ORECON PROGRAMMING LANGUAGES SUMMER SCHOOL

Introduction to Type Theories — Anja Petković Komel

Lecture 1 - Introductory Lecture *June 24, 2025*

1 Introduction

1.1 Why Type Theories?

For mathematicians: they're beautiful mathematical foundations that can express a constructive approach

For everyone else: proof assistants are built on them

1.2 So why proof assistants?

It's important to have high assurance systems be highly assured when lives and livelihoods are at stake. Also formalizing mathematics is nice, like four color map theorem, odd order theorem, Lean maths proofs, etc.

There are many proof assistants, including Rocq, Agda, LEAN, and other less popular proof assistants such as Nuprl, Isabelle/HOL, Mizar, redTT, cubicaltt, F*, Andromeda 2, Arend, etc.

1.3 Structure of lectures

1. Martin-Löf Type Theory. Chapter 1 of textbook
2. Calculus of Inductive Constructions (CIC) and Rocq. Interactively proving (easy) theorems, Live coding + slides.
3. If time permits: more Rocq / Meta-theory of type theories.

We will discuss type theories syntactically.

2 Martin-Löf Type Theory

- In type theory, every element comes with a type ($a : A$)
- Type theory = deductive system

$$\frac{\Gamma \vdash a : A \quad \Gamma \vdash f : A \to B}{\Gamma \vdash f(a) : B}$$

There are 4 kinds of judgments:

1. well-formed types: $\Gamma \vdash A$ type
2. judgmentally equal types
3. well-formed term
4. judgmentally equal terms

Contexts contain terms and their types. A context of length 0 is called an **empty context**. Dependent types such as type families (based off of term types) and sections (terms themselves) may depends on types present in the context.

3 Inference rules

See slides.

Composed of two kinds of rules:

1. Structural rules (for all)
2. Rules for MLTT

3.1 Structural rules

Judgmental equality is an equivalence relation.

(derivable) Variable conversion: equivalent types may be "swapped out"

Substitution: values may be "swapped in" for variables of the same type, and substituting judgmentally equal stuff results in judgmentally equal types.

Weakening: bad stuff doesn't happen when we add stuff to the context

Generic element/variable rule: if you assume something, you can use it.

3.2 Derivation

We want to derive the rule

$$\frac{\Gamma \vdash A \doteq A' \text{ type} \quad \Gamma \vdash a : A}{\Gamma \vdash a : A'}$$

Substitution is the only rule that could be applied first. See slides for full derivation.

4 MLTT Specifics

4.1 Dependent functions

These are functions where the type of the output may depend on the input.

Dependent function rule, which tells us how we may form dependent function types.

Formation rule, which tells us how to introduce new terms of dependent function types.

Introduction rule, which tells us how to use arbitrary terms of dependent function types.

Elimination (evaluation) rule, which tell us how the introduction and elimination rules interact. These computation rules guarantee that every term of a dependent function type behaves as expected: as a dependent function.

Dependent function rule

$$\textstyle\prod_{(x:A)} B(x)$$