Introduction to dependent type theory

Ali Caglayan

March 12, 2019

Contents

1 Judgements **1**

1 Judgements

We will now describe simply typed lambda calculus using our developed way of working with syntax. We will first describe judgements and how to specify a type system. Then our first example will be the simply typed lambda calculus. We use the ideas developed in [1] though these ideas are much older. [Probably tracable back to Gentzen].

Definition 1.0.1. The notion of a *judgement* or *assertion* is a logical statement about an abt. The property or relation itself is called a *judgement form.* The judgement that an object or objects have that property or stand in relation is said to be an *instance* of that judgement form. A judgment form has also historically been called a *predicate* and its instances called *subjects.*

Remark 1.0.2. Typically a judgement is denoted J. We can write $a\ \mathsf{J}$, $\mathsf{J}\ a$ to denote the judgment asserting that the judgement form J holds for the abt a. For more abts this can also be written prefix, infix, etc. This will be done for readability. Typically for an unspecified judgement, that is an instance of some judgement form, we will write J.

Definition 1.0.3. An *inductive definition* of a judgement form consists of a collection of rules of the form

$$\frac{J_1 \quad \cdots \quad J_k}{J}$$

in which J and $J_1, \ldots, J_k$ are all judgements of the form being defined. THe judgements above the horizontal line are called the *preimises* of the rules, and the judgement below the line is called its *conclusion.* A rule with no premises is called an *axiom.*

Remark 1.0.4. An inference rule is read as starting that the premises are *sufficient* for the conclusion: to show J, it is enough to show each of $J_1, \dots J_k$. Axioms hold unconditionally. If the conclusion of a rule holds it is not necesserily the case that the premises held, in that the conclusion could have been derived by another rule.

Example 1.0.5. Consider the following judgement from $-$ nat, where a nat is read as "a is a natural number". The following rules form an inductive definition of the judgement form $-$ nat:

$$\frac{}{\texttt{zero}\ \mathsf{nat}} \qquad\qquad \frac{a\ \mathsf{nat}}{\texttt{succ}(a)\ \mathsf{nat}}$$

References

[1] Robert Harper. *Practical Foundations for Programming Languages.* Cambridge University Press, 2 edition, 2016.