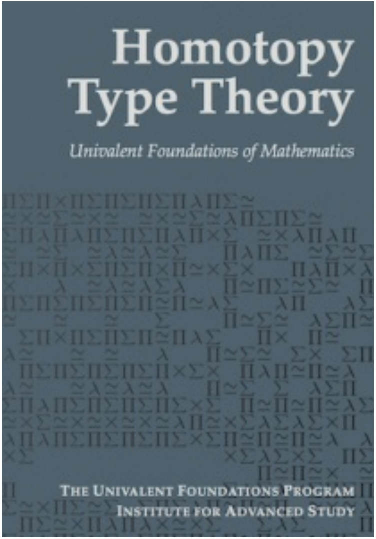
Homotopy Type Theory For Dummies

Thorsten Altenkirch

Functional Programming Laboratory
School of Computer Science
University of Nottingham

October 30, 2013

The HoTT book



- Outcome of the Special Year on Homotopy Type Theory at Princeton.
- Proposes an extension of Martin-Löf Type Theory as a new foundation of Mathematics.
- *Informal* use of Type Theory to appeal to Mathematicians (but not only).
- Additional principles inspired by Homotopy Theory.

Type Theory 101

- Martin-Löf Type Theory: foundational system for constructive Mathematics
- Based on Curry-Howard equivalence

$$\text{proofs} : \text{proposition} = \text{programs} : \text{type}$$

- E.g. as 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 proof:
  $\mathrm{assoc} : \Pi_{i,j,k:\mathbb{N}}(i + j) + k = i + (j + k)$

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

- Theorem proving becomes functional programming!
- Basis of a number of proofs assistants / programming languages : NuPRL, LEGO, Coq, Agda, Idris, ...

Proof-relevant mathematics

- The axiom of choice (AC_∞) in Type Theory (given A, B : **Type**, $R : A \to B \to$ **Type**):

$$(\Pi a : A.\Sigma b : B.R(a,b)) \to (\Sigma f : A \to B.\Pi a : A.R(a, f(a)))$$

- AC_∞ is provable, the proof is given by

$$\mathrm{ac}\, f = (\lambda a.\mathrm{fst}(f(a)), \lambda a.\mathrm{snd}(f(a)))$$

  where fst,snd are the projections associated with Σ.
- Proofs in Type Theory can contain information.
- We call a type A *propositional* (A : **Prop**), if it contains no information.
- A propositional version of the axiom AC_{-1} is not provable and implies excluded middle.

Identity types

- Given A : **Type** and $a, b : A$ we can form

$$a =_A b : \mathbf{Type}$$

  the type of proofs that a is equal to b.
- The canonical inhabitant is

$$\mathrm{refl}(a) : a =_A a$$

  given $a : A$.
- The eliminator for equality types is

$$\mathrm{J(a)} : \Pi_{P:\Pi b:A.a=b\to\mathbf{Type}} P(a, \mathrm{refl}(a)) \to \Pi_{x:A,\alpha:a=x} P(x, \alpha)$$

  for $a : A$, with the definitional equality $J(P, p, \mathrm{refl}(a)) \equiv p$.

Groupoid structure

- Using the non-dependent version of J

$$\text{transport} : \Pi_{a,b:A}\Pi_{P:A\to\mathbf{Type}} a = b \to P(a) \to P(b)$$

we can show that $=$ is an equivalence relation:

$$\begin{aligned} \alpha^{-1} &: b = a && \text{for } \alpha : a = b \\ \alpha;\beta &: a = c && \text{for } \alpha : a = b, \beta : b = c \end{aligned}$$

- Using J we can also show:

$$\begin{aligned} &\alpha, \text{refl} = \alpha \\ &\text{refl}; \alpha = \alpha \\ &(\alpha;\beta);\gamma = \alpha;(\beta;\gamma) \\ &\alpha;\alpha^{-1} = \text{refl} \\ &\alpha^{-1};\alpha = \text{refl} \end{aligned}$$

- That is $=$ forms a groupoid.

Should equality be propositional?

- Should equality be propositional ? I.e. can we prove *Uniqueness of equality proofs* (UIP):

$$\Pi_{a,b:A}\Pi_{p,q:a=b}q = p$$

- Using J this is can be reduced to:

$$\Pi_{a:A}\Pi_{p:a=a}p = \mathrm{refl}$$

- Hofmann and Streicher showed that UIP cannot be derrived from J using a groupoid model of Type Theory.
- More recently, Voevodsky observed that UIP is also refuted by a homotopy model of Type Theory.
- Basic idea:

Types	topological spaces
$a : A$	points in the space
$a =_A b$	paths between the points

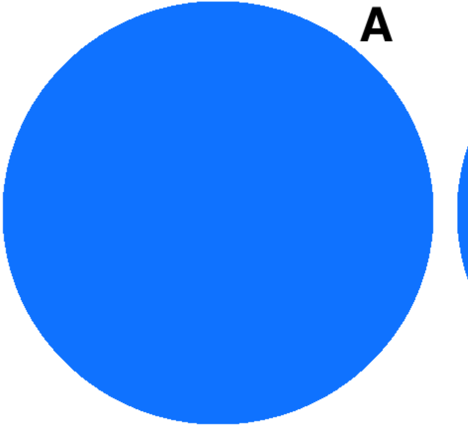
Homotopic Model

A : **Type**

$a, b : A$

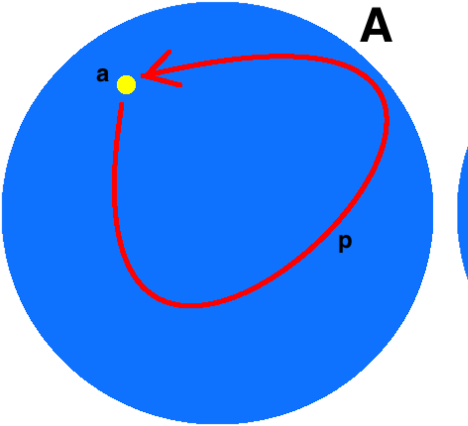
$p, q : a = b$

$H : p = q$



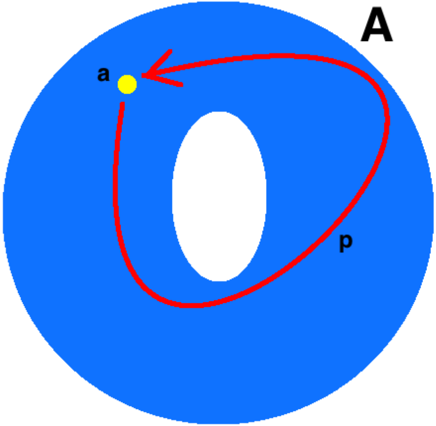
Uniqueness of Identity proofs ?

$\Pi_{p:a=_a a} p = \mathrm{refl}$?



Uniqueness of Identity proofs ?

Okay, but what now?



- While we cannot show UIP

$$\Pi_{p:a=a} p = \mathrm{refl}(a)$$

- We can show:

$$\Pi_{p:a=a}(a,p) =_{\Sigma_{x:A} x=a} (a, \mathrm{refl}(a))$$

- This follows from SCTR (singletons are contractible):

$$\Pi_{(b,p):\Sigma b:A, a=b}(a, \mathrm{refl}(a)) = (b,p)$$

  assuming $a : A$.

- On the other hand

$$J(a) : \Pi_{P:\Pi b:A.a=b \to \mathbf{Type}} P(a, \mathrm{refl}(a)) \to \Pi_{x:A,\alpha:a=x} P(x,\alpha)$$

  can be derived from transport

$$\mathrm{transport} : \Pi_{a,b:A} \Pi_{P:A \to \mathbf{Type}} a = b \to P(a) \to P(b)$$

  and SCTR.

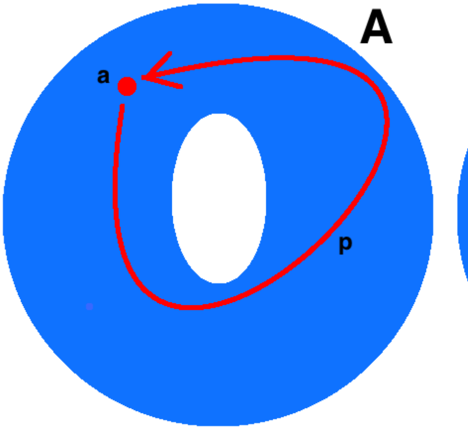
J in the Homotopy interpretation

To show

$$\Pi_{p:a=a}(a,p) =_{\Sigma_{x:A}x=a} (a, \mathrm{refl}(a))$$

we use

$$\Pi_{(b,p):\Sigma b:A, a=b}(a, \mathrm{refl}(a)) = (b, p)$$



Univalence

- Rejection of UIP is not the only thing the Homotopy interpretation is good for.
- It also suggests an additional principle: *the univalence axiom*.
- This axiom can also be justified from a purely logical perspective.
- And it is inconsistent with UIP!

What is equality of functions ?

- Given $f, g : A \to B$ we define extensional equality :

$$f \sim g :\equiv \Pi_{a:A} f(a) = g(a)$$

- We can show

$$\mathrm{app} : f = g \to f \sim g$$

- Extensionality corresponds to requiring app to be an isomorphism.
- It's inverse is *functional extensionality*.
- This can be justified by the black box view of functions.

What is the equality of types?

- Given A, B : **Type** we define isomorphism $A \sim B$ as

$$
\begin{aligned}
&f : A \to B \\
&g : B \to A \\
&\eta : \Pi_{a:A} a = g(f(a)) \\
&\epsilon : \Pi_{b:B} b = f(g(b))
\end{aligned}
$$

- In the absence of UIP we can require coherence properties which correspond to the triangle equalities of an adjunction:

$$
\begin{aligned}
&\delta : \Pi_{a:A} \mathrm{ap}(f, \eta(a)) = \epsilon(f(b)) \\
&\delta' : \Pi_{b:B} \eta(g(b)) = \mathrm{ap}(g, \epsilon(n))
\end{aligned}
$$

  where $\mathrm{ap}(f) : \Pi_{a,a':A} a = a' \to f(a) = f(a')$

- We can derive δ from δ' and vice versa, because equality is a groupoid.
- We define equivalence $A \approx B$ as given by isomorphism and δ.

What is the equality of types?

- We can show:

$$\mathrm{coe} : A = B \to A \approx B$$

- The *univalence axiom* sates that coe is an equivalence.
- This is justified by a black box view of types.
- Note that the univalence axiom implies functional extensionality.

Truncation levels

- In the absence of UIP we classify types according to the complexity of their equality types.
- We say a type A : **Type** is contractible or a -2-type, if

$$\Sigma_{a:A}\Pi_{x:A}x = a$$

  is inhabited.
- A type A : **Type** is a $n+1$-type, if all its equality types $a =_A a'$ for $a, a' : A$ an n-types.
- To show that any n-Type is also a $n+1$-type we need to show that if a type is contractible, then its equalities are contractible too.

A contractible $\to a =_A a'$ contractible

- Assume that A is contractible, that means we have

$$a_0 : A$$
$$h : \Pi_{x:A} x = a_0$$

- Now for any $a, a' : A$ we have

$$\alpha_0(a, a') : a = a'$$
$$\alpha_0(a, a') :\equiv h(a); h(a')^{-1}$$

- We need to show that

$$\Pi_{a,a':A}\Pi_{p:a=a'} p = \alpha_0(a, a')$$

- Using J it is sufficient to show

$$\Pi_{a:A}\mathrm{refl}(a) = \alpha(a, a) \equiv h(a); h(a)^{-1}$$

- Which is just one of the groupoid laws.

The Truncation Hierarchy

level		
-2	contractible types	
-1	propositions	The theorem we just proved implies that it is enough that $\Pi_{a,a':A} a = a'$.
0	sets	UIP corresponds to the assumption that every type is a set.
1	groupoids	
⋮	⋮	

Higher types

- How do we get types at higher levels?
- The 1st universe $\mathbf{Type}_0$ cannot be a set.
- Consider $\mathrm{Bool} = \mathrm{Bool}$, using univalence there are two equivalences: id, and negation.
- However, id and negation cannot be equal.
- My student Nicolai Kraus proved that the *n*th universe is an cannot be an *n*-type.

Higher inductive types

- There are other ways to obtain higher types in HoTT.
- A higher inductive type has not only constructors for elements but also for equalities.
- A simple example is S^1 : **Type**

$$\begin{aligned} \mathrm{base} &: S^1 \\ \mathrm{loop} &: \mathrm{base} = \mathrm{base} \end{aligned}$$

- S^1 homotopically behaves like the circle.
- Indeed we can embedd $i : \mathbb{Z}$ into $\phi(i) : \mathrm{base} = \mathrm{base}$:

$$\begin{aligned} i & \mapsto \mathrm{loop}^n \\ 0 & \mapsto \mathrm{refl} \\ -n & \mapsto (\mathrm{loop}^{-1})^n \end{aligned}$$

Higher inductive types

- S^1 also comes with eliminators: just consider the non-dependent eliminator:

$$\mathrm{Elim}^{S^1} : \Pi_{M:\mathbf{Type}} \Pi_{b:M} (b = b) \to S^1 \to M$$

satisfying the equalities

$$\mathrm{Elim}^{S^1}(M, b, q, \mathrm{base}) \equiv b$$
$$\mathrm{ap}(\mathrm{Elim}^{S^1}(M, b, q), \mathrm{loop}) \equiv q$$

- Using univalence we can prove $\alpha : \mathbb{Z} = \mathbb{Z}$ using the equivalence $(\lambda n.n + 1, \lambda n.n - 1, \dots)$.
- We define a family:

$$F : S^1 \to \mathbf{Type}$$
$$F = \mathrm{Elim}^{S^1}(\mathbf{Type}, \mathbb{Z}, \alpha)$$

Higher inductive types

- This allows us to define an inverse to ϕ: Given $p : \mathrm{base} = \mathrm{base}$ we can construct

$$\mathrm{transport}(F, p) \qquad : F(\mathrm{base}) \to F(\mathrm{base})$$

Note that $F(\mathrm{base}) \equiv \mathbb{Z}$. Hence we define:

$$\phi^{-1}(p) :\equiv \mathrm{transport}(F, p, 0) : \mathbb{Z}$$

- Dan Licata showed how we can use the dependent eliminator to derive that ϕ, ϕ^{-1} are inverse and hence:

$$(\mathrm{base} = \mathrm{base}) = \mathbb{Z}$$

The sphere

- The circle S^1 is 1-type (groupoid), all higher equalities are propositional.
- However, the situation is different for the sphere S^2 : **Type** which can be defined as a HIT:

$$\text{base} : S^2$$
$$\text{loop} : \text{refl}(\text{base}) = \text{refl}(\text{base})$$

- None of the higher equalities of S^2 are propositional.
- Hence S^2 has no finite truncation level.

Truncations

- We can use HITs to define truncations, which is the adjoint to the embedding.
- E.g. given A : **Type** the -1-truncation $||A||_{-1}$: **Type** (also called bracket types, squash types) is given by

$$\eta : A \to ||A||_{-1}$$
$$\mathrm{uip} : \Pi_{a,a':||A||_{-1}} a = a'$$

- Clearly, $||A||_{-1}$: **Prop** and

$$A \to P = ||A||_{-1} \to P$$

  for P : **Prop**.
- We can use this to define AC_{-1}:

$$(\Pi a : A.||\Sigma b : B.R(a,b)||_{-1}) \to (||\Sigma f : A \to B.\Pi a : A.R(a, f(a))||_{-1})$$

- AC_{-1} implies excluded middle for **Prop**, i.e. $P + \neg P$ for P : **Prop** (Diaconescu's construction).
- We can generalize this to arbitrary levels $||A||_n$ is an n-type.

Canonicity

- We have introduced additional constants proving equalities:
  - functional extensionality
  - univalence
  - equality constructors for HITs
- While we have introduced some definitional equality, they are insufficient to guarantuee *canonicity*:
  All closed terms of type $\mathbb{N}$ are definitionally equal (indeed reducible) to a numeral.
- While we (with Conor McBride and Wouter Swierstra) have developed an approach to solve the problem for functional extensionality (*Observational Type Theory*), this relies on a strong form of *UIP*.

Constructive models

- The homotopy interpretation of type theory uses Kan Fibrations in simplicial sets (the simplicial set model).
- This construction relies essentially on classical principles.
- Thierry Coquand has suggested a modification of this construction using *semi-simplicial* sets.
- This models only weak Type Theory (i.e. no conversion under λ).
- An alternative would be to directly use weak ω-groupoids.
- It seems likely that an answer to this question would also provide a solution to the canonicity problem.

Why should we care?

- Structures can be 1st class citizens.
  (equality is isomorphism)
- HITs provide better ways to define the Cauchy Reals and the constructible hierarchy (avoiding choice).
- Quotient containers (like multisets) become ordinary containers.

The HoTT people