perp$	9 T-Abst

■

Problem

(3.12) Given the Polymorphic Church Numerals:

$$\begin{aligned} \text{nat} &\in \mathbb{T}_2 := \Pi\alpha : *. (\alpha \rightarrow \alpha) \rightarrow \alpha \rightarrow \alpha \\ \bar{0} &\equiv \lambda\alpha : *. \lambda f : \alpha \rightarrow \alpha. \lambda x : \alpha. x : \text{nat} \\ \bar{1} &\equiv \lambda\alpha : *. \lambda f : \alpha \rightarrow \alpha. \lambda x : \alpha. f x : \text{nat} \\ \bar{2} &\equiv \lambda\alpha : *. \lambda f : \alpha \rightarrow \alpha. \lambda x : \alpha. f(f x) : \text{nat} \\ \text{succ} &\equiv \lambda n : \text{nat}. \lambda\beta : *. \lambda f : \beta \rightarrow \beta. \lambda x : \beta. f(n\beta f x) \end{aligned}$$

Prove that

$$\begin{aligned} \text{succ } \bar{0} &\underset{\beta}{=} \bar{1} \\ \text{succ } \bar{1} &\underset{\beta}{=} \bar{2} \end{aligned}$$

Solution.

$$\begin{aligned}
& \text{succ } \bar{0} \\
& \equiv (\lambda n : \text{nat}. \lambda \beta : * . \lambda f : \beta \rightarrow \beta. \lambda x : \beta. f(n \beta f x))(\lambda \alpha : * . \lambda f : \alpha \rightarrow \alpha. \lambda x : \alpha. x) \\
& \xrightarrow{\beta} (\lambda \beta : * . \lambda f : \beta \rightarrow \beta. \lambda x : \beta. f((\lambda \alpha : * . \lambda f : \alpha \rightarrow \alpha. \lambda x : \alpha. x) \beta f x)) \\
& \xrightarrow{\beta_{T2}} (\lambda \beta : * . \lambda f : \beta \rightarrow \beta. \lambda x : \beta. f((\lambda f : \beta \rightarrow \beta. \lambda x : \beta. x) f x)) \\
& \xrightarrow[\beta]{\beta} (\lambda \beta : * . \lambda f : \beta \rightarrow \beta. \lambda x : \beta. f x) \xrightarrow[\alpha_{T2}]{\beta \rightarrow \alpha} \bar{1} \\
\\
& \text{succ } \bar{1} \\
& \equiv (\lambda n : \text{nat}. \lambda \beta : * . \lambda f : \beta \rightarrow \beta. \lambda x : \beta. f(n \beta f x))(\lambda \alpha : * . \lambda f : \alpha \rightarrow \alpha. \lambda x : \alpha. f x) \\
& \xrightarrow{\beta} (\lambda \beta : * . \lambda f : \beta \rightarrow \beta. \lambda x : \beta. f((\lambda \alpha : * . \lambda f : \alpha \rightarrow \alpha. \lambda x : \alpha. f x) \beta f x)) \\
& \xrightarrow{\beta_{T2}} (\lambda \beta : * . \lambda f : \beta \rightarrow \beta. \lambda x : \beta. f((\lambda f : \beta \rightarrow \beta. \lambda x : \beta. f x) f x)) \\
& \xrightarrow[\beta]{\beta} (\lambda \beta : * . \lambda f : \beta \rightarrow \beta. \lambda x : \beta. f(f x)) \xrightarrow[\alpha_{T2}]{\beta \rightarrow \alpha} \bar{2}
\end{aligned}$$

Problem

(3.13 a) We define addition in Polymorphic Church Numerals as

$$\text{add} \equiv \lambda m, n : \text{nat}. \lambda \alpha : * . \lambda f : \alpha \rightarrow \alpha. \lambda x : \text{nat}. m \alpha f(n \alpha f x)$$

Show that

$$\text{add } \bar{1} \quad \bar{1} \xrightarrow[\beta]{} \bar{2}$$

Solution.

$$\begin{aligned}
& \text{add } \bar{1} \quad \bar{1} \\
& \equiv (\lambda m, n : \text{nat}. \lambda \alpha : * . \lambda f : \alpha \rightarrow \alpha. \lambda x : \alpha. m \alpha f(n \alpha f x)) \bar{1} \quad \bar{1} \\
& \xrightarrow[\beta]{\alpha} \lambda \alpha : * . \lambda f : \alpha \rightarrow \alpha. \lambda x : \alpha. \bar{1} \alpha f(\bar{1} \alpha f x) \\
& \equiv \lambda \alpha : * . \lambda f : \alpha \rightarrow \alpha. \lambda x : \alpha. \bar{1} \alpha f((\lambda \alpha : * . \lambda f : \alpha \rightarrow \alpha. \lambda x : \alpha. f x) \alpha f x) \\
& \xrightarrow[\beta]{\alpha} \lambda \alpha : * . \lambda f : \alpha \rightarrow \alpha. \lambda x : \alpha. \bar{1} \alpha f(f x) \\
& \equiv \lambda \alpha : * . \lambda f : \alpha \rightarrow \alpha. \lambda x : \alpha. (\lambda \alpha : * . \lambda f : \alpha \rightarrow \alpha. \lambda x : \alpha. f x) \alpha f(f x) \\
& \xrightarrow[\beta]{\alpha} \lambda \alpha : * . \lambda f : \alpha \rightarrow \alpha. \lambda x : \alpha. f(f x) \xrightarrow[\alpha]{\alpha} \bar{2}
\end{aligned}$$

Problem

(3.13 b) Find a term mul simulates multiplication on **nat**.

Solution.

$$\text{mul} \equiv \lambda m, n : \text{nat}. \lambda \alpha : * . \lambda f : \alpha \rightarrow \alpha. \lambda x : \alpha. m \alpha(n \alpha f) x$$

Proof. We derive the type first to prove a legal term.

1.	$m, n : \text{nat}$	Bound
2.	$\alpha : *$	Bound
3.	$f : \alpha \rightarrow \alpha$	Bound
4.	$x : \alpha$	Bound
5.	$m \alpha : (\alpha \rightarrow \alpha) \rightarrow \alpha \rightarrow \alpha$	$*, * \text{ T2-App}$
6.	$n \alpha : (\alpha \rightarrow \alpha) \rightarrow \alpha \rightarrow \alpha$	$*, * \text{ T2-App}$
7.	$n \alpha f : \alpha \rightarrow \alpha$	$6, * \text{ T-App}$
8.	$m \alpha(n \alpha f) : \alpha \rightarrow \alpha$	$5, 7 \text{ T-App}$
9.	$m \alpha(n \alpha f) x : \alpha$	$8, * \text{ T-App}$
10.	$\lambda x : \alpha. m \alpha(n \alpha f) x : \alpha \rightarrow \alpha$	9 T-Abst
11.	$\lambda f : \alpha \rightarrow \alpha. \lambda x : \alpha. m \alpha(n \alpha f) x : (\alpha \rightarrow \alpha) \rightarrow \alpha \rightarrow \alpha$	10 T-Abst
12.	$\lambda \alpha : * . \lambda f : \alpha \rightarrow \alpha. \lambda x : \alpha. m \alpha(n \alpha f) x : \text{nat}$	11 T2-Abst
13.		
	$\lambda m, n : \text{nat}. \lambda \alpha : * . \lambda f : \alpha \rightarrow \alpha. \lambda x : \alpha. m \alpha(n \alpha f) x$	
	$: \text{nat} \rightarrow \text{nat} \rightarrow \text{nat}$	12 T-Abst

This proves that the term does indeed produce a natural number from two. Next let's prove that

$$\forall \bar{n}, \bar{m} : \text{nat} \quad \text{mul } \bar{n} \bar{m} = \overline{\bar{n} \times \bar{m}}$$

It could be proven by induction that

$$\forall \bar{a} : \text{nat} \quad \bar{a} \equiv \lambda \alpha : * . \lambda f : \alpha \rightarrow \alpha. \lambda x : \alpha. f^a x$$

$$\begin{aligned} \text{mul } \bar{n} \bar{m} &\equiv (\lambda m, n : \text{nat}. \lambda \alpha : * . \lambda f : \alpha \rightarrow \alpha. \lambda x : \alpha. m \alpha(n \alpha f) x)(\bar{n})(\bar{m}) \\ &\xrightarrow{\beta} \lambda \alpha : * . \lambda f : \alpha \rightarrow \alpha. \lambda x : \alpha. \bar{n} \alpha(\bar{m} \alpha f) x \\ &\xrightarrow{\beta} \lambda \alpha : * . \lambda f : \alpha \rightarrow \alpha. \lambda x : \alpha. \bar{n} \alpha(\lambda u : \alpha. f^m u) x \\ &\xrightarrow{\beta} \lambda \alpha : * . \lambda f : \alpha \rightarrow \alpha. \lambda x : \alpha. (\lambda f : \alpha \rightarrow \alpha. \lambda x : \alpha. f^n x)(\lambda u : \alpha. f^m u) x \\ &\xrightarrow{\beta} \lambda \alpha : * . \lambda f : \alpha \rightarrow \alpha. \lambda x : \alpha. (\lambda u : \alpha. f^m u)^n x \end{aligned}$$

By induction this can be further beta-reduced to

$$\xrightarrow[\beta]{\Rightarrow} \lambda\alpha : * . \lambda f : \alpha \rightarrow \alpha . \lambda x : \alpha . f^{mn} x \equiv \overline{m} \ \overline{n}$$

■

Problem

(3.14) We present the Church-Encoded Boolean:

$$\begin{aligned}\text{bool} &\in \mathbb{T}_2 := \Pi\alpha : * . \alpha \rightarrow \alpha \rightarrow \alpha \\ \text{true} &\equiv \lambda\alpha : * . \lambda x, y : \alpha . x \\ \text{false} &\equiv \lambda\alpha : * . \lambda x, y : \alpha . y\end{aligned}$$

Construct a $\lambda 2$ term neg such that $\text{neg true} \xrightarrow[\beta]{=} \text{false}$ and $\text{neg false} \xrightarrow[\beta]{=} \text{true}$.

Solution.

$$\text{neg} \equiv \lambda b : \text{bool} . \lambda\alpha : * . b \alpha(\text{false } \alpha)(\text{true } \alpha)$$

Neg True.

$$\begin{aligned}\text{neg true} &\equiv (\lambda b : \text{bool} . \lambda\alpha : * . b \alpha(\text{false } \alpha)(\text{true } \alpha)) \text{ true} \\ &\xrightarrow[\beta]{\Rightarrow} \lambda\alpha : * . (\lambda x, y : \alpha . x)(\text{false } \alpha)(\text{true } \alpha) \\ &\xrightarrow[\beta]{\rightarrow} \lambda\alpha : * . \text{false } \alpha \xrightarrow[\eta]{\rightarrow} \text{false}\end{aligned}$$

■

Neg False.

$$\begin{aligned}\text{neg false} &\equiv (\lambda b : \text{bool} . \lambda\alpha : * . b \alpha(\text{false } \alpha)(\text{true } \alpha)) \text{ false} \\ &\xrightarrow[\beta]{\Rightarrow} \lambda\alpha : * . (\lambda x, y : \alpha . y)(\text{false } \alpha)(\text{true } \alpha) \\ &\xrightarrow[\beta]{\rightarrow} \lambda\alpha : * . \text{true } \alpha \xrightarrow[\eta]{\rightarrow} \text{true}\end{aligned}$$

■

Problem

(3.15) Define

$$M \equiv \lambda u, v : \text{bool} . \lambda\beta : * . \lambda x, y : \beta . u \beta(v \beta x y)(v \beta y y)$$

And reduce M true true, M true false, M false true, M false false, and decide which logical operator is represented by M .

Solution.

$$\begin{aligned}
 M \text{ true true} &\equiv (\lambda u, v : \text{bool}. \lambda \beta : * . \lambda x, y : \beta. u \beta(v \beta x y)(v \beta y y)) \text{ true true} \\
 &\xrightarrow[\beta]{\Rightarrow} \lambda \beta : * . \lambda x, y : \beta. \text{true } \beta(\text{true } \beta x y)(\text{true } \beta y y) \\
 &\xrightarrow[\beta]{\Rightarrow} \lambda \beta : * . \lambda x, y : \beta. (\text{true } \beta x y) \\
 &\xrightarrow[\beta]{\Rightarrow} \lambda \beta : * . \lambda x, y : \beta. x \underset{\alpha}{\equiv} \text{true}
 \end{aligned}$$

$$\begin{aligned}
 M \text{ true false} &\equiv (\lambda u, v : \text{bool}. \lambda \beta : * . \lambda x, y : \beta. u \beta(v \beta x y)(v \beta y y)) \text{ true false} \\
 &\xrightarrow[\beta]{\Rightarrow} \lambda \beta : * . \lambda x, y : \beta. \text{true } \beta(\text{false } \beta x y)(\text{false } \beta y y) \\
 &\xrightarrow[\beta]{\Rightarrow} \lambda \beta : * . \lambda x, y : \beta. (\text{false } \beta x y) \\
 &\xrightarrow[\beta]{\Rightarrow} \lambda \beta : * . \lambda x, y : \beta. y \underset{\alpha}{\equiv} \text{false}
 \end{aligned}$$

$$\begin{aligned}
 M \text{ false true} &\equiv (\lambda u, v : \text{bool}. \lambda \beta : * . \lambda x, y : \beta. u \beta(v \beta x y)(v \beta y y)) \text{ false true} \\
 &\xrightarrow[\beta]{\Rightarrow} \lambda \beta : * . \lambda x, y : \beta. \text{false } \beta(\text{true } \beta x y)(\text{true } \beta y y) \\
 &\xrightarrow[\beta]{\Rightarrow} \lambda \beta : * . \lambda x, y : \beta. (\text{true } \beta y y) \\
 &\xrightarrow[\beta]{\Rightarrow} \lambda \beta : * . \lambda x, y : \beta. y \underset{\alpha}{\equiv} \text{false}
 \end{aligned}$$

$$\begin{aligned}
 M \text{ false false} &\equiv (\lambda u, v : \text{bool}. \lambda \beta : * . \lambda x, y : \beta. u \beta(v \beta x y)(v \beta y y)) \text{ false false} \\
 &\xrightarrow[\beta]{\Rightarrow} \lambda \beta : * . \lambda x, y : \beta. \text{false } \beta(\text{false } \beta x y)(\text{false } \beta y y) \\
 &\xrightarrow[\beta]{\Rightarrow} \lambda \beta : * . \lambda x, y : \beta. (\text{false } \beta y y) \\
 &\xrightarrow[\beta]{\Rightarrow} \lambda \beta : * . \lambda x, y : \beta. y \underset{\alpha}{\equiv} \text{false}
 \end{aligned}$$

Therefore M is equivalent to logical AND.

Problem

(3.16) Find $\lambda 2$ term representing the logical OR, XOR, IMP.

Solution.

$$\begin{aligned}
 \text{OR} &\equiv \lambda u, v : \text{bool}. \lambda \beta : * . \lambda x, y : \beta. u \beta x (v \beta x y) \\
 \text{XOR} &\equiv \lambda u, v : \text{bool}. \lambda \beta : * . \lambda x, y : \beta. u \beta(v \beta y x)(v \beta x y) \\
 \text{IMP} &\equiv \lambda u, v : \text{bool}. \lambda \beta : * . \lambda x, y : \beta. u \beta(v \beta x y) x
 \end{aligned}$$

All of them could be checked by finite enumeration over $\text{bool} \times \text{bool}$.

Problem

(3.17) Find $\text{isZero} : \text{nat} \rightarrow \text{bool}$ such that $\forall n : \text{nat}, \text{isZero } n \underset{\beta}{=} \text{false}$ except when $n \equiv \bar{0}$.

Solution.

$$\text{isZero} \equiv \lambda n : \text{nat}. n \text{ bool } (\lambda u : \text{bool}. \text{ false}) \text{ true}$$

Proof.

$$\begin{aligned} \text{isZero } \bar{0} &\equiv (\lambda n : \text{nat}. n \text{ bool } (\lambda u : \text{bool}. \text{ false}) \text{ true}) \bar{0} \\ &\xrightarrow{\beta} (\lambda \alpha : * . \lambda f : \alpha \rightarrow \alpha. \lambda x : \alpha. x) \text{ bool } (\lambda u : \text{bool}. \text{ false}) \text{ true} \\ &\xrightarrow{\beta} (\lambda f : \text{bool} \rightarrow \text{bool}. \lambda x : \text{bool}. x)(\lambda u : \text{bool}. \text{ false}) \text{ true} \\ &\xrightarrow{\beta} \text{ true} \end{aligned}$$

■

By induction it could be proven that any other natural numbers must be applied $\lambda u : \text{bool}. \text{ false}$ to the body, making the result false, except for $\bar{0}$, where the function $f : \alpha \rightarrow \alpha$ never got applied.

Problem

(3.18 a) Define type

$$\text{tree} \equiv \Pi \alpha : *. (\text{bool} \rightarrow \alpha) \rightarrow (\text{bool} \rightarrow \alpha \rightarrow \alpha \rightarrow \alpha) \rightarrow \alpha$$

Then we construct an inhabitant

$$\lambda \alpha. * . \lambda u : \text{bool} \rightarrow \alpha. \lambda v : \text{bool} \rightarrow \alpha \rightarrow \alpha \rightarrow \alpha. M$$

Is a node of a binary tree. Sketch graphs of trees where M is

$$\begin{aligned} &u \text{ false} \\ &v \text{ true } (u \text{ false})(u \text{ true}) \\ &v \text{ true } (u \text{ true})(v \text{ false } (u \text{ true})(u \text{ false})) \end{aligned}$$

Solution. A binary tree is usually defined as this:

```
inductive Tree (a : Type) where
| leaf (value : a) : Tree a
| node (left right : Tree a) : Tree a
```

With two constructors: a leaf or a node. Here α is the type of the payload at each node. There are two constructors: u is the left constructor, taking a `bool` value for the direction of the node. The v term is the node constructor, taking a `bool` as the direction, two α -typed terms as its children.



Problem

(3.18 b) Give a $\lambda 2$ term, which, given input a polymorphic boolean p and two trees s and t , delivers the combined tree with p on top, left subtree s and right subtree t .

Solution.

$$\begin{aligned} \text{leaf} &:= \lambda p : \text{bool}. \lambda s, t : \text{tree}. \\ &\quad \lambda \alpha. \lambda u : \text{bool} \rightarrow \alpha. \lambda v : \text{bool} \rightarrow \alpha \rightarrow \alpha \rightarrow \alpha. \\ &\quad v p (s \alpha u v)(t \alpha u v) \end{aligned}$$

Proof. We suppose

$$\begin{aligned} s &\equiv \lambda \beta : * . \lambda u_s : \text{bool} \rightarrow \beta. \lambda v_s : \text{bool} \rightarrow \beta \rightarrow \beta \rightarrow \beta. S \\ t &\equiv \lambda \gamma : * . \lambda u_t : \text{bool} \rightarrow \gamma. \lambda v_t : \text{bool} \rightarrow \gamma \rightarrow \gamma \rightarrow \gamma. T \end{aligned}$$

We want

$$\text{leaf } p \ s \ t \underset{\beta}{=} \lambda \alpha : * . \lambda u : \text{bool} \rightarrow \alpha. \lambda v : \text{bool} \rightarrow \alpha \rightarrow \alpha \rightarrow \alpha. v p S T$$

For compactness we denote

$$\begin{aligned} \alpha : * \vdash \tau_{\text{mkleaf}} &\equiv \text{bool} \rightarrow \alpha \\ \alpha : * \vdash \tau_{\text{mknnode}} &\equiv \text{bool} \rightarrow \alpha \rightarrow \alpha \rightarrow \alpha \end{aligned}$$

By beta reduction we have

$$\begin{aligned}
\text{leaf } p \ s \ t &\equiv (\lambda p : \text{bool} . \lambda s, t : \text{tree}....) \ p \ s \ t \\
&\xrightarrow[\beta]{} (\lambda \alpha : * . \lambda u . \tau_{\text{mkleaf}} . \lambda v : \tau_{\text{mknoden}} . v \ p \ (s \ \alpha \ u \ v) \ (t \ \alpha \ u \ v)) \\
&\xrightarrow[\beta]{} (\lambda \alpha : * . \lambda u . \tau_{\text{mkleaf}} . \lambda v : \tau_{\text{mknoden}} . v \ p \ (s \ \alpha \ u \ v) \ (t \ \alpha \ u \ v)) \\
&\xrightarrow[\beta]{} \lambda \alpha : * . \lambda u . \tau_{\text{mkleaf}} . \lambda v : \tau_{\text{mknoden}} . v \ p \ S \ T
\end{aligned}$$

■

Problem

(3.19) If $\Gamma \vdash L : \sigma$, then Γ is a valid $\lambda 2$ context.

Solution. The definition of “valid” here would be taken as relative to a judgement as the fact to be able to derive the judgement. Thus, it meant a complete inference path could be made using only statements and judgements derived from the context. Now proof by induction on inference rule that deducted $L : \sigma$.

Case 1 : T-Var. The premise is that Γ is a $\lambda 2$ context. ■

Case 2 : T-App. Therefore $L \equiv M \ N$ for some $M, N \in \Lambda_{T_2}$. Therefore,

$$\Gamma \vdash M : \tau \rightarrow \sigma \quad \Gamma \vdash N : \tau$$

for some $\tau \in T_2$. By the inductive hypothesis on any premise Γ is a valid $\lambda 2$ context. ■

Case 3 : T-Abst. Therefore $L \equiv \lambda x : \alpha . N$ such that

$$\Gamma, x : \alpha \vdash N : \beta$$

for some $\alpha, \beta \in T_2$ such that $\sigma \equiv \alpha \rightarrow \beta$. By the inductive hypothesis, $\Gamma, x : \alpha$. By the recursive definition of $\lambda 2$ contexts, for some valid context Δ , $\forall n \in \text{dom } \Delta$, n could not depend on statement declared after n in the context. Therefore, no statement in Γ could depend on $x : \alpha$. Therefore, Γ is a valid context. ■

Case 4 : T-Form. The premise is that Γ is a valid $\lambda 2$ context. ■

Case 5 : T2-App. Therefore $L \equiv N \ B$ for some $N, B \in V_2$ such that

$$\Gamma \vdash N : \Pi \alpha : * . \sigma \quad \Gamma \vdash B : *$$

By the inductive hypothesis on the any premise Γ is a valid $\lambda 2$ context. ■

Case 6 : T2-Abst. Therefore $L \equiv \lambda \alpha : * . M$ for some $M \in \Lambda_{T_2}$ such that

$$\Gamma, \alpha : * \vdash M : \beta$$

and $\sigma \equiv \Pi\alpha : * . \beta$. By reasoning in *Case 3* no statement in the context could depend on any statement before the latter's declaration. Therefore no statement in Γ could depend on $\alpha : *$, making it a valid context. ■

Problem

(3.20) Prove the free variable lemma for λ 2.

$$\Gamma \vdash L : \sigma \Rightarrow \text{FV } L \subseteq \text{dom } \Gamma$$

Solution. Proof by induction on inference rules that deduced $L : \sigma$. The only rule not considered is T-Form since all terms apparent in \mathbb{T}_2 .

Case 1 : T-Var. Therefore L is the only free variable in L . By the generation lemma $L : \sigma \in \Gamma$, so $\{L\} \subseteq \text{dom } \Gamma$ ■

Case 2 : T-App. Therefore by the generation lemma $L \equiv M N$ for some $M, N \in \Lambda_{\mathbb{T}_2}$ such that

$$\Gamma \vdash M : \tau \rightarrow \sigma \quad \Gamma \vdash N : \tau$$

For some type τ . By the inductive hypothesis $\text{FV } M \subseteq \Gamma$ and $\text{FV } N \subseteq \text{dom } \Gamma$. Therefore $\text{FV } L = (\text{FV } M) \cup (\text{FV } N) \subseteq \text{dom } \Gamma$. ■

Case 3 : T-Abst. Therefore by the generation lemma $L \equiv \lambda x : \alpha . M$ for some $M \in \Lambda_{\mathbb{T}_2}$ such that

$$\Gamma, x : \alpha \vdash M : \beta$$

and $\sigma \equiv \alpha \rightarrow \beta$. By the inductive hypothesis $\text{FV } M \subseteq \text{dom } \Gamma \cup \{x\}$. Therefore

$$\text{FV } L = \text{FV } M \setminus \{x\} \subseteq (\text{dom } \Gamma \cup \{x\}) \setminus \{x\} = \text{dom } \Gamma$$

Case 4 : T2-App. Therefore $L \equiv B$ for some $N, B \in \mathbb{V}_2$ such that

$$\Gamma \vdash N : \Pi\alpha : * . \sigma \quad \Gamma \vdash B : *$$

By the inductive hypothesis $\text{FV } N \subseteq \text{dom } \Gamma$. Since $B \in \mathbb{T}_2$ then $\text{FV } B = \emptyset$. Therefore $\text{FV } L = \text{FV } N \subseteq \text{dom } \Gamma$. ■

Case 5 : T2-Abst. Therefore $L \equiv \lambda\alpha : * . M$ for some $M \in \Lambda_{\mathbb{T}_2}$ such that

$$\Gamma, \alpha : * \vdash M : \beta$$

and $\sigma \equiv \Pi\alpha : * . \beta$. Because $\alpha \in \mathbb{T}_2$ so $\alpha \notin \text{FV } M$ and $\text{FV } L = \text{FV } M$, thus $\text{FV } M = \text{FV } L \subseteq \Gamma$. ■

Problem

(3.21) Give a recursive definition for $\text{FTV} : \mathbb{T}_2 \cup \Lambda_{\mathbb{T}2} \rightarrow \mathbb{V}_2$

Solution. Here $\alpha \in \mathbb{V}_2$, $A, B \in \mathbb{T}_2$, $x \in V$ and $M \in \Lambda_{\mathbb{T}2}$.

Form	Value
$\text{FTV } \alpha$	$\{\alpha\}$
$\text{FTV } x$	\emptyset
$\text{FTV } (A \rightarrow B)$	$\text{FTV}(A) \cup \text{FTV}(B)$
$\text{FTV } (\Pi \alpha : * . A)$	$\text{FTV}(A) \setminus \{\alpha\}$
$\text{FTV } (M N)$	$\text{FTV}(M) \cup \text{FTV}(N)$
$\text{FTV } (\lambda x : A . M)$	$\text{FTV}(A) \cup \text{FTV}(M)$
$\text{FTV } (M B)$	$\text{FV}(M) \cup \text{FTV}(B)$
$\text{FTV } (\lambda \alpha : * . M)$	$\text{FTV}(M) \setminus \{\alpha\}$

—

Completed Dec 16 10:11 pm.