2.8 型別檢查於$\lambda \to$

$x : \alpha \to \alpha,\ y : (\alpha \to \alpha) \to \beta \vdash (\lambda z : \beta.\lambda u : \gamma.z)(y\ x) : \gamma \to \beta$

(a)| $^*x : \alpha \to \alpha^*$

(b)|| $^*y : (\alpha \to \alpha) \to \beta^*$

|| ...

(n)||$(\lambda z : \beta.\lambda u : \gamma.z)(y\ x) : \gamma \to \beta$

appl 拆解

(a) | $^*x : \alpha \to \alpha^*$

(b) || $^*y : (\alpha \to \alpha) \to \beta^*$

(m1)|| $\lambda z : \beta.\lambda u : \gamma.z :?_1$

(m2)|| $y\ x :?_2$

(n) ||$(\lambda z : \beta.\lambda u : \gamma.z)(y\ x) : \gamma \to \beta$

繼續反推

(a) | $^*x : \alpha \to \alpha^*$

(b) || $^*y : (\alpha \to \alpha) \to \beta^*$

||| $^*z : \beta^*$

|||| $^*u : \gamma^*$

|||| $\lambda u.z : \gamma \to \beta$

(m1)|| $\lambda z : \beta.\lambda u : \gamma.z : \beta \to (\gamma \to \beta)$

(m2)|| $y\ x : \beta$

(n) ||$(\lambda z : \beta.\lambda u : \gamma.z)(y\ x) : \gamma \to \beta$

就可以得到一串推論過程。

但是，$(\lambda z : \beta\lambda u : \gamma.z)(y\ x)$

用 beta-reduction 處理可得：

$(\lambda u : \gamma.(y\ x))$

型別是：$\gamma \to \beta$，也和剛才的推論出來的(n)的型別相同。

2.9 找 term 於$\lambda \to$

$a \in T \Rightarrow$ a 是 T 的 inhabitant（居民）

type 是邏輯表達式(proposition)

term 構築型別的證明

證明有一個 term，滿足

$? : A \to B \to A$

證明法：逆著回推，箭頭→型別表示 abst 抽象。

|*$x : A$*

||*$y : B$*

|| $x : A$

| $\lambda y.x : B \to A$

$\lambda x.\lambda y.x : A \to B \to A$

PAT-interpretation

- proposition as types 命題即型別
- proof as terms 證明即 term

2.10 $\lambda \to$的一般性質

dom、↾投影(projection)、⊆次上下文(subcontext)

定義

1. 上下文：$\Gamma \equiv x_1 : \sigma_1, ..., x_n : \sigma_n \Rightarrow$即上下文的定義域(domain)是$(x_1, x_2, ..., x_n)$
2. 若所有的宣告變數一樣，且排序一致，則$\Gamma' \subseteq \Gamma$，即Γ'是Γ的次上下文
3. Γ'是Γ上下文的 permutation(排列)，若Γ'的變數宣告於Γ有，且反之亦然。
4. 若Γ是上下文，Φ是變數集合，Γ在Φ上的投影(projection)，即$\Gamma \upharpoonright \Phi$是$\Gamma'$，滿足 $dom(\Gamma') = dom(\Gamma) \cap \Phi.$

定理 2.10.3 自由變數公理

$\Gamma \vdash L : \sigma \Rightarrow FV(L) \subseteq dom(\Gamma)$

任何自由變數 x 在 L 出現，則在Γ有$x : \sigma$這個宣告。

任何變數的型別不會搞混。

Lemma 2.10.5

1. thinning(厚化): 令Γ'和Γ''是滿足$\Gamma' \subseteq \Gamma''$的上下文，$\Gamma' \vdash M : \sigma \Rightarrow \Gamma'' \vdash M : \sigma$
2. condensing(濃縮): $\Gamma \vdash M : \sigma \Rightarrow \Gamma \upharpoonright FV(M) \vdash M : \sigma$. 意思是可以移除$x : \sigma$，若是 x 不是 M 的自由變數，所以保持這些和 M 有關的宣告。
3. Permutation(排列): 若$\Gamma \vdash M : \sigma$且 Γ'是Γ的 permutation，則Γ'是上下文，滿足$\Gamma' \vdash M : \sigma$. 指 context 的變數順序不影響推演能力（變數宣告互相獨立）

一個人可以加減 junk 變數到上下文，而不影響衍推能力(derivibility)。

junk 包含於 M 中不自由的變數（typing 的過程不會用到）

Lemma 2.10.7 - Generation Lemma

1. $\Gamma \vdash x : \sigma \Rightarrow x : \sigma \in \Gamma$
2. if $\Gamma \vdash M\ N : \tau$, then $\exists \sigma,\ s.t.\ \Gamma \vdash M : \sigma \to \tau$ and $\Gamma \vdash N : \sigma$
3. $\Gamma \vdash \lambda x : \sigma.M : \rho \Rightarrow \exists \tau\ s.t.\ \Gamma, x : \sigma \vdash M : \tau \wedge \rho \equiv \sigma \to \tau$

Lemma 2.10.8 subterm Lemma

若 M 是 legal⇒任何 M 的 subterm 也 legal。

if $\exists\Gamma_1,\ \sigma_1\ s.t.\ \Gamma_1 \vdash M : \sigma_1$ 且 L 為 M 的 subterm⇒ $\exists\Gamma_2, \sigma_2\ s.t.\ \Gamma_2 \vdash L : \sigma_2$

在一個上下文 F，一個 term 至多有一個 type，也就是：

$\Gamma \vdash M : \sigma \wedge \Gamma \vdash M : \tau \Rightarrow \gamma \equiv \tau$

我們可以分出 4 種求型別或 term 的類型：

1. well-typeness

? ⊢ term: ?

1a. type assignment

context ⊢ term : ?

2. type checking

context $\overset{?}{\vdash}$ term: type

3. term finding

context ⊢? : type

以上 4 種皆為 decidible（可決定的）。

Sec.2.11 歸約與 $\lambda \rightarrow$

Ch1.6 有 beta-reduction（β歸約）的公式：

1a) x[x:=N]≡N

1b) y[x:=N]≡y (if x≠y)

2) (P Q)[x:=N]≡(P[x:=N])(Q[x:=N])

3) (λy.P)[x:=N]≡($\lambda z.P^{y\to z}$[x:=N])若 $\lambda z.P^{y\to z}$是 λy.P 的 variation，且 z∉FV(N)

為了型別系統的改變，所以我們把 3)改為：

3)($\boldsymbol{\lambda y : \sigma}$.P)[x:=N]≡($\boldsymbol{\lambda z : \sigma.P^{y\to z}}$[x:=N]) 若$\lambda z : \sigma.P^{y\to z}$是$\lambda y : \sigma.P$的 alpha variant，且滿足 $z \notin FV(N)$

Lemma 2.11.1 替換 Lemma

假設$\Gamma', x : \sigma, \Gamma'' \vdash M : \tau$且$\Gamma' \vdash N : \sigma$

則$\Gamma', \Gamma'' \vdash M[x := N] : \tau$

Def 定義 One-step reduction$\underset{\beta}{\rightarrow}$ for $\Lambda_{\mathbb{T}}$

1. (Basis)

$(\lambda x : \sigma.M)N \underset{\beta}{\rightarrow} M[x := N]$

(compatibility)見定義 1.8.1。

Theorem 2.11.3

Church-Rosser Theorem (Theorem 1.9.8)在 $\lambda \rightarrow$也成立。

Corollary 推論 2.11.4

假設有 $M \underset{\beta}{=} N$則$\exists L\ s.t. M \underset{\beta}{\twoheadrightarrow} L$且$N \underset{\beta}{\twoheadrightarrow} L$

Lemma 2.11.5 Subject Reduction:

$\Gamma \vdash L : \rho$且$L \twoheadrightarrow L'$ 則$\Gamma \vdash L' : \rho$

beta 歸約不改變 typability 和其型別。

Reduction 是一種形式的 calculation。

$x : \alpha \to \alpha, y : (\alpha \to \alpha) \to \beta \vdash \lambda u : \gamma.y\ x : \gamma \to \beta$

任何計算皆為 finite（有限的）！

Theorem 2.11.6

strong normalization theorem (Termination theorem)

任何合法的 M 是 strongly normalizing（強正規化的）。

strong normalizing 保證運算有結果

因為$\lambda \to$的能力有限，有足夠能力的運算系統會有無法終止的情況發生。雖然能終止的系統也可能要很久（雖然是有限久）才能算出結果。

2.12 結果

untyped lambda 運算的缺點會消失：

- 沒有自我代入(self-application)
- 保證有 beta-normal forms
- 不是所有的 legal lambda term 都有不動點

2.13 結論

positive points of untyped lambda calculation 擴展到簡單型別 lambda 運算 system$\lambda \to$太弱，不能封裝所有運算的函數，不能作為數學的形式化。

- call-by-value

$(\lambda x : \sigma.M)N$先求出 N 再代入

- call-by-name

$(\lambda x : \sigma.M)N$直接將 N 代入 M 中的各個 x（$M[x := N]$）