universität
innsbruck

Extensional Type Theory

by the example of Martin-Löf Type Theory

Daniel Ranalter

Equality

$$2 = 2$$

$$2 = Succ(Succ(0))$$

$$2 = (\lambda x.x + x)1$$

$$Id_A(a + b, b + a)$$

Equality

$$2 = 2$$

Reflexivity

$$2 = Succ(Succ(0))$$

$$2 = (\lambda x.x + x)1$$

$$Id_A(a + b, b + a)$$

Equality

$$2 = 2$$

Reflexivity

$$2 = Succ(Succ(0))$$

Definitional Equality

$$2 = (\lambda x.x + x)1$$

$$Id_A(a + b, b + a)$$

Equality

$$2 = 2$$

Reflexivity

$$2 = Succ(Succ(0))$$

Definitional Equality

$$2 = (\lambda x.x + x)1$$

Computational Equality

$$Id_A(a + b, b + a)$$

Equality

$$2 = 2$$

Reflexivity

$$2 = Succ(Succ(0))$$

Definitional Equality

$$2 = (\lambda x.x + x)1$$

Computational Equality

$$Id_A(a + b, b + a)$$

Propositional Equality

Equality

$$2 = 2$$

Reflexivity

$$2 = Succ(Succ(0))$$

Definitional Equality

$$2 = (\lambda x.x + x)1$$

Computational Equality

$$Id_A(a + b, b + a)$$

~~Propositional~~ Equality

Equality

$$2 = 2$$

Reflexivity

$$2 = Succ(Succ(0))$$

Definitional Equality

$$2 = (\lambda x.x + x)1$$

Computational Equality

$$Id_A(a + b, b + a)$$

Equality Type

The Example Theory

Martin-Löf Type Theory

- Invented by Per Martin-Löf
- Mentioned several times in presentations last week
- Several versions, presented is MLTT9 published in 1984
- Known by several names
  - Constructive Type Theory
  - Intuitionistic Type Theory
- Dependent Types
- Infinite Chain of non-self-referential Universes
- Only a fragment will be presented

The Example Theory

λ_P - Judgements

$$\Gamma \vdash k : \Box \qquad \Gamma \vdash \phi : k$$

$$\Gamma \vdash M : \tau$$

Martin-Löf Type Theory - Judgements

$$\Gamma \vdash A \; type \qquad \Gamma \vdash a : A$$

$$\Gamma \vdash A = B \; type \qquad \Gamma \vdash a = b : A$$

The Example Theory

Rules of Inference - Overview

- Only Equality Type rules will be given explicitly
- Additionally there are:
  - Product Types, similar to λ_P
  - Sum Types, allowing for Disjunction and Existentials
  - Finite Set Types, allowing for Recursion and definition of False, True and Booleans
  - Type for $\mathbb{N}$
  - Universe Types

The Example Theory

$$\frac{A\ Type \quad a : A \quad b : A}{I(A, a, b)\ Type}\ \text{I-F I} \qquad \frac{A = C\ Type \quad a = b : A \quad c = d : C}{I(A, a, b) = I(C, c, d)\ Type}\ \text{I-F II}$$

$$\frac{a = b : A}{r : I(A, a, b)}\ \text{I-I I} \qquad \frac{a = b : A}{r = r : I(A, a, b)}\ \text{I-I II}$$

$$\frac{c : I(A, a, b)}{a = b : A}\ \text{I-EI} \qquad \frac{c : I(A, a, b)}{c = r : I(A, a, b)}\ \text{I-Eq}$$

The Example Theory

$$\frac{A\ Type \quad a : A \quad b : A}{I(A,a,b)\ Type}\ \text{I-F I}$$

$$\frac{A = C\ Type \quad a = b : A \quad c = d : C}{I(A,a,b) = I(C,c,d)\ Type}\ \text{I-F II}$$

$$\frac{a = b : A}{r : I(A,a,b)}\ \text{I-I I}$$

$$\frac{a = b : A}{r = r : I(A,a,b)}\ \text{I-I II}$$

$$\frac{c : I(A,a,b)}{a = b : A}\ \text{I-El}$$

$$\frac{c : I(A,a,b)}{c = r : I(A,a,b)}\ \text{I-Eq}$$

$$\frac{\begin{array}{c}[x : A]^h \\ \vdots \\ b : B\end{array}}{\lambda x.b : \Pi x : A.B}\ \Pi\text{-El I}$$

$$\frac{\begin{array}{c}[x : A]^h \\ \vdots \\ a = c : A \quad B = D\ Type\end{array}}{B[x \rightarrowtail a] = D[x \rightarrowtail c]\ Type}\ \text{Sub II}$$

The Example Theory

$$\frac{A\ Type \quad a : A \quad b : A}{I(A, a, b)\ Type}\ \text{I-F I}$$

$$\frac{A = C\ Type \quad a = b : A \quad c = d : C}{I(A, a, b) = I(C, c, d)\ Type}\ \text{I-F II}$$

$$\frac{a = b : A}{r : I(A, a, b)}\ \text{I-I I}$$

$$\frac{a = b : A}{r = r : I(A, a, b)}\ \text{I-I II}$$

$$\frac{c : I(A, a, b)}{a = b : A}\ \text{I-El}$$

$$\frac{c : I(A, a, b)}{c = r : I(A, a, b)}\ \text{I-Eq}$$

$$\frac{\begin{array}{c}[x : A]^h \\ \vdots \\ b : B\end{array}}{\lambda x.b : \forall x : A.B}\ \Pi\text{-El I}$$

$$\frac{\begin{array}{c}[x : A]^h \\ \vdots \\ a = c : A \quad B = D\ Type\end{array}}{B[x \rightarrowtail a] = D[x \rightarrowtail c]\ Type}\ \text{Sub II}$$

The Example Theory

$$\frac{A\ Type \quad a : A \quad b : A}{I(A,a,b)\ Type}\ \text{I-F I}$$

$$\frac{A = C\ Type \quad a = b : A \quad c = d : C}{I(A,a,b) = I(C,c,d)\ Type}\ \text{I-F II}$$

$$\frac{a = b : A}{r : I(A,a,b)}\ \text{I-I I}$$

$$\frac{a = b : A}{r = r : I(A,a,b)}\ \text{I-I II}$$

$$\frac{c : I(A,a,b)}{a = b : A}\ \text{I-El}$$

$$\frac{c : I(A,a,b)}{c = r : I(A,a,b)}\ \text{I-Eq}$$

$$\frac{\begin{array}{c}[x : A]^h \\ \vdots \\ b : B\end{array}}{\lambda x.b : A \to B}\ \Pi\text{-El I}\ (xnotfreeinb)$$

$$\frac{\begin{array}{c}[x : A]^h \\ \vdots \\ a = c : A \quad B = D\ Type\end{array}}{B[x \rightarrowtail a] = D[x \rightarrowtail c]\ Type}\ \text{Sub II}$$

Derivation example

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[z : I(A,x,y)]^{h2}}{x = y : A}\ \text{I-El} \qquad \dfrac{B(x)\ type\ (x : A)}{B(x) = B(y)}\ \text{Sub II}}{[w : B(x)]^{h1}}\ \text{Sub I}}{w : B(y)}\ \Pi\text{-El I (h1)}}{\lambda w.w : B(x) \to B(y)}\ \Pi\text{-El I (h2)}}{\lambda zw.w : I(A,x,y) \to B(x) \to B(y)}\ \Pi\text{-El I (y : A)}}{\lambda yzw.w : \forall y : A.I(A,x,y) \to B(x) \to B(y)}\ \Pi\text{-El I (x : A)}}{\lambda xyzw.w : \forall x : A.\forall y : A.I(A,x,y) \to B(x) \to B(y)}$$

Intensional vs. Extensional

Intensional Type Theory

- Makes use of subset of
  - (Reflexivity)
  - Definitional Equality (includes Comp. Equality)
  - Propositional Equality
- Used by all Calculi discussed in the lecture
- Type Checking is *decidable*

Intensional vs. Extensional

Intensional Type Theory

- Makes use of subset of
  - (Reflexivity)
  - Definitional Equality (includes Comp. Equality)
  - Propositional Equality
- Used by all Calculi discussed in the lecture
- Type Checking is *decidable*

Extensional Type Theory

- Definitional Equality and Propositional Equality are equal
- MLTT9 is an example of ETT
- Used practically nowhere $\rightarrow$ Homotopy Type Theory
- Type Checking is *undecidable*

The Example Theory - Again

$$\frac{A\ Type \quad a : A \quad b : A}{I(A, a, b)\ Type}\ \text{I-F I}$$

$$\frac{A = C\ Type \quad a = b : A \quad c = d : C}{I(A, a, b) = I(C, c, d)\ Type}\ \text{I-F II}$$

$$\frac{a = b : A}{r : I(A, a, b)}\ \text{I-I I}$$

$$\frac{a = b : A}{r = r : I(A, a, b)}\ \text{I-I II}$$

$$\frac{c : I(A, a, b)}{a = b : A}\ \text{I-El}$$

$$\frac{c : I(A, a, b)}{c = r : I(A, a, b)}\ \text{I-Eq}$$

$$\frac{\begin{array}{c}[x : A]^h \\ \vdots \\ b : B\end{array}}{\lambda x.b : \Pi x : A.B}\ \Pi\text{-El I}$$

$$\frac{\begin{array}{c}[x : A]^h \\ \vdots \\ a = c : A \quad B = D\ Type\end{array}}{B[x \rightarrowtail a] = D[x \rightarrowtail c]\ Type}\ \text{Sub II}$$

Intensional vs. Extensional

Undecidability of ETT

Sketch of proof sketch:

- Rule I-El implies that Type Checking is equivalent to Definitional Equality
- Fix Gödel Numbering of Turing Machines to allow self-application
- Construct primitive recursive function $f(\epsilon, t)$ s.t. $f(\epsilon, t) = 1$ iff TM ϵ applied to itself halts with result 0 after at most t steps and 0 otherwise
- It can be shown that, if ϵ applied to itself halts with a result different from 0, $t : N \vdash I(N, f\ \epsilon\ t, 0)$ is inhabited

Intensional vs. Extensional

Undecidability of ETT

- Then I-El implies $t : N \vdash f\ \epsilon\ t = 0 : N$
- Note that this does not hold if ϵ applied to itself does halt with result 0
- Assume Definitional Equality is decidable, then there is a TM ϵ_0 that decides whether $t : N \vdash f\ \epsilon\ t = 0 : N$ holds
- ϵ_0 applied to itself terminates by construction and any result gives rise to a contradiction
- Definitional Equality is therefor not decidable and thus neither is Type Checking □

Thank you for your attention!

Daniel Ranalter

References I

- Martin Hofmann, Extensional concepts in intensional type theory.
- Per Martin-Loef, Intuitionistic type theory, Studies in proof theory, vol. 1, Bibliopolis, 1984.
- P. Martin-Löf, Constructive mathematics and computer programming, Proc. of a Discussion Meeting of the Royal Society of London on Mathematical Logic and Programming Languages (USA), Prentice-Hall, Inc., 1985, p. 167–184.