Cartesian Closed Category and Simply Typed λ-Calculus

Prepared by CanaanZhou ;)

April 20

1 Cartesian Closed Category

Recall that a cartesian closed category C, or a ccc, is a category with finite products (including 0-ary product, the terminal object) and for each $c \in \mathsf{C}$, the functor $- \times c : \mathsf{C} \to \mathsf{C}$ has a right adjoint $(-)^c$.

Intuitively, d^c may be thought of as the *function space* from c to d.

For our purpose, we assume that every ccc C has a *choice* of d^c for any pair of objects c, d, and a *choice* of $\prod_{i \in I} c_i$ for each finitely-indexed family of object in C.

The prototypical example is Set.

Example 1.1

Set *is a ccc. For every pair of sets* X, Y*, the set* Y^X *is simply* $\mathsf{Set}(X, Y)$*. We have a natural isomorphism:*

$$\mathsf{Set}(X \times Y \to Z) \cong \mathsf{Set}(X, Z^Y).$$

The idea is that for each binary function $f : X \times Y \to Z$*, given an element* $x \in X$*, one get a unary function* $f(x, -)$ *by inserting* x *to the first input of* f*. This is called* Currying.

Aside from this example, one way to justify that d^c acts like the *function space* is by looking at the global elements. Suppose we have a $x : 1 \to d^c$, by ccc, x uniquely corresponds to a $1 \times c \to d$, which also unique corresponds to a $c \to d$.

Exercise 1.2

Show that for any object c*,* $1 \times c \cong c$*.*

We now introduce the idea of *generalized* elements.

Definition 1.3 (Generalized Elements)
A generalized element x *of an object* c *over another object* u *is a morphism* $x : u \to c$*.*

You can think of a generalized element $x : u \to c$ as a *term* of c with a *free variable* of type u.

The point of generalized elements is that:

- Just like global elements, every morphism $f : c \to d$ now becomes a function between sets of generalized elements $f_* : \mathsf{C}(u, c) \to \mathsf{C}(u, d)$.
- Unlike global elements, the choice of u is usually arbitrary. By Yoneda lemma, to study any object c, it suffices to study its generalized elements over an arbitrary u.

2 Simply typed λ-calculus

WARNING! I basically wrote this part all by myself. Please be extra careful when reading!

2.1 The Syntax

We give a quick informal definition of the syntax of simply typed λ-calculus.

A *type world* T is a set with a binary operation $(\sigma, \tau) \mapsto (\sigma \to \tau) : \mathsf{T} \times \mathsf{T} \to \mathsf{T}$, where each element $\sigma \in \mathsf{T}$ is called a *type*. The type $\sigma \to \tau$ is the *function type* from σ to τ.

For each type τ we have countably infinite variables of type τ.

An *environment* is a finite list $\Gamma = (x_0^{\sigma_0}, \cdots, x_{r-1}^{\sigma_{r-1}})$, where for each i, $x_i^{\sigma_i}$ is a variable of type σ_i.[1]

For two environment Γ and Δ, the environment Γ, Δ is Γ and Δ concatenated.

We have a set C of *constants*, where each $c^\sigma \in C$ has an assigned type σ.

Now we define *well-typed terms*. They are generated by the following rules.

Constants	Variables	Permutation	
$\Gamma \vdash c^\sigma : \sigma$	$\Gamma, x^\sigma, \Delta \vdash x^\sigma : \sigma$	$\dfrac{\Gamma \vdash M : \tau}{\overline{\Gamma} \vdash M : \tau}$	$[\overline{\Gamma}$ is a permutation of $\Gamma]$

[1] In most literatures Γ are structured differently. There are mainly for choices: list, non-repetition list, unordered list (bag), unordered non-repetition list (set). After contemplating for a while, I think list is the best choice. I encourage you to form your own opinion on this issue and feel free to disagree.

λ-ABSTRACTION

$$\frac{\Gamma, x^\sigma, \Delta \vdash M : \tau}{\Gamma, \Delta \vdash \lambda x^\sigma . M : \sigma \to \tau}$$

λ-APPLICATION

$$\frac{\Gamma \vdash M : \sigma \to \tau \qquad \Gamma \vdash N : \sigma}{\Gamma \vdash MN : \tau}$$

For any sequent $\Gamma \vdash M : \tau$, each $x^\sigma \in \Gamma$ is called a *free variable* in M. The λ-abstraction rule above *bounds* the free variable x^σ.

WEAKENING

$$\frac{\Gamma \vdash M : \tau}{\Gamma, \Delta \vdash M : \tau}[\Delta \text{ is any environment}]$$

Lemma 2.1 (Weakening)

Weakening is admissible.

Proof. We perform induction on the deduction of $\Gamma \vdash M : \tau$.

- If $M : \tau$ is a variable y^τ and the sequent is $\Gamma, y^\tau, \Gamma' \vdash y^\tau : \tau$, then $\Gamma, y^\tau, \Gamma', \Delta \vdash y^\tau : \tau$ is also valid. Contant is similar.
- Suppose $\overline{\Gamma}$ is a permutation of Γ and we have deduced $\overline{\Gamma} \vdash M : \tau$ from $\Gamma \vdash M : \tau$ by the rule of permutation, then by IH, we have a deduction of $\Gamma, \Delta \vdash M : \tau$. But $\overline{\Gamma}, \Delta$ is also a permutation of Γ, Δ, thus by permutation, we can deduce $\overline{\Gamma}, \Delta \vdash M : \tau$.
- Suppose the last step is λ-abstraction, from $\Gamma, x^\sigma, \Gamma' \vdash M : \tau$ to $\Gamma, \Gamma' \vdash \lambda x^\sigma . M : \sigma \to \tau$, then adding Δ to each side of the deduction changes nothing.
- λ-application is similar.

□

SUBSTITUTION

$$\frac{\Gamma, x^\sigma, \Gamma' \vdash M : \tau \qquad \Gamma, \Gamma' \vdash N : \sigma}{\Gamma, \Gamma' \vdash M[x^\sigma \mapsto N] : \tau}$$

where $M[x^\sigma \mapsto N]$ means the term M, but every occurence of x^σ is substituted by N.

Lemma 2.2 (Substitution)

Substitution is admissible.

The proof of this lemma is also a *inductive definition* of $M[x^\sigma \mapsto N]$.

Proof. Again, we perform induction on the deduction of the premises.

- Suppose $M : \tau$ is a variable $y^\tau : \tau$ distinct from x^σ and the first premise is given by variable rule, then $M[x^\sigma \mapsto N]$ is just y^τ. Constant is similar.

- Suppose $M : \tau$ is x^σ, then $M[x^\sigma \mapsto N]$ is defined to be N, and by hypothesis we have a deduction of $\Gamma, \Gamma' \vdash N : \sigma$.

- Permutation case is easy.

- Suppose the left premise is:

$$\frac{\Sigma, y^\mu, \Sigma' \vdash P : \tau}{\Sigma, \Sigma' \vdash \lambda y^\mu . P : \mu \to \tau}$$

then $x^\sigma \in \Sigma$ or $x^\sigma \in \Sigma'$, doesn't really matter. Suppose $x^\sigma \in \Sigma$, say $\Sigma = \Sigma_0, x^\sigma, \Sigma_1$.

$$\frac{\dfrac{\Sigma_0, x^\sigma, \Sigma_1, y^\mu, \Sigma' \vdash P : \tau \qquad \Sigma_0, \Sigma_1, y^\mu, \Sigma' \vdash N : \sigma}{\Sigma_0, \Sigma_1, y^\mu, \Sigma' \vdash P[x^\sigma \mapsto N] : \tau}\text{[Induction Hypothesis]}}{\Sigma_0, \Sigma_1, \Sigma' \vdash \lambda y^\mu . P[x^\sigma \mapsto N] : \mu \to \tau}$$

- λ-application case is also easy, where for $M : \nu \to \tau$ and $P : \nu$, $MP[x^\sigma \mapsto N]$ is defined to be $M[x^\sigma \mapsto N]P[x^\sigma \mapsto N]$.

□

In particular we have:

$$\frac{\dfrac{\Gamma, x^\sigma, \Gamma' \vdash M : \tau}{\Gamma, x^\sigma, y^\sigma, \Gamma' \vdash M : \tau} \qquad \Gamma, y^\sigma, \Gamma' \vdash y^\sigma : \sigma}{\Gamma, y^\sigma, \Gamma' \vdash M[x^\sigma \mapsto y^\sigma] : \tau}$$

where x^σ and y^σ are two distinct variables of the same type σ.

In the above deduction we can perform λ-abstraction on both the premise and the conclusion, having two sequents:

$$\Gamma, \Gamma' \vdash \lambda x^\sigma . M : \sigma \to \tau,$$

$$\Gamma, \Gamma' \vdash \lambda y^\sigma . M[x^\sigma \mapsto y^\sigma] : \sigma \to \tau.$$

Sequents related in this way are *α-equivalent*, denoted as $\lambda x^\sigma.M \equiv_\alpha \lambda y^\sigma.M[x^\sigma \mapsto y^\sigma]$. Note that even though we omit the environment and the type, they should be clear from context. In literatures, some congruence conditions are often imposed on α-equivalence, for example $M \equiv_\alpha N \implies PM \equiv_\alpha PN$. Here we choose to impose these conditions after definition all three kinds of "raw equivalence": α, β and η-equivalence.

Here's something important proposed by Ye Lingyuan. My definition of substitution is different from most literatures. Sometimes free/bounded variables might be confusing. Suppose our type world T has only one type σ, and $\sigma \to \sigma = \sigma$.[2] Consider the following deduction.

$$\dfrac{\dfrac{\dfrac{\dfrac{x, y \vdash x \qquad x, y \vdash y}{x, y \vdash xy}}{x \vdash \lambda y.xy}}{\underline{y}, x \vdash \lambda y.xy} \qquad \underline{y} \vdash \underline{y}}{\underline{y} \vdash \lambda y.\underline{y}y}$$

If we only look at the last term $y \vdash \lambda y.yy$, we have no idea what's going on. That's the thing about our system: instead of dealing with terms, we deal with *deductions*. The full deduction shows that $y \vdash \lambda y.yy$ in fact has two *different* ys. The underlined one is the one substituting x, while the normal one is the one λ-abstracted in the second step of the deduction. Since everything is defined by induction on deductions, there's no ambiguity.

Normally people only deal with *terms*, or more precisely, *α-equivalence classes of terms*. When doing substitution, they require everything to be properly α-converted so that there won't be any variables clashing. For example, when substituting y for x in $\lambda y.xy$, we have to α-convert $\lambda y.xy$ to, for example, $\lambda z.xz$. After substitution we get $\lambda z.yz$. I think my system is more elegant. (Feel free to disagree but come on!) No matter what you prefer, please always avoid stuff like directly substituting y for x in $\lambda y.xy$.

Now, given two sequents $\Gamma, x^\sigma, \Gamma' \vdash M : \tau$, $\Gamma, \Gamma' \vdash N : \sigma$, we have the following deductions:

$$\dfrac{\dfrac{\Gamma, x^\sigma, \Gamma' \vdash M : \tau}{\Gamma, \Gamma' \vdash \lambda x^\sigma.M : \sigma \to \tau} \qquad \Gamma, \Gamma' \vdash N : \sigma}{\Gamma, \Gamma' \vdash (\lambda x^\sigma.M)N : \tau}$$

[2]This is exactly *untyped* λ-calculus.

And by substitution we can also deduce $\Gamma, \Gamma' \vdash M[x^\sigma \mapsto N] : \tau$ directly. We say the latter sequent is a one-step *β-reduction* of the former. If there's a one-step β-reduction chain (of finite length) from $\Gamma \vdash M : \tau$ to $\Gamma \vdash N : \tau$, we say the latter is a β-reduction of the former, denoted as $M \twoheadrightarrow_\beta N$.

In fact, β-reduction is the central concept in λ-calculus. This string-rewriting process captures the concept of *computation*. We have the following important theorem:

Theorem 2.3 (Church-Rosser Theorem)
Suppose we have terms M_1, M_2, M_3 of the same type τ under the same environment Γ, and $M_1 \twoheadrightarrow_\beta M_2$, $M_1 \twoheadrightarrow M_3$. Then there is a $\Gamma \vdash M_4 : \tau$ such that $M_2 \twoheadrightarrow M_4 : \tau$ and $M_3 \twoheadrightarrow M_4 : \tau$.

The equivalence relation generated by $\twoheadrightarrow_\beta$ is called *β-equivalence*, denoted as $\equiv_\beta$.

$$\eta\text{-CONVERSION}\quad \frac{\Gamma \vdash M : \sigma \to \tau}{\Gamma \vdash \lambda x^\sigma . M x^\sigma : \sigma \to \tau}$$

Lemma 2.4 (η-conversion)
η-conversion is admissible.

Proof.

$$\frac{\dfrac{\Gamma \vdash M : \sigma \to \tau}{\Gamma, x^\sigma \vdash M : \sigma \to \tau} \qquad \Gamma, x^\sigma \vdash x^\sigma : \sigma}{\dfrac{\Gamma, x^\sigma \vdash M x^\sigma : \tau}{\Gamma \vdash \lambda x^\sigma . M x^\sigma : \sigma \to \tau}}$$

□

We say $\Gamma \vdash \lambda x^\sigma . M x^\sigma : \sigma \to \tau$ is the η-conversion of $\Gamma \vdash M : \sigma \to \tau$. The equivalence relation generated by it is denoted as $\equiv_\eta$.

The equivalence relation generated by $\equiv_\alpha, \equiv_\beta, \equiv_\eta$ altogether is denoted as $\equiv$. We further impose that:

- $M \equiv N \implies PM = PN$,
- $M \equiv N \implies \lambda x^\sigma . M = \lambda x^\sigma . N$,
- $M \equiv N \implies MP = NP$.

If $M \equiv N$ we say they are *equivalent*.

2.2 The Semantics

Now we interpret this formal language in a ccc.

Fix a ccc C. An interpretation of T to C is a function from T to the class of objects of C. So each type τ is assigned to an object of C. For simplicity let's call it τ. $\sigma \to \tau$ is interpreted as τ^σ.

Each constant $c^\sigma \in \sigma$ is interpreted as a global element $c^\sigma : 1 \to \sigma \in \mathsf{C}$.

An environment $\Gamma = (x_1^{\sigma_1}, \cdots, x_n^{\sigma_n})$ is interpreted as the product $\Gamma = \prod_{i=1}^n \sigma_i$. We now inductively define the interpretation of sequent $\Gamma \vdash M : \tau$ as a morphism $M : \Gamma \to \tau$.

- $\Gamma \vdash c^\sigma : \sigma$ is interpreted as the morphism $\Gamma \xrightarrow{!} 1 \xrightarrow{c^\sigma} \sigma$.
- $\Gamma, x^\sigma, \Gamma' \vdash x^\sigma : \sigma$ is interpreted as the projection morphism $\pi_\sigma : \Gamma \times \sigma \times \Gamma' \to \sigma$.
- Suppose $\Gamma \vdash M : \tau$ is interpreted as $M : \Gamma \to \tau$, then for $\overline{\Gamma}$ a permutation of Γ, we have a permutation map $\pi : \overline{\Gamma} \to \Gamma$. $\overline{\Gamma} \vdash M : \tau$ is interpreted as $\overline{\Gamma} \xrightarrow{\pi} \Gamma \xrightarrow{M} \tau$.
- Suppose $\Gamma, x^\sigma, \Gamma' \vdash M : \tau$ is interpreted as $M : \Gamma \times \sigma \times \Gamma' \to \tau$, then $\Gamma, \Gamma' \vdash \lambda x^\sigma.M : \sigma \to \tau$ is interpreted as the transpose of M, $M^\flat : \Gamma \times \Gamma' \to \tau^\sigma$.
- Suppose $\Gamma \vdash M : \sigma \to \tau$ and $\Gamma \vdash N : \sigma$ are interpreted as $M : \Gamma \to \tau^\sigma$ and $N : \Gamma \to \sigma$, then $\Gamma \vdash MN : \tau$ is interpreted as $\Gamma \xrightarrow{\langle M,N \rangle} \tau^\sigma \times \sigma \xrightarrow{\mathrm{ev}_\tau} \tau$, where ev is the counit of $- \times \sigma \vdash (-)^\sigma$.

A bit of calculations tells us:

- Suppose $\Gamma \vdash M : \tau$ is interpreted as $M : \Gamma \to \tau$, then $\Gamma, \Delta \vdash M : \tau$ is interpreted as $\Gamma \times \Delta \xrightarrow{\pi_\Gamma} \Gamma \xrightarrow{M} \tau$.
- Suppose $\Gamma, x^\sigma, \Gamma' \vdash M : \tau$ and $\Gamma, \Gamma' \vdash N : \sigma$ are interpreted as $M : \Gamma \times \sigma \times \Gamma' \to \tau$ and $N : \Gamma \times \Gamma' \to \sigma$, then $\Gamma, \Gamma' \vdash M[x^\sigma \mapsto N] : \tau$ is interpreted as $\Gamma \times \Gamma' \xrightarrow{\langle 1_\Gamma, N, 1_{\Gamma'} \rangle} \Gamma \times \sigma \times \Gamma \xrightarrow{M} \tau$.
- Suppose $\Gamma \vdash M : \sigma \to \tau$ is interpreted as $M : \Gamma \to \tau^\sigma$, then $\Gamma \vdash \lambda x^\sigma.Mx^\sigma : \sigma \to \tau$ is interpreted as M itself, which is the transposition of $\Gamma \times \sigma \xrightarrow{M \times 1_\sigma} \tau^\sigma \times \sigma \xrightarrow{\mathrm{ev}_\sigma} \tau$.

$$\begin{tikzcd} \Gamma \times \sigma \arrow[d, "M \times 1_\sigma"'] & \\ \tau^\sigma \times \sigma \arrow[r, "\mathrm{ev}_\sigma"'] & \tau \end{tikzcd} \qquad \begin{tikzcd} \Gamma \arrow[d, "M"'] & \\ \tau^\sigma \arrow[r, "1_{\tau^\sigma}"'] & \tau^\sigma \end{tikzcd}$$

Now fix two sequents $\Gamma \vdash M : \tau, \Gamma \vdash N : \tau$.

Lemma 2.5

If $M \equiv_\alpha N$, then their interpretations are equal.

Proof. Suppose $\Gamma, x^\sigma, \Gamma' \vdash M : \tau$ is interpreted as $\Gamma \times \sigma \times \Gamma' \xrightarrow{M} \tau$, then $\Gamma, y^\sigma, \Gamma' \vdash M[x^\sigma \mapsto y^\sigma] : \tau$ is interpreted as the same exact thing. Essentially what happened is

$$(A \times B \xrightarrow{\langle \pi_A, 1_{A\times B}\rangle} A \times A \times B \xrightarrow{\pi_{L,R}} A \times B) = 1_{A \times B}$$

where $\pi_{L,R}$ means projection on the left and the right components. This can be shown by a straightforward diagram chasing. Then of course, their λ-abstractions are interpreted as transpositions of the corresponding morphisms, which are equal. □

Lemma 2.6

If $M \to_\beta N$, then their interpretations are equal, where $\to_\beta$ means one-step β-reduction.

Proof. Given two sequents $\Gamma, x^\sigma, \Gamma' \vdash M : \tau$ and $\Gamma, \Gamma' \vdash N : \sigma$.

- $\Gamma, \Gamma' \vdash (\lambda x^\sigma.M)N : \tau$ is interpreted as $\Gamma \times \Gamma' \xrightarrow{\langle M^\flat, N\rangle} \tau^\sigma \times \sigma \xrightarrow{\mathrm{ev}_\sigma} \tau$.
- $\Gamma, \Gamma' \vdash M[x^\sigma \mapsto N] : \tau$ is interpreted as $\Gamma \times \Gamma' \xrightarrow{\langle 1_\Gamma, N, 1_{\Gamma'}\rangle} \Gamma \times \sigma \times \Gamma' \xrightarrow{M} \tau$.

To see that they're equal, look at the following diagram.

$$\begin{tikzcd}
\Gamma \times \Gamma' \arrow[r, "\langle \eta, N\rangle"] \arrow[d, "\langle 1_\Gamma, N, 1_{\Gamma'}\rangle"'] & (\Gamma \times \sigma \times \Gamma')^\sigma \times \sigma \arrow[r, "M^\sigma \times 1_\sigma"] \arrow[d, "\mathrm{ev}_{\Gamma\times\Gamma'}"] & \tau^\sigma \times \sigma \arrow[d, "\mathrm{ev}_\sigma"] \\
\Gamma \times \sigma \times \Gamma' \arrow[r, "1"'] \arrow[ur, "\eta"] & \Gamma \times \sigma \times \Gamma' \arrow[r, "M"'] & \tau
\end{tikzcd}$$

□

Thus if $M \equiv_\beta N$, their interpretations are equal.

We have shown that it's the same for $\equiv_\eta$. The following theorem is the final reward for all these hardwork.

Theorem 2.7 (Soundness of the Calculus)

If $M \equiv N$, then their interpretations are equal.

For example, let's try to define the *composition* morphism $X^Y \times Y^Z \to X^Z$. If we try to do it categorically, we need to define its transposition $Z \times X^Y \times Y^Z \to X$. We may define it as:

$$Z \times X^Y \times Y^Z \xrightarrow{\mathrm{ev}_Y} X^Y \times Y \xrightarrow{\mathrm{ev}_X} X.$$

But if we use our beautiful λ-calculus, this is simply

$$f : Y \to X, g : Z \to Y \vdash \lambda z.f(g(z)) : Z \to X.$$

If you think this is not a big simplification, try proving the composition morphism is associative. I don't even want to prove it categorically. But using our formal language, this is just easy calculus. Fix the environment $\Gamma = (f : Y \to X, g : Z \to Y, h : W \to Z)$, then we have:

$$\lambda w.(\lambda z.fgz)(hw) \to_\beta \lambda w.gfhw.$$

Our formal language has absorbed the calculus rules of ccc, so there's almost nothing to prove!