Extensionality in Type Theory
or
How to fix a broken mirror?

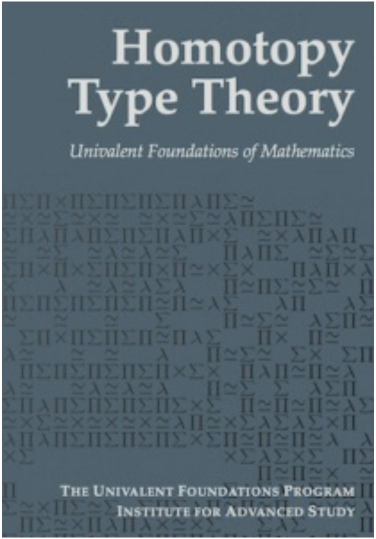
In honour of Pierre-Louis Curien's 60th birthday

Thorsten Altenkirch

Functional Programming Laboratory
School of Computer Science
University of Nottingham

September 10, 2013

The HoTT book



- Outcome of the Special Year on Homotopy Type Theory at Princeton.
- Introduces a very extensional type theory as a new foundation of Mathematics.
- *Informal* use of Type Theory.
- However HoTT as a programming language has a serious defect:
- We don't know how to execute its programs . . .

Type Theory at its best

- We define the type $\mathbb{N}$ by the constructors:

$$0 : \mathbb{N}$$
$$S : \mathbb{N} \to \mathbb{N}$$

- We recursively define the function $(+) : \mathbb{N} \to \mathbb{N} \to \mathbb{N}$:

$$0 + n :\equiv n$$
$$S(m) + n :\equiv S(m + n)$$

- We recursively define the function:
  $\mathrm{assoc} : \Pi_{i,j,k:\mathbb{N}}(i + j) + k =_{\mathbb{N}} i + (j + k)$

$$\mathrm{assoc}(0, j, k) :\equiv \mathrm{refl}(j + k)$$
$$\mathrm{assoc}(S(i), j, k) :\equiv \mathrm{respS}(\mathrm{assoc}(i, j, k))$$

- Theorem proving becomes functional programming!

What is $=_{\mathbb{N}}$?

- We define $(=_{\mathbb{N}}) : \mathbb{N} \to \mathbb{N} \to \textbf{Type}$ by the constructors:

$$
\begin{aligned}
\mathit{refl0} &: 0 =_{\mathbb{N}} 0 \\
\mathit{respS} &: \Pi_{m,n:\mathbb{N}}(m =_{\mathbb{N}} n) \to S(m) =_{\mathbb{N}} S(n)
\end{aligned}
$$

- We can show that $=_{\mathbb{N}}$ is an equivalence relation by deriving refl,sym,trans using recursion.
- We can also show that $=_{\mathbb{N}}$ is substitutive by constructing $\mathrm{subst}_{\mathbb{N}} : \Pi_{P:\mathbb{N}\to\textbf{Type}}\Pi_{m,n:\mathbb{N}} m =_{\mathbb{N}} n \to P(m) \to P(n)$

The type of computations

- Given A : **Type**, we introduce the type $\infty(A)$: **Type**, of *computations* of type A.
- Values are $\#(t) : \infty(A)$ where $t : A$ is a term of type A.
- E.g. $\#(3+4) \not\equiv \#(7)$.
- Given $d : \infty(A)$ we can force the computation $\flat(d) : A$.
- $\flat(\#(t))$ is the value associated to t.
- E.g. $\flat(\#(3+4)) \equiv 7$.

Conatural numbers

- Using ∞ we define the type of conatural numbers $\mathbb{N}^\infty$ by the constructors:

$$0 : \mathbb{N}^\infty$$
$$S : \infty(\mathbb{N}^\infty) \to \mathbb{N}^\infty$$

- We can recursively define $\perp : \mathbb{N}^\infty$:
  $\perp :\equiv S(\#(\perp))$.
- We recursively define the function $(+) : \mathbb{N}^\infty \to \mathbb{N}^\infty \to \mathbb{N}^\infty$:

$$0 + n :\equiv n$$
$$S(m) + n :\equiv S(\#(\flat(m) + n))$$

- We recursively define the function:
  $\mathrm{assoc} : \Pi_{i,j,k:\mathbb{N}^\infty}(i + j) + k =_{\mathbb{N}^\infty} i + (j + k)$

$$\mathrm{assoc}(0, j, k) :\equiv \mathrm{refl}(j + k)$$
$$\mathrm{assoc}(S(i), j, k) :\equiv \mathrm{respS}(\#(\mathrm{assoc}(\flat(i), j, k))$$

What is $=_{\mathbb{N}^\infty}$?

- We define $(=_{\mathbb{N}^\infty}) : \mathbb{N}^\infty \to \mathbb{N}^\infty \to \textbf{Type}$ by the constructors:

$$\begin{aligned} \mathit{refl0} :& 0 =_{\mathbb{N}^\infty} 0 \\ \mathit{respS} :& \Pi_{m,n:\infty(\mathbb{N}^\infty)} \infty(\flat(m) =_{\mathbb{N}^\infty} \flat(n)) \to S(m) =_{\mathbb{N}^\infty} S(n) \end{aligned}$$

- We can show that $=_{\mathbb{N}^\infty}$ is an equivalence relation by deriving refl,sym,trans using recursion.
- It is impossible to derive $\mathrm{subst}_{\mathbb{N}^\infty} : \Pi_{P:\mathbb{N}^\infty \to \textbf{Type}} \Pi_{m,n:\mathbb{N}} m =_{\mathbb{N}^\infty} n \to P(m) \to P(n)$

Underivability of $\mathrm{subst}_{\mathbb{N}^\infty}$

- We recursively define $\bot' : \mathbb{N}^\infty$:
  $\bot' :\equiv S(\#(S(\#(\bot'))))$.
- Note that $\bot \not\equiv \bot'$.
- However, we can prove $p : \bot =_{\mathbb{N}^\infty} \bot'$:
  $p :\equiv \mathit{respS}(\#(\mathit{respS}(\#(p))))$
- Consider the context $\Gamma \equiv P : \mathbb{N}^\infty \to \mathbf{Type}, p : P((\bot)$.
- If there is a proof $\Gamma \vdash q : P(\bot')$ then $q \equiv p$ and $\bot \equiv \bot'$,
- This follows from an analysis of normal forms.
- Hence $\mathrm{subst}_{\mathbb{N}^\infty}$ is underivable.

The broken mirror

- Type Theory (as we know it) works well for finitary types like natural numbers ...
- To define infinitary types (like $\mathbb{N}^{\infty}$) we need to use computations to *describe* infinite structures.
- We would like to consider infinite structures as *propositionally equal*, if their infinite unfoldings are equal.
- Hence, propositional equality does not reflect definitional equality.
- Π-types are another instance of an infinite type where the intended equality (extensional equality of functions) doesn't agree with definitional equality of λ-abstractions.

The Universe of Types

- A similar issue arises for the universe of types.
- An element of the universe $A : \textbf{Type}$ is an intensional description of the actual type.
- We would like to identify types that are *semantically* equivalent.
- We define $(=_{\textbf{Type}}) : \textbf{Type} \to \textbf{Type} \to \textbf{Type}$:
  An element of $A =_{\textbf{Type}} B$ is given by the following components:

$$
\begin{aligned}
f &: A \to B \\
g &: B \to A \\
p &: \Pi_{a:A,b:B}\infty((f(a) =_B b) =_{\textbf{Type}} (a =_A g(b)))
\end{aligned}
$$

- We can show that $(=_{\textbf{Type}})$ is reflexive, symmetric and transitive.
- We cannot derive $\mathrm{subst}_{\textbf{Type}}$ (e.g. $\mathbb{N} \times \mathbb{N} =_{\textbf{Type}} \mathbb{N}$).
- $A =_{\textbf{Type}} B$ is equivalent to the equivalences defined in the HoTT book.
- Indeed its 2.5 times unfolding is the semiadjoint equivalence defined there.

Fixing the mirror ?

- Given $f : A \to B$ we call $\mathrm{resp}(f) : \Pi_{a,a':A} a =_A a' \to f(a) =_B f(a')$
- We can reduce $\mathrm{subst}_X(P)$ to $\mathrm{resp}(P)$:
  Given $p : a =_A a'$
  $\mathrm{resp}(P, p) : P(a) =_{\mathbf{Type}} P(b)$
  and the first component of $\mathrm{resp}(P, p)$ is a function $P(a) \to P(b)$.
- Can we add a computationally well behaved generalisation of resp to Type Theory?

What about J?

- Paulin-Mohring's version of the eliminator.
- Assume as given $x : A$

$$\begin{aligned} J : \Pi_{P:\Pi_{y:A} x =_A y \to \mathbf{Type}} & \\ P(x, \mathrm{refl}(x)) \to & \\ \Pi_{y:A} \Pi_{p:x =_A y} P(y, p) & \end{aligned}$$

- with the computation rule:
  $J(P, m, x, \mathrm{refl}(x)) :\equiv m$
- subst arises as the special case if P doesn't depend of $x =_A y$.
- Can we reduce J to subst and hence to resp?

Reducing J to subst

- We can rewrite J using a Σ-type:

$$
\begin{aligned}
J : \Pi_{P:(\Sigma_{y:A} x =_A y) \to \mathbf{Type}} \\
P(x, \mathrm{refl}(x)) \to \\
\Pi_{y:A} \Pi_{p:x=_A y} P(y, p)
\end{aligned}
$$

- To reduce J to subst we need to show that:
  $(x, \mathrm{refl}(x)) =_{\Sigma_{y:A} x =_A y)} (y, p)$
  given $p : x =_A y$.
- a proof of the equality of pairs is a pair of equality proofs, here:

$$
\begin{aligned}
q &: x =_A y \\
r &: \mathrm{subst}(\lambda y . x = y, q, \mathrm{refl}(x)) =_{x =_A y} p
\end{aligned}
$$

- We set $q :\equiv p$ then the type of r is equivalent to:
  $r : \mathrm{trans}(p, \mathrm{refl}(x)) =_{x =_A y} p$
- This shows that we need the laws of a category to derive J.

Equalities of proofs

- We assume that equality proofs form a category:

$$\begin{aligned}
\mathrm{trans}(\mathrm{refl}, p) &\equiv p \\
\mathrm{trans}(p, \mathrm{refl}) &\equiv p \\
\mathrm{trans}(\mathrm{trans}(p, q), r) &\equiv \mathrm{trans}(p, \mathrm{trans}(q, r))
\end{aligned}$$

- resp is functorial:

$$\begin{aligned}
\mathrm{resp}(f, \mathrm{refl}) &\equiv \mathrm{refl} \\
\mathrm{resp}(f, \mathrm{trans}(p, q)) &\equiv \mathrm{trans}(\mathrm{resp}(f, p), \mathrm{resp}(f, q))
\end{aligned}$$

- Given this we can derive J and its computation rule.
- Is the resulting definitional equality decidable?

Summary

- Type Theory as we know it only treats finitary types properly.
- To integrate infinite structures (∞, Π-types, **Type**) we need to define propositional equality by recursion over the structure of types.
- To obtain a computational well behaved theory we need to explain resp by recursion over the structure of terms.
- We define the equality of the universe **Type** in a way so that the univalence principle holds automatically.
- We are considering a strong definitional equality of equality proofs such that the laws of an ω-category hold definitionally, while the symmetry is only weak.

Related work

- *Extensional Equality in Intensional Type Theory*, A., LICS 99
- *Observational Equality, Now!*; A., McBride, Swiersta; PLPV 2007
- *Canonicity for 2-dimensional type theory*, Harper, Licata; POPL 2012
- *A Generalization of the Takeuti-Gandy Interpretation*, Barras, Coquand, Huber; Draft 2013