

# Dependent Type Theory

About Extensional Type Systems

Hojune Lee

School of Computing, KAIST

# Contents

<b>1</b>	<b>Context and Substitutions</b>	<b>3</b>
1.1	Intuitions . . . . .	3
1.2	Contexts Judgements . . . . .	5
1.3	Substitution Judgements . . . . .	5
<b>2</b>	<b>Internalizing Judgemental Structure : <math>\Pi, \Sigma, \text{Eq}, \text{Unit}</math></b>	<b>13</b>
2.1	Basic Intuitions . . . . .	13
2.2	Dependent Sums . . . . .	14

# Chapter 1

## Context and Substitutions

### 1.1 Intuitions

In simple type theory, every thing was quite clear. This is because that, if we once fix the rules for types in simple type theory, then all possible types are determined immediately by inductive way. We can informally think this as ‘propositional logic’-like type construction. However, many things are different in dependent type system. Let’s think dependent type system as ‘first order logic’-like type construction. Imagine the formulas in first order logic as types in dependent type theory. I hope that this is proper intuitive thinking for the main differences between two type systems.

Once we imagine FOL formula-style type, it means that each type can contain variables and constants as it’s representation. It means that for each variables which are appeared in type representation, we must know what are the types of each variables. That is Context’s role.

Now, we need to construct the intuition for contexts and substitutions. Contexts are ‘worlds’ of terms and types. And substitutions are ‘traffic way’ between each worlds( contexts ). Then naturally, following questions are arising.

#### Questions

1. Is this world ( Context ) well-formed ?
2. In this world ( Context )  $\Gamma$ , what terms of type  $A$  are well-formed ?
3. In this world ( Context )  $\Gamma$ , what types are well-formed ?
4. Between 2 worlds ( Context )  $\Delta$  and  $\Gamma$ , is traffic way ( substitution )  $\gamma$  well-formed ?
5. In this world ( Context ), which terms/types/substitutions are equivalent?

Actually, this is all about ‘judgement rules’ referred in Notation 2.3.1. of textbook. Formally, we can write format for above judgements.

### Formal Representations for Judgements

1.  $\vdash \Gamma \text{ cx}$
2.  $\Gamma \vdash a : A$
3.  $\Gamma \vdash A \text{ type}$
4.  $\Delta \vdash \gamma : \Gamma$
5.  $\Gamma \vdash a = a' : A, \quad \Gamma \vdash A = A' \text{ type}, \quad \Delta \vdash \gamma = \gamma' : \Gamma$

In this chapter, we'll discuss the rules for above judgements. Now, it's time to define explicit judgement rules that we referred.

## 1.2 Contexts Judgements

### Construction 1.2.1.

$$\frac{}{\vdash \mathbf{1} \text{ cx}} \qquad \frac{\vdash \Gamma \text{ cx} \quad \Gamma \vdash A \text{ type}}{\vdash \Gamma.A \text{ cx}}$$

Here,  $\mathbf{1}$  is empty context and each context is just list of types. (No variable names) It means that we use De Bruijn index, i.e. in each context, index automatically determines the variable in context.

## 1.3 Substitution Judgements

However, before that the most important one is understanding direction of traffic way (from now on, substitution).

### Definition 1.3.1.

If  $\Delta \vdash \gamma : \Gamma$ , then  $\gamma$  is substitution from  $\Delta$  to  $\Gamma$ . i.e.

$$\gamma : \Delta \rightarrow \Gamma$$

However, it's role is send types and terms of  $\Gamma$  into  $\Delta$ . (Note : Direction is important)

### 1.3. Intuitions for direction of substitutions

To understand why here use this notation, see following example. Imagine that contexts are 'sets' and substitutions are function between them. There are 2 sets  $\Delta, \Gamma$  and mapping  $\gamma : \Delta \rightarrow \Gamma$ . Suppose that there is a function  $g : \Gamma \rightarrow \mathbb{R}$ , which is defined on set  $\Gamma$ . How can we 'use' this function in  $\Delta$  set? One way is that,

$$g : \Gamma \rightarrow \mathbb{R} \implies g \circ \gamma : \Delta \rightarrow \mathbb{R}$$

Then we can 'use' the function  $g$  in domain  $\Delta$ . This exactly corresponds in our notation. For  $\Delta \vdash \gamma : \Gamma$ , when we define  $\gamma : \Delta \rightarrow \Gamma$ ,

$$\Gamma \vdash A \text{ type} \implies \Delta \vdash A[\gamma] \text{ type}$$

Since  $A$  is type in  $\Gamma$  and the direction of substitution is  $\gamma : \Delta \rightarrow \Gamma$ , we say that  $A[\gamma]$  is pull-backed type of  $A$  through  $\gamma : \Delta \rightarrow \Gamma$ .

First, as above we can easily define the application rule of substitution.

## Construction 1.3.2.

$$\frac{\Delta \vdash \gamma : \Gamma \quad \Gamma \vdash A \text{ type}}{\Delta \vdash A[\gamma] \text{ type}} \qquad \frac{\Delta \vdash \gamma : \Gamma \quad \Gamma \vdash a : A}{\Delta \vdash a[\gamma] : A[\gamma]}$$

These rules give us that we can immigrate(pull-back) terms and types of  $\Gamma$  into  $\Delta$ . And now, in our setting each contexts are different ‘worlds’. So we need to introduce explicit **weakening rule**, which intuitively means that we can bring well-formed types and terms into  $\Gamma$  into  $\Gamma.A$ , expanded context.

## Construction 1.3.3.

$$\frac{\Gamma \vdash A \text{ type}}{\Gamma.A \vdash \mathbf{p} : \Gamma}$$

In diagram,

$$\Gamma.A \xrightarrow{p} \Gamma$$

Means that, we can pull-back types  $B$  and terms  $t$  in world  $\Gamma$  into  $\Gamma.A$  by write  $B[\mathbf{p}], t[\mathbf{p}]$ . Moreover, we can introduce following rules. (Actually, following rules need because our context-substitution system become a category.)

## Construction 1.3.4.

$$\frac{\vdash \Gamma \text{ cx}}{\Gamma \vdash \mathbf{id} : \Gamma} \qquad \frac{\Gamma_2 \vdash \gamma_1 : \Gamma_1 \quad \Gamma_1 \vdash \gamma_0 : \Gamma_0}{\Gamma_2 \vdash \gamma_0 \circ \gamma_1 : \Gamma_0}$$

Second one can be conflict when we see first. However, let’s draw the diagram.

$$\begin{array}{ccc} \Gamma_2 & \xrightarrow{\gamma_1} & \Gamma_1 \\ \gamma_0 \circ \gamma_1 \downarrow & \swarrow \gamma_0 & \\ \Gamma_0 & & \end{array}$$

Then, above representation is very clear. It become same notation in our function calculus. Similarly,

## Construction 1.3.5.

$$\frac{\Delta \vdash \gamma : \Gamma}{\Delta \vdash \gamma \circ \mathbf{id} = \mathbf{id} \circ \gamma = \gamma : \Gamma} \qquad \frac{\Gamma_3 \vdash \gamma_2 : \Gamma_2 \quad \Gamma_2 \vdash \gamma_1 : \Gamma_1 \quad \Gamma_1 \vdash \gamma_0 : \Gamma_0}{\Gamma_3 \vdash (\gamma_0 \circ \gamma_1) \circ \gamma_2 = \gamma_0 \circ (\gamma_1 \circ \gamma_2) : \Gamma_0}$$

This two rules are also very clear when we see the diagram.

$$id \hookrightarrow \Delta \xrightarrow{\gamma} \Gamma \hookrightarrow id$$

$$\begin{array}{ccc}
\Gamma_3 & \xrightarrow{\gamma_2} & \Gamma_2 \\
(\gamma_0 \circ \gamma_1) \circ \gamma_2 \downarrow & \swarrow \gamma_0 \circ \gamma_1 & \downarrow \gamma_1 \\
\Gamma_0 & \xleftarrow{\gamma_0} & \Gamma_1
\end{array}
\iff
\begin{array}{ccc}
\Gamma_3 & \xrightarrow{\gamma_2} & \Gamma_2 \\
\gamma_0 \circ (\gamma_1 \circ \gamma_2) \downarrow & \swarrow \gamma_1 \circ \gamma_2 & \downarrow \gamma_1 \\
\Gamma_0 & \xleftarrow{\gamma_0} & \Gamma_1
\end{array}$$

Then we can imagine following equivalent rules clearly.

#### Construction 1.3.6.

$$\frac{\Gamma \vdash A \text{ type}}{\Gamma \vdash A[\mathbf{id}] = A \text{ type}} \qquad \frac{\Gamma \vdash a : A}{\Gamma \vdash a[\mathbf{id}] = a : A}$$

$$\frac{\Gamma_2 \vdash \gamma_1 : \Gamma_1 \quad \Gamma_1 \vdash \gamma_0 : \Gamma_0 \quad \Gamma_0 \vdash A \text{ type}}{\Gamma_2 \vdash A[\gamma_0 \circ \gamma_1] = A[\gamma_0][\gamma_1] \text{ type}}$$

$$\frac{\Gamma_2 \vdash \gamma_1 : \Gamma_1 \quad \Gamma_1 \vdash \gamma_0 : \Gamma_0 \quad \Gamma_0 \vdash a : A}{\Gamma_2 \vdash a[\gamma_0 \circ \gamma_1] = a[\gamma_0][\gamma_1] : A[\gamma_0 \circ \gamma_1]}$$

These rules are very intuitable. When we draw diagram,

$$\begin{array}{ccc}
\Gamma_2 & \xrightarrow{\gamma_1} & \Gamma_1 \\
\gamma_0 \circ \gamma_1 \downarrow & \swarrow \gamma_0 & \\
\Gamma_0 & & 
\end{array}$$

For example, look the last rule.  $a$  is term in the world  $\Gamma_0$ . We want to bring this term into  $\Gamma_2$ . Bring means that, we want to use ‘function’  $a$  in  $\Gamma_2$ . i.e. such term in  $\Gamma_2$  is working same roles with  $a$  in  $\Gamma_0$ . We’ve studied in box 1.3, there are 2 ways to bring  $a$  from  $\Gamma_0$  into  $\Gamma_2$ .

$$\begin{array}{ccc}
\Gamma_2 & \xrightarrow{\gamma_1} & \Gamma_1 \\
\gamma_0 \circ \gamma_1 \downarrow & \swarrow \gamma_0 & \\
\Gamma_0 & & 
\end{array}$$

One is pulling back via red way, and another one is via blue way.  $a$  via red way is  $a[\gamma_0 \circ \gamma_1]$  in  $\Gamma_2$ , and via blue way is  $a[\gamma_0][\gamma_1]$  (Actually,  $a[\gamma_0]$  is  $a$  in  $\Gamma_1$  world. So we take it again into  $\Gamma_2$ .) However, how can we use variable in well-formed context  $\Gamma$  without own names? What is the meaning of De Bruijn index? Let’s see how can we distinguish variables in each ‘world’  $\Gamma$ .

#### Construction 1.3.7.

$$\frac{\Gamma \vdash A \text{ type}}{\Gamma.A \vdash \mathbf{q} : A[\mathbf{p}]}$$

We can see context as a stack.  $\mathbf{q}$  always means the top element of stack. However, stack  $\Gamma.A$  contains type  $A$  as top, which is type in  $\Gamma$  world. So the top variable's type in  $\Gamma.A$  is  $A[\mathbf{p}]$  which we take  $A$  from  $\Gamma$  into  $\Gamma.A$ . Similarly, we can access the any index of stack by following rule which is derivable from 1.3

**Corolary 1.3.8.**

$$\frac{\Gamma \vdash A \text{ type} \quad \Gamma.A \vdash B_1 \text{ type} \quad \cdots \Gamma.A.B_1 \cdots \vdash B_n \text{ type}}{\Gamma.A.B_1 \cdots .B_n \vdash \mathbf{q}[\mathbf{p}^n] : A[\mathbf{p}^{n+1}]}$$

Now, it's time to discuss our main theme in this subsection 1.3. Our ultimate goal is defining 'valid traffic way from valid world to another valid world'. Let's discuss substitution judgement rules very intuitively. First, suppose that we only have rules for context judgements 1.2. The obvious thing that we must do first is, construct way from any  $\Gamma$  into  $\mathbf{1}$ . The rules are simple.

**Construction 1.3.9.**

$$\frac{\vdash \Gamma \text{ ex}}{\Gamma \vdash ! : \mathbf{1}} \qquad \frac{\Gamma \vdash \delta : \mathbf{1}}{\Gamma \vdash \delta =! : \mathbf{1}}$$

What does it mean? It is 'way' from  $\Gamma$  to  $\mathbf{1}$ . The role of this way is transfer terms and types in  $\mathbf{1}$  into  $\Gamma$  world. What terms and types are well-formed in  $\mathbf{1}$ ? Just closed terms and types (No Variables). It is obviously terms and types in  $\Gamma$  also. So we define the way is 'unique' here. With this substitution, we can always transfer closed terms (e.g. 1, (1+1)) and closed types from  $\mathbf{1}$  to any well-formed context  $\Gamma$ . Let's think this as a base case of our definition of traffic ways. Following construction determines how can we extends substitutions.

**Construction 1.3.10.**

$$\frac{\Delta \vdash \gamma : \Gamma \quad \Gamma \vdash A \text{ type} \quad \Delta \vdash a : A[\gamma]}{\Delta \vdash \gamma.a : \Gamma.A}$$

Let's draw diagram.

$$\begin{array}{ccc} \Delta & \xrightarrow{\gamma} & \Gamma \\ & \searrow \gamma.a & \uparrow p \parallel id.? \\ & & \Gamma.A \end{array}$$

Is it clear? It means that, suppose that  $\Gamma$  has  $n$  variables  $(x_1, \dots, x_n)$ . Here we denote names for each variables for representation. And then, in this context  $\Gamma$ ,  $A$  is well-formed type. Then automatically  $\Gamma.A$  is defined, where it has  $n + 1$  variables  $(x_1, \dots, x_n, x_{n+1})$ .



The key understanding is that, when we bring terms and types in  $\Gamma.A$  world, it can have  $x_1, \dots, x_{n+1}$  as symbol. When we meet  $x_{n+1}$ , then substitution it by  $a$ , and when we meet  $x_1, \dots, x_n$ , then substitution it by following  $\gamma$ . Then since  $a$  is well-formed term in  $\Delta$  and it's type is  $A[\gamma]$ , which do roles of  $A$  in  $\Delta$  world, so such substitution also works well. This is very powerful intuition. Now, with above understanding, let's define more challenging and powerful rules for substitutions.

**Construction 1.3.11.**

$$\frac{\Delta \vdash \gamma : \Gamma \quad \Gamma \vdash A \text{ type} \quad \Delta \vdash a : A[\gamma]}{\Delta \vdash \mathbf{p} \circ (\gamma.a) = \gamma : \Gamma}$$

$$\frac{\Delta \vdash \gamma : \Gamma \quad \Gamma \vdash A \text{ type} \quad \Delta \vdash a : A[\gamma]}{\Delta \vdash \mathbf{q}[\gamma.a] = a : A[\gamma]}$$

$$\frac{\Gamma \vdash A \text{ type} \quad \Delta \vdash \gamma : \Gamma.A}{\Delta \vdash \gamma = (\mathbf{p} \circ \gamma).\mathbf{q}[\gamma] : \Gamma.A}$$

There rules are seemed to be very unclear. However, see following diagram. For first rule,

$$\begin{array}{ccc} \Delta & \xrightarrow{\gamma} & \Gamma \\ & \searrow \gamma.a & \uparrow p \downarrow id.? \\ & & \Gamma.A \end{array}$$

We can represent the first rule as following way.

$$\begin{array}{ccc} \Delta & \xrightarrow{\gamma} & \Gamma \\ & \searrow \gamma.a & \uparrow p \downarrow id.? \\ & & \Gamma.A \end{array}$$

So we can intuitively know that  $\gamma$  and  $\mathbf{p} \circ (\gamma.a)$  represent same traffic way from  $\Delta$  to  $\Gamma$ . For second rule, we already explained in front page. When we meet  $x_{n+1}$  (i.e. meet  $q$  in  $\Gamma.A$ ) then bring it from  $\Gamma.A$  into  $\Delta$  and substitute by  $a$ , term of  $\Delta$ . This rule explicitly claim this intuition. For third rule, let's draw slightly changed diagram.

$$\begin{array}{ccc} \Delta & & \Gamma \\ & \searrow \gamma & \uparrow p \downarrow id.? \\ & & \Gamma.A \end{array}$$

Can we see what is  $(\mathbf{p} \circ \gamma)$  here? We can draw

$$\begin{array}{ccc} \Delta & \xrightarrow{p \circ \gamma} & \Gamma \\ & \searrow \gamma & \uparrow p \downarrow id.? \\ & & \Gamma.A \end{array}$$

Now, see ???. When we make  $\mathbf{p} \circ \gamma : \Delta \rightarrow \Gamma$  to something that  $\Delta \rightarrow \Gamma.A$ , we append something to  $\mathbf{p} \circ \gamma$ . Here, actually the answer is already given. Suppose  $q$  in  $\Gamma.A$ , it has type  $A[\mathbf{p}]$  in  $\Gamma.A$ . We define  $\gamma$  by, how we take variable  $\mathbf{q}$  in  $\Gamma.A$  to  $\Delta$ ? Actually its answer is  $\mathbf{q}[\gamma]$ . It has type  $A[\mathbf{p}][\gamma] = A[\mathbf{p} \circ \gamma]$  in  $\Delta$ . So to make same substitution with  $\gamma$ , we must append  $\mathbf{q}[\gamma]$  at  $\mathbf{p} \circ \gamma$ . This intuition gives us  $\gamma = (\mathbf{p} \circ \gamma).\mathbf{q}[\gamma]$  in  $\Delta$ .

### Exercise 2.1

Prove that if  $\Gamma \vdash \gamma : \Gamma.A$ , then  $\mathbf{p} \circ \gamma = \mathbf{id}$

*Proof.* How can we define  $\gamma$ ? The intuitive meaning of  $\gamma$  is that, way to bring types and terms of  $\Gamma.A$  into  $\Gamma$ . Imagine that  $\Gamma.A$  contains types of  $x_1, \dots, x_{n+1}$  and  $\Gamma$  contains types of  $x_1, \dots, x_n$ . To bring types and terms in  $\Gamma.A$  into  $\Gamma$ , only to do is when we meet  $x_{n+1}$ , substitute it with well-formed term in  $\Gamma$  context. By this way we can delete the appearance of  $x_{n+1}$  and make types and terms in  $\Gamma.A$  become types and terms in  $\Gamma$ . So,  $\gamma$  is defined by following way. This is construction 1.3.10.

$$\frac{\Gamma \vdash \mathbf{id} : \Gamma \quad \Gamma \vdash A \text{ type} \quad \Gamma \vdash a : A}{\Gamma \vdash \underbrace{\mathbf{id}.a}_{\gamma} : \Gamma.A}$$

However, according to construction 1.3.11 we can know that

$$\mathbf{p} \circ (\mathbf{id}.a) = \mathbf{id}$$

□

### Exercise 2.2

Show that  $(\gamma.a) \circ \delta = (\gamma \circ \delta).a[\delta]$

*Proof.* Recall third rule in construction 1.3.11.,

$$\frac{\Gamma \vdash A \text{ type} \quad \Delta \vdash \gamma^* : \Gamma.A}{\Delta \vdash \gamma^* = (\mathbf{p} \circ \gamma^*).\mathbf{q}[\gamma^*] : \Gamma.A}$$

Here, the pre-suppositions of problem is that  $\Delta \vdash \gamma : \Gamma$ ,  $\Delta \vdash a : A[\gamma]$ ,  $\Gamma \vdash A \text{ type}$  and  $\delta$  be any substitution  $\delta : \Lambda \rightarrow \Delta$ . In diagram,

$$\begin{array}{ccc} \Delta & \xrightarrow{\gamma} & \Gamma \\ \delta \uparrow & \searrow \gamma.a & \\ \Lambda & & \Gamma.A \end{array}$$

(Continue in Next Page . . .)

□

## Exercise 2.2

For intuition, we can draw following substitution.

$$\begin{array}{ccc} \Delta & \xrightarrow{\gamma} & \Gamma \\ \delta \uparrow & \searrow \gamma.a & \\ \Lambda & \xrightarrow{(\gamma.a) \circ \delta} & \Gamma.A \end{array}$$

And also,

$$\begin{array}{ccc} \Delta & \xrightarrow{\gamma} & \Gamma \\ \delta \uparrow & \nearrow \gamma \circ \delta & \\ \Lambda & & \Gamma.A \end{array} \quad \Rightarrow \quad \begin{array}{ccc} \Delta & \xrightarrow{\gamma} & \Gamma \\ \delta \uparrow & \nearrow \gamma \circ \delta & \\ \Lambda & & \Gamma.A \end{array}$$

Here our claim is that ? part is  $a[\delta]$  and  $(\gamma.a) \circ \delta = (\gamma \circ \delta).a[\delta]$ . At first, let

$$\gamma^* = (\gamma.a) \circ \delta$$

Then  $\Lambda \vdash \gamma^* : \Gamma.A$  because

$$\frac{\Lambda \vdash \delta : \Delta \quad \Delta \vdash \gamma.a : \Gamma.A}{\Lambda \vdash \underbrace{(\gamma.a) \circ \delta}_{\gamma^*} : \Gamma.A}$$

So we can apply construction 1.3.11.

$$\frac{\Gamma \vdash A \text{ type} \quad \Lambda \vdash (\gamma.a) \circ \delta : \Gamma.A}{\Lambda \vdash (\gamma.a) \circ \delta = \underbrace{(\mathbf{p} \circ ((\gamma.a) \circ \delta)).\mathbf{q}[(\gamma.a) \circ \delta]}_{(*)} : \Gamma.A}$$

Here, our claim is that  $(*) = (\gamma \circ \delta).a[\delta]$ . This is because,

$$\begin{aligned} & (\mathbf{p} \circ ((\gamma.a) \circ \delta)).\mathbf{q}[(\gamma.a) \circ \delta] \\ &= ((\mathbf{p} \circ (\gamma.a)) \circ \delta).\mathbf{q}[(\gamma.a) \circ \delta] \\ &= (\gamma \circ \delta).\mathbf{q}[(\gamma.a) \circ \delta] \\ &= (\gamma \circ \delta).(\mathbf{q}[\gamma.a][\delta]) \\ &= (\gamma \circ \delta).a[\delta] \end{aligned}$$

This is end of proof.

## Exercise 2.3

Given  $\Delta \vdash \gamma : \Gamma$  and  $\Gamma \vdash A$  type, construct  $\gamma.A$  such that  $\Delta.A[\gamma] \vdash \gamma.A : \Gamma.A$ .

*Proof.* Actually, this is also very clear notion. Let's see following diagram.

$$\begin{array}{ccc} \Delta & \xrightarrow{\gamma} & \Gamma \\ p_\Delta \uparrow & & \\ \Delta.A[\gamma] & & \Gamma.A \end{array}$$

This implies that

$$\begin{array}{ccc} \Delta & \xrightarrow{\gamma} & \Gamma \\ p_\Delta \uparrow & \nearrow \gamma \circ p_\Delta & \\ \Delta.A[\gamma] & & \Gamma.A \end{array}$$

And our intuition for  $\gamma.A$  is that

$$\begin{array}{ccc} \Delta & \xrightarrow{\gamma} & \Gamma \\ p_\Delta \uparrow & \nearrow \gamma \circ p_\Delta & \\ \Delta.A[\gamma] & \xrightarrow[\text{(\gamma \circ p_\Delta).?}]{\gamma.A} & \Gamma.A \end{array}$$

We can apply substitution extend rule here.

$$\frac{\Delta.A[\gamma] \vdash \gamma \circ \mathbf{p}_\Delta : \Gamma \quad \Gamma \vdash A \text{ type} \quad \Delta.A[\gamma] \vdash \mathbf{q} : A[\gamma][\mathbf{p}_\Delta] = A[\gamma \circ \mathbf{p}_\Delta]}{\Delta.A[\gamma] \vdash (\gamma \circ \mathbf{p}_\Delta).\mathbf{q} : \Gamma.A}$$

Actually, this is our construction for  $\gamma.A$ .

$$\gamma.A := (\gamma \circ \mathbf{p}).\mathbf{q}, \quad \gamma.A : \Delta.A[\gamma] \rightarrow \Gamma.A$$

□

# Chapter 2

## Internalizing Judgemental Structure : $\Pi, \Sigma, \mathbf{Eq}, \mathbf{Unit}$

### 2.1 Basic Intuitions

We noted that Dependent Type Theory is similar with First Order Logic. In first order logic, there exists notion of assignment  $s$  in Interpretation  $\mathfrak{A} = \langle |\mathfrak{A}|, I \rangle$  and variants of assignments. Both are corresponds into context  $\Gamma$  and substitution  $\gamma$  in chapter 1. And then, what's the remaining things? Since dependent type is similar with FOL formulas, we need to construct intuition of correspondings between them. First, FOL terms are corresponds with ‘terms’ in our system. Second, predicates are corresponds with ‘type(dependent type)’ in our system. These are all what we talk about in previous chapter. Remain things are connectives, quantifiers, true and false. (And one specific predicate, equality judgement.) They are corresponds w.r.t. non-dependent connectives, dependent sum( $\exists$ ), dependent product( $\forall$ ), Unit and Void.

I’ll try to make more clear between those similarity, in each section. Before that, let’s claim the slogan for (any) these connective or quantifier constructors.

#### Slogan

Any such connectives or quantifiers in our system is given by claiming followings :

1. Natural Type-forming Operation
2. Natural Isomorphism Relating that Type’s terms to judgementally-determined structure.

First one means that for each newly introduced types in this chapter, there are explicit and meta-level clear construct operation for such type. i.e. We must have the recipe of type.

Second one means that, if we collect all terms that have types that introduced here, then the set must have isomorphism (meta-level meaning) with external (meta-level) structure.

## 2.2 Dependent Sums

First of all, let's make clear all the intuitions for dependent sum.

1. The type former of dependent sum is  $\Sigma$ . We'll define soon.
2. Dependent sum's judgementally-determined structure is 'dependent pair'.
3. Dependent sum is similar to  $\exists$ .

What is dependent pair? Imagine following 'type' in dependent type system.

$$(n : Nat, Vec A n)$$

Actually, this is not matched with our construction of De Bruijn index. Matched version is that

$$(Nat, Vec A \mathbf{q})$$

Then imagin the Curry-Howard corresponds here. The term of above type is like 'one' specific example of above pair, for example

$$(2, [a, a'])$$

We can understand that this is proof for

$$(\exists n : Nat)(Vec A n)$$

In this intuition, we say that dependent sum is similar to  $\exists$ . From now on, let's construct above intuitions explicitly by defining operations, isomorphisms, and rules.

### Construction 2.2.1. Natural Type Formal

$$\Sigma_{\Gamma} : \left( \biguplus_{A \in Ty(\Gamma)} Ty(\Gamma.A) \right) \rightarrow Ty(\Gamma)$$

The domain space's meaning is (1) choose valid type  $A$  in  $\Gamma$  context and (2) Under previous choose, choose valid type in  $\Gamma.A$ . Then the codomain space's meaning is that, by this  $\Sigma_{\Gamma}$  operator, such pair become valid type in  $\Gamma$ . Then where is 'Natural' ? It's intuitive meaning is that, we can feel free to choose context  $\Gamma$  here. i.e. freely substitute and then change context, the operator's working does not changed. We can write above natural mapping as following inference rules.

### Construction 2.2.1. Natural Type Formal ( inference rule ver. )

$$\frac{\Gamma \vdash A \text{ type} \quad \Gamma.A \vdash B \text{ type}}{\Gamma \vdash \Sigma(A, B) \text{ type}} \qquad \frac{\Delta \vdash \gamma : \Gamma \quad \Gamma \vdash A \text{ type} \quad \Gamma.A \vdash B \text{ type}}{\Delta \vdash \Sigma(A, B)[\gamma] = \Sigma(A[\gamma], B[\gamma.A]) \text{ type}}$$

For the second rule, it can be not clear that why  $B[\gamma.A]$  introduced. Let's draw the diagram. See this construction for  $\gamma.A$

$$\Delta \xrightarrow{\gamma} \Gamma$$

$$\Delta.A[\gamma] \xrightarrow[\gamma.A]{\text{red}} \Gamma.A$$

We can pull-back  $A$  form  $\Gamma$  to  $\Delta$  through  $\gamma$ . And note that  $B$  is type in the  $\Gamma.A$ . So if we pulling back this into  $\Delta.A[\gamma]$ , using  $\gamma.A$ . So, the pull-backed type is  $B[\gamma.A]$  in  $\Delta.A[\gamma]$ . It means that

$$\Delta \vdash A[\gamma] \text{ type}, \quad \Delta.A[\gamma] \vdash B[\gamma.A] \text{ type}$$

Then the type-formal implies that

$$\Delta \vdash \Sigma(A[\gamma], B[\gamma.A]) \text{ type}$$

And naturality means that, this type works same with  $\Sigma(A, B)$  in  $\Gamma$ . So construct identity with it's pull-backed version  $\Sigma(A, B)[\gamma]$  and above.

.