Simple Type Theory

November 19, 2020

1 Introduction

In this document we describe typed lambda calculus with sums. This is a formalization of the type theory described in [1], but so that well formed contexts and types are generated following explicit inference rules, like in the appendix A2 of [2]. The terminology is mostly taken from [3].

2 Basics

A **term** is a value of a **type**. Some terms are **variables** (as we explain later). Each term t has a set $FV(t)$ of **free variables** (as we explain later).

There are six kinds of expressions:

1. A **typing declaration** $x : A$ says that x is a term of type A.
2. A **universal declaration** $A \quad \mathrm{type}$ says that A is a type. When we have a stack of universes, this is equivalent to saying $A : \mathbb{U}_0$ in HoTT.
3. An **equality declaration** $x = y : A$ says values x and y of type A are equal.
4. A **context** Γ is a list, with each of its entries as a typing declaration or a universal declaration. We write $\Gamma :: \Delta$ to denote the concatenation of lists.
5. A **context declaration** $\Gamma \quad \mathrm{ctx}$ is a declaration that the context Γ is "well formed" (the meaning will be clear later from the rules).
6. A **judgment** is something of the form $\Gamma \vdash d$ where Γ is a context, and d is either a typing declaration or a universal declaration or an equality declaration. Sometimes we call d the **declaration** of the judgment $\Gamma \vdash d$.

A **rule** is something of the form

$$\frac{J_1 \quad J_2 \quad \dots \quad J_n}{K}$$

where $J_1, J_2, \ldots, J_n$ and K are all judgements. The meaning of the rule is that if each judgement in $J_1, J_2, \ldots, J_n$ can be derived in the type theory then judgement K may also be derived. Judgements can be stacked to make proof trees. An axiom is a rule

$$\frac{}{K}$$

with no pre-requisits.

In addition to the assumed rules (which we name)

3 Forming base types

We write . to denote the empty context. The fact that the empty context is well formed is formalized by the rule:

$$\frac{}{. \quad \mathrm{ctx}} \text{ ctx-EMP} \tag{1}$$

The next rule allows a well formed context to be extended by introducing a **base type** A**:**

$$\frac{\Gamma \quad \mathrm{ctx}}{\Gamma :: (A \quad \mathrm{type}) \quad \mathrm{ctx}} \text{ ctx-EXT1} \tag{2}$$

The base type A must not appear in the context Γ. Here we assume we have some list of base types [1]. If we are trying to model a paricular system we may have specific base types ready, but for now let us just think of base types as variable types (although in this document we reserve the phrase “variable” for terms). So it is fine for A to be any type new to the context.

We can convert from well formed contexts to judgements about universal declarations using the following:

$$\frac{\Gamma :: (A \quad \mathrm{type}) :: \Delta \quad \mathrm{ctx}}{\Gamma :: (A \quad \mathrm{type}) :: \Delta \vdash A \quad \mathrm{type}} \text{ Vble1} \tag{3}$$

where Γ and Δ are contexts.

All the rules we have discussed could be derived in homotopy type theory (HoTT).

3.1 Example

Here is an example of how we derive the judgement $A \quad \mathrm{type} \vdash A \quad \mathrm{type}$.

$$\frac{\dfrac{\dfrac{}{. \quad \mathrm{ctx}}}{(A \quad \mathrm{type}) \quad \mathrm{ctx}}}{(A \quad \mathrm{type}) \vdash A \quad \mathrm{type}} \tag{4}$$

Here we use ctx-EMP then ctx-EXT1 then Vble1.

4 Forming Other Types

This rule lets us form the unit type

$$\frac{\Gamma \quad \text{ctx}}{\Gamma \vdash 1 \quad \text{type}} \text{ Unit-Form} \tag{5}$$

Next, product types

$$\frac{\Gamma \vdash A \quad \text{type} \quad \Gamma \vdash B \quad \text{type}}{\Gamma \vdash A \times B \quad \text{type}} \text{ Product-Form} \tag{6}$$

Next the empty type

$$\frac{\Gamma \quad \text{ctx}}{\Gamma \vdash 0 \quad \text{type}} \text{ Empty-Intro} \tag{7}$$

Next, sum types

$$\frac{\Gamma \vdash A \quad \text{type} \quad \Gamma \vdash B \quad \text{type}}{\Gamma \vdash A + B \quad \text{type}} \text{ Sum-Form} \tag{8}$$

5 Forming Variables

We make a variable x of type A using the following rule

$$\frac{\Gamma \vdash A \quad \text{type}}{\Gamma :: (x : A) \quad \text{ctx}} \text{ ctx-EXT2} \tag{9}$$

Note that x must be distinct from each term in the context Γ.

Note the ctx-EXT2 is the first time we have made a term (the variable x is a term of type A). The set of free variables of this newly made x is $FV(x) = \{x\}$.

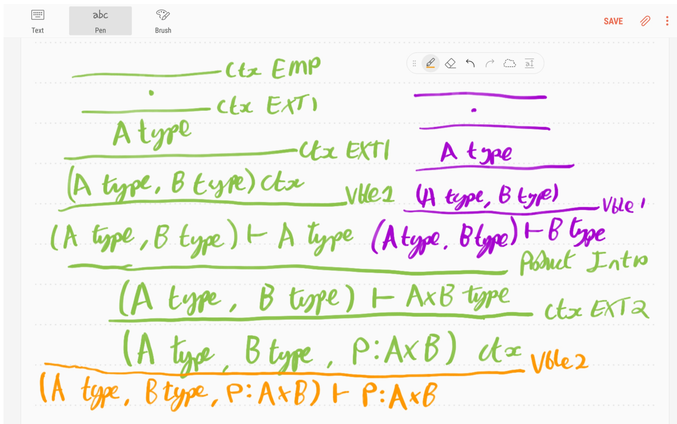
Judgments about variables can be formed with the following rule

$$\frac{\Gamma :: (x : A) :: \Delta \quad \text{ctx}}{\Gamma :: (x : A) :: \Delta \vdash x : A} \text{ Vble2} \tag{10}$$

5.1 Example

The following picture shows how we derive the rule

$$\frac{}{(A \quad \text{type}, B \quad \text{type}, p : A \times B) \vdash p : A \times B} \tag{11}$$



6 The Other Typing Rules

6.1 Products

$$\frac{\Gamma \quad \text{ctx}}{\Gamma \vdash * : 1} \text{ Unit-Intro} \tag{12}$$

Here $*$ has the empty set $FV(*) = \{\}$ of free variables.

$$\frac{\Gamma \vdash a : A \quad \Gamma \vdash b : B}{\Gamma \vdash \langle a, b \rangle : A \times B} \text{ Product-Intro} \tag{13}$$

Here $FV(\langle a, b \rangle) = FV(a) \cup FV(b)$ where $\cup$ denotes the set theoretic union.

$$\frac{\Gamma \vdash p : A \times B}{\Gamma \vdash \text{fst}(p) : A} \text{ Product-Elim1} \tag{14}$$

$FV(\text{fst}(p)) = FV(p)$.

$$\frac{\Gamma \vdash p : A \times B}{\Gamma \vdash \text{snd}(p) : B} \text{ Product-Elim2} \tag{15}$$

$FV(\text{snd}(p)) = FV(p)$.

6.2 Sums

References

[1] *Extensional Normalisation and Type-Directed Partial Evaluation for Typed Lambda Calculus with Sums* Vincent Balat, Roberto Di Cosmo, Marcelo Fiore

[2] Homotopy Type Theory Book

[3] ncatlab type theory page ncatlab type theory

[4]