Homotopy Type Theory

Logic, Sets and n-Types

Matteo Ferrari and Sam Adam-Day

11am, Wednesday 4^{th} October 2017

1 Introduction on Types

1.1 Notation

Given a type X and $x, y\colon X$, for ease of notation we will use $x =_X y$ (or simply $x = y$) for $\mathsf{Id}_X(x, y)$.

1.2 Extensionality

All we can say within HTT is that exists a function such that

$$\mathsf{happly} : (f = g) \to \prod_{x:A} (f(x) = g(x))$$

Therefore we use the axiom "function extensionality" to express the other direction of the implication.

$$\mathsf{funext} : \prod_{x:A} (f(x) = g(x)) \to (f = g)$$

1.3 Equivalence

A function $f : A \to B$ is an equivalence if and only if there exist a $g : A \to B$ and a $h : B \to A$ such that $f \circ g \sim id_B$ and $g \circ f \sim id_A$.

If there exists such a function the types A and B are said to be Equivalent

$$(A \simeq B) \stackrel{def}{=} \sum_{f:A\to B} \left(\left(\sum_{g:B\to A} (f \circ g \sim id_B) \right) \times \left(\sum_{h:A\to B} (h \circ f \sim id_A) \right) \right)$$

2 Logic and Sets as Types

2.1 Homotopic Sets

We want to be able to describe when a type behave like a Set. we call these objects "Homotopic sets".

What we ask is that any two parallel path within elements of the Type are equal.

$$\mathsf{isSet}(A) \stackrel{def}{=} \prod_{(x,y:A)} \prod_{(p,q:x=y)} (p =_{x=y} q)$$

Some example of Sets are 1, 0 and $\mathbb{N}$.

2.2 Homotopic Propositions

From the interpretation of Types as proposition and their terms as proofs or witnesses we know that types can contain more than one information.

Namely it contains as many information as many different terms it has. What we want to do in order to obtain a more traditional logic is to restrict our attention to those types that have either one or no terms.

We call these types Homotopic Propositions, and we characterize them by saying that all the terms are the same:

$$\mathsf{isProp}(A) \stackrel{def}{=} \prod_{(x,y:A)} (x =_A y)$$

From this fact we can conclude that if $\mathsf{isProp}(A)$ and $x_0 : A$ then $A \simeq 1$.

Lemma 1. *Every Homotopic Proposition is a set.*

2.3 Excluded Middle vs Univalence Axiom

A theorem we can prove within this Type Theory is that for types A and B there is a function

$$idtoeqv : (A = B) \to (A \simeq B)$$

In 2.10 the *Univalence Axiom* is introduced, so that the identity within types is equivalent to their equivalence, that is to say:

$$(A = B) \simeq (A \simeq B)$$

Having this axiom allows as to have only Univalent models for the theory.

An interesting fact is that this axiom is incoherent with the Law of Excluded Middle (LEM), which states that for every type A

$$LEM : \neg(\neg A) \to A$$

This happens for the same reason why we decided to introduce the notion of isProp.

What we can do to use LEMis adding a condition. For every type A, if isProp(A), then the LEM holds for A

$$LEM \upharpoonright \mathsf{isProp}(A) \to (\neg(\neg A) \to A)$$

Moreover we want to distinguish Types where LEM holds. We call such them "*decidable*" and we say that

- A type is **decidable** if $A + \neg A$
- A has a **decidable equality** if $a, b : A$ and $(a = b) + \neg(a = b)$

The second Statement implies that all sets have decidable equation, we are going to prove that later.

3 Propositional Truncation

To work better within Classical Logic we say that for any type A there is a type $\|A\|$, which is the truncation of A. Its rules are

- for any $a : A$ we have $|a| : \|A\|$
- For any $x, y : A$, we have $x = y$

Using truncation we can write all the traditional operators:

$$\begin{aligned}
\top &\stackrel{def}{=} 1 \\
\bot &\stackrel{def}{=} 0 \\
P \wedge Q &\stackrel{def}{=} P \times Q \\
P \vee Q &\stackrel{def}{=} \|P + Q\| \\
P \Rightarrow Q &\stackrel{def}{=} P \to Q \\
P \Leftrightarrow Q &\stackrel{def}{=} P = Q \\
\neg P &\stackrel{def}{=} P \to 0 \\
\forall (x : A).P(x) &\stackrel{def}{=} \prod_{x} : AP(x) \\
\exists (x : A).P(x) &\stackrel{def}{=} \left\| \sum_{x} : AP(x) \right\|
\end{aligned}$$

4 Contractibility

We say that a type A is *contractible* if there is a $a : A$, called the center of contraction, such that $a = x$ for all $x : A$.

$$\mathsf{isContr}(A) \stackrel{def}{=} \sum_{(a:A)} \prod_{(x:A)} (a = x)$$

From this definitions we can derive the following lemma:
A type A is a mere proposition if and only if for all $x, y : A$, the type $x =_A y$ is contractible.

4.1 Retractions

A retraction is a function $r : A \to B$ such that exists a function $s : B \to A$, called section, and an homotopy $\epsilon : \prod_{y:B}(r(s(y)) = y)$. In this case we say that B is a retract of A-

Lemma 2. *If B is a retract of A, and A is contractible, them so is B*

Proof. Let $a_0 : A$ be the centre of the contraction in A. We claim that if $b_0 \stackrel{def}{=} r(a_0) : B$ then b_0 is a centre of contraction for B. Take a $b : B$, we need to show that there is a path $b = b_0$. We know that $\epsilon_b : r(s(b)) = b$ and $\mathsf{contr}_{s(b)} : s(b) = a_0$ thus $\epsilon_b^{-1} \circ r(\mathsf{contr}_{s(b)}) : b = r(a_0) = b_0$ □

5 n-Types

5.1 Definition and First Results

Contractible types, propositions and sets form part of a larger hierarchy of what are called *n-types*. The idea is that n-types contain no non-trivial paths above level n. From the homotopy point of view, this corresponds to having no interesting homotopical structure above dimension n. To begin with, we first define a predicate which singles out the n-types for each $n \in \mathbb{N}$.

Definition 3 (n-type)**.** For each $n \in \mathbb{N}$ and type X, we define the predicate $\mathsf{is}\text{-}n\text{-}\mathsf{type}(X)$ by recursion:

$$\begin{aligned}
\mathsf{is}\text{-}0\text{-}\mathsf{type}(X) &:= \mathsf{isContr}(X) \\
\mathsf{is}\text{-}n\text{-}\mathsf{type}(X) &:= \prod_{(x,y\,:\,X)} \mathsf{is}\text{-}(n-1)\text{-}\mathsf{type}(x =_X y)
\end{aligned}$$

A type X is an *n-type* if $\mathsf{is}\text{-}n\text{-}\mathsf{type}(X)$ is inhabited.

It is easy to see that contractible types are -2-types, propositions are -1-types and sets are 0-types. Just as in the case of propositions and sets, the property of being an n-type is quite stable. Indeed, it is preserved by many of the operations we have encountered so far.

Theorem 4. *Let X be an n-type, and let $r\colon X \to Y$ be a retraction. Then Y is an n-type.*

Proof. We prove this by induction on n. The base case $n = -2$ is done by Lemma 2. So assume that the retraction of every n-type is an n-type, and that X is an $(n+1)$-type. Let s be a section of r, and $\epsilon\colon \prod_{(y\colon Y)}(r(s(y)) = y)$ be a homotopy.[1] Take $y, z\colon Y$. We must show that $y =_Y z$ is an n-type.

We first note that since X is an $(n+1)$-type, $s(y) =_X s(z)$ is an n-type. We will show that $y = z$ is a retraction of $s(y) =_X s(z)$, then the result will follow by induction. We define $t\colon (s(y) = s(z)) \to (y = z)$ by:

$$t(p) := \epsilon(y)^{-1} \centerdot \mathsf{ap}_r(q) \centerdot \epsilon(z)$$

The section of this retraction will be:

$$\mathsf{ap}_s\colon (y = z) \to (s(y) = s(z))$$

To show that t is indeed a retraction, we need to exhibit its homotopy. This is equivalent to showing that for every $p\colon y = z$:

$$t(\mathsf{ap}_s(p)) = \epsilon(y)^{-1} \centerdot \mathsf{ap}_r(\mathsf{ap}_s(p)) \centerdot \epsilon(z) = p$$

By path-induction (the elimination rule for identity-types), it suffices to prove this when $z = y$ and $p = \mathsf{refl}_y$. But then by definition $\mathsf{ap}_s(\mathsf{refl}_y) = \mathsf{refl}_{s(y)}$ and $\mathsf{ap}_r(\mathsf{refl}_{s(y)}) = \mathsf{refl}_{r(s(y))}$, so by the rules of path composition, we are done. □

Corollary 5. *Let X be an n-type, and suppose $X \simeq Y$ then Y is an n-type.*

Theorem 6. *Let A be an n-type and let $B(x) \mid x\colon A$ be a family of n-types. Then $\sum_{(x\colon A)} B(x)$ is an n-type.*

For this, we need the following basic lemma about paths in Σ-types. The proof involves invoking Σ-induction and path-induction an appropriate number of times, and can be found at [Uni13, p. 84].

Lemma 7. *Let $B(x) \mid x\colon A$ be a type family over A, and take $\mathsf{pair}_\Sigma(a_1, b_1), \mathsf{pair}_\Sigma(a_2, b_2)\colon \sum_{(x\colon A)} B(x)$. Then:*

$$(\mathsf{pair}_\Sigma(a_1, b_1) = \mathsf{pair}_\Sigma(a_2, b_2)) \simeq \sum_{(p : a_1 = a_2)} p_*(b_1) = b_2$$

Proof of Theorem 6. Again, we proceed by induction on n.

For the base case $n = -2$ we need to show that $\sum_{(x\colon A)} B(x)$ is contractible if A and $B(x) \mid x\colon A$ are. In this case there is $a_0\colon A$ which is the centre of contraction. Since $B(a_0)$ is contractible, it has a centre $b_0\colon B(a_0)$. Consider $\mathsf{pair}_\Sigma(a_0, b_0)\colon \sum_{(x\colon A)} B(x)$, which we claim is the centre of contraction for $\sum_{(x\colon A)} B(x)$. Take any $\mathsf{pair}_\Sigma(a, b)\colon \sum_{(x\colon A)} B(x)$ (by Σ-induction (the elimination rule for Σ-types) these are the only kind of elements we need to consider). By contractibility, there is a path $p\colon a = a_0$ and a path $q\colon p_*(b) = b_0$. Then by Lemma 7, this gives a path $\mathsf{pair}_\Sigma(a, b) = \mathsf{pair}_\Sigma(a_0, b_0)$.

For the inductive step we want to show that $\sum_{(x\colon A)} B(x)$ is an $(n+1)$-type if A and $B(x) \mid x\colon A$ are. So assume this. Take $\mathsf{pair}_\Sigma(a_1, b_1), \mathsf{pair}_\Sigma(a_2, b_2)\colon \sum_{(x\colon A)} B(x)$. By Lemma 7:

$$(\mathsf{pair}_\Sigma(a_1, b_1) = \mathsf{pair}_\Sigma(a_2, b_2)) \simeq \sum_{(p : a_1 = a_2)} (p_*(b_1) = b_2)$$

Now each $p_*(b_1) = b_2$ is an n-type (since $B(a_2)$ is an $(n+1)$-type) and so is $a_1 = a_2$. Hence by induction hypothesis, the right hand side is an n-type, and thus by Corollary 5, the left hand side is too. □

If we also have **Functional Extensionality** then we can prove that n-types are stable under taking Π-types.

[1] This means nothing more than that ϵ exists.

Theorem 8. *Let A be an n-type and let $B(x) \mid x\colon A$ be a family of n-types. Then $\prod_{(x\colon A)} B(x)$ is an n-type.*

Proof. This is by induction on n.

For the base case $n = -2$ we need to show that $\prod_{(x\colon A)} B(x)$ is contractible if A and $B(x) \mid x\colon A$ are. In this case, we can define a function $f_0\colon \prod_{(x\colon A)} B(x)$ which sends each $x\colon A$ to the centre of contraction of $B(x)$. Take any $f\colon \prod_{(x\colon A)} B(x)$. For every $x\colon A$, we have by contractibility of $B(x)$ a path $f(x) = f_0(x)$. Then by **Functional Extensionality** this gives a path $f = f_0$.[2]

For the inductive step we want to show that $\prod_{(x\colon A)} B(x)$ is an $(n+1)$-type if A and $B(x) \mid x\colon A$ are. So assume the latter. Take $f, g\colon \prod_{(x\colon A)} B(x)$; we want to show that $f = g$ is an n-type. By **Functional Extensionality**:

$$f = g \simeq \prod_{(x\colon A)} (f(x) = g(x))$$

Now each $f(x) = g(x)$ is an n-type. Thus, as before, $f = g$ is an n-type. □

From our intuition about n-types as being those types with no non-trivial paths above level n, we would expect the hierarchy to be cumulative. This does indeed turn out to be the case.

Theorem 9 (Cumulative Hierarchy of Types)**.** *If X is an n-type then X is an $(n+1)$-type.*

Proof. This is by induction on n.

For the base case, we need to show that contractible types have contractible identity-types. So let X be contractible with centre x_0. For every $x\colon X$, we have a path $\mathsf{contr}_x\colon x = x_0$. Take $y, x\colon X$. We will show that $x = y$ is contractible, with centre $\mathsf{contr}_x \cdot \mathsf{contr}_y^{-1}\colon x = y$. We want to show that for every $p\colon x = y$ we have $p = \mathsf{contr}_x \cdot \mathsf{contr}_y^{-1}$. By path-induction (the elimination rule for identity-types), it suffices to show this for $y = x$ and $p = \mathsf{refl}_x$. But this is just $\mathsf{refl}_x \cdot \mathsf{refl}_x^{-1} = \mathsf{refl}_x$ which follows by definition.

For the induction case, we need to show for X an $(n+1)$-type and $x, y\colon X$, that $x = y$ is a also an $(n+1)$-type. But this follows by induction hypothesis since $x = y$ is an n-type. □

5.2 Hedberg's Theorem

In this subsection, we explore several properties that are equivalent to being a set, and more generally an n-type. We first look at an axiom, due to Streicher, that gives an alternative characterisation of sets.

Definition 10 (Axiom K)**.** A type X satisfies ***Axiom K*** if:

$$\prod_{(x\colon X)} \prod_{(p\colon x=x)} (p = \mathsf{refl}_x)$$

In other words, for every $x\colon X$ and $p\colon x =_X x$, we have that p is identical with refl_x.

Proposition 11. *A type X is a set if and only if it satisfies **Axiom K**.*

Proof. Note that if X is a set, then for any $x, y\colon X$ and $p, q\colon x = y$ we have $p = q$, so in particular $p = \mathsf{refl}_x$ for every path $p\colon x = x$. Conversely, assume that X satisfies **Axiom K**. We want to show that:

$$\prod_{(x,y\colon X)} \prod_{(p,q\colon x=y)} p = q$$

By path-induction (the elimination rule for identity-types), we may assume that $y = x$ and $p = \mathsf{refl}_x$. So we need to show that $\prod_{(q\colon x=x)} (\mathsf{refl}_x = q)$. But this is exactly what **Axiom K** gives us. □

We saw the property of *decidable equality* for types. We aim to show now that the sets are precisely the types which have decidable equality. But first we give another characterisation of sets.

[2] Notice that we didn't actually use that A is contractible.

Theorem 12. *Let X be a type, and let R be a reflexive* homotopic *relation on X (that is, for $x, y\colon X$ we have a type $R(x,y)$ such that $\prod_{(x\colon X)} R(x,x)$ and $\prod_{(x,y\colon X)} \mathsf{isProp}(R(x,y))$) such that $\prod_{(x,y\colon X)}(R(x,y) \to x =_X y)$. Then X is a set.*

Before we prove this, we need an a handy little lemma about the commutativity of transports.

Lemma 13. *Let $A(x), B(x) \mid x\colon X$ be families of types, $x, y\colon X$, $p\colon x = y$ and take functions $f\colon A(x) \to B(x)$ and $g\colon A(y) \to B(y)$. Then we have the following equivalence*[3]*:*

$$(p_*(f) = g) \simeq \prod_{(a\colon A(x))} (p_*(f(a)) = g(p_*(a)))$$

Proof. By path-induction (the elimination rule for identity-types), it suffices to assume that $x = y$ and $p = \mathsf{refl}_x$. But for any z, we have $\mathsf{refl}_*(z) = z$, so the the equivalence reduces to the statement of **Functional Extensionality**. □

Proof of Theorem 12. Let $\rho\colon \prod_{(x\colon X)} R(x,x)$ witness the reflexivity of R, and let:

$$f\colon \prod_{(x,y\colon X)} (R(x,y) \to x =_X y)$$

Take $x\colon X$ and $p\colon x =_X x$. By Proposition 11, it suffices to show that $p = \mathsf{refl}_x$. Now consider:

$$f(x)\colon \prod_{(y\colon X)} (R(x,y) \to x =_X y)$$

Then p defines a transport:

$$p_*\colon (R(x,x) \to x =_X x) \to (R(x,x) \to x =_X x)$$

So we can look at $f(x)(x) = f(x,x)$ and we get a path $\mathsf{apd}_{f(x)}(p)\colon p_*(f(x,x)) = f(x,x)$. Now, $f(x,x)$ is a function $R(x,x) \to x =_X x$, so by the equivalence in Lemma 13 this gives us, for every $r\colon R(x,x)$ a path:

$$p_*(f(x,x,r)) = f(x,x,p_*(r)) \tag{1}$$

Here is where we use that R is a homotopic relation: we have $r, p_*(r)\colon R(x,x)$, so we must have $r = p_*(r)$. Moreover, the left hand side of Eq. (1) is the transport in an identity type, so it corresponds to path concatenation (that this is true can be proved easily using path-induction). Thus we get:

$$f(x,x,r) \centerdot p = f(x,x,r)$$

By applying $f(x,x,r)^{-1}$ to both sides, we get that $p = \mathsf{refl}_x$, as required. □

Using **Functional Extensionality**, we can then derive the following.

Corollary 14. *If X is a type such that for every $x, y\colon X$ we have $\neg\neg(x =_X y) \to (x =_X y)$ then X is a set.*

Proof. By Theorem 12, it suffices to show that $\neg\neg(x =_X y)\,[= (\neg(x =_X y) \to \mathbf{0})]$ is a mere relation. So take $f, g\colon (\neg(x =_X y) \to \mathbf{0})$. To show that $f = g$, by **Functional Extensionality** it suffices to show that $\prod_{(d\colon \neg(x =_X y))} f(d) = g(d)$. So take $d\colon \neg(x =_X y)$. Then $f(d)\colon \mathbf{0}$ and $g(d)\colon \mathbf{0}$, so by $\mathbf{0}$-induction (the elimination rule for $\mathbf{0}$), we have that $f(d) = g(d)$, as required. □

We are now very close to our goal, but we need to translate the condition $\neg\neg(x =_X y) \to (x =_X y)$ into $(x =_X y) + \neg(x =_X y)$. This is what the following lemma allows us to do.

Lemma 15. *For every type A we have that $(A + \neg A) \to (\neg\neg A \to A)$.*

[3]Note that the p_*s here are different. Different type families give rise to different transports, but we usually notate them in the same way. It is possible to determine exactly which transport we're talking about by examining the types.

Proof. We want to construct a function $f\colon (A + \neg A) \to (\neg\neg A \to A)$. By disjunction-induction (the elimination rule for sum-types), it suffices to give the values of f on $\mathsf{inc_l}(a)$ for $a\colon A$ and $\mathsf{inc_r}(b)$ for $b\colon \neg A$. We map the former to the constant function $\mathsf{c}_a\colon \neg\neg A \to A$ which maps everything to a. For the latter, we note that we want $f(\mathsf{inc_r}(b))\colon (\neg\neg A \to A)$. Given an element of $c\colon \neg\neg A = (\neg A \to \mathbf{0})$, we have that $c(b)\colon \mathbf{0}$, and so we can use $\mathbf{0}$-induction (the elimination rule for $\mathbf{0}$) to obtain an element of A as required. □

Theorem 16 (Hedberg's Theorem)**.** *If X is a type with decidable equality, then X is a set.*

Proof. If X has decidable equality, then by Lemma 15, for every $x, y\colon X$ we have $\neg\neg(x =_X y) \to (x =_X y)$, and so by Corollary 14, X is a set. □

This gives us a nice characterisation of sets (0-types), but ideally we'd like to generalise it to get a characterisation of all n-types. However, the condition of decidable equality seems quite specific to sets, and doesn't generalise so readily. **Axiom K**, on the other hand, lends itself quite well to generalisation, as we will see shortly.

Definition 17 (Pointed type)**.** A *pointed type* is a pair (X, x) where X is a type and $x\colon X$.

All properties of normal types extend naturally to pointed types: if $P(-)$ is a property of types then $P((X, x))$ is the same as $P(X)$.

Definition 18 (Loop Space)**.** The *loop space* of a pointed type (X, x) is:

$$\Omega(X, x) := (x =_X x, \mathsf{refl}_x)$$

Definition 19 (n-fold iterated loop space)**.** Given a type X and $x\colon X$, the *n-fold iterated loop space* of (X, x) is defined recursively:

$$\Omega^1(X, x) := \Omega(X, x)$$
$$\Omega^{n+1}(X, x) := \Omega(\Omega^n(X, x))$$

Definition 20 (Axiom K$_n$)**.** A type X satisfies ***Axiom K$_n$*** if:

$$\prod_{(x\colon X)} \mathsf{isContr}(\Omega^{n+1}(X, x))$$

And so, we can state the last theorem, which gives a general characterisation of n-types for every n. For a proof see [Uni13, p.228].

Theorem 21. *A type X is an n-type if and only if it satisfies **Axiom K$_n$**.*

6 Exercises

(a) We can can define m-ary product types $\mathsf{Product}_m(A_1, \ldots, A_m)$ in a similar way to binary product types $A_1 \times A_2$. The canonical elements of $\mathsf{Product}_m(A_1, \ldots, A_m)$ will be $\mathsf{tuple}_m(a_1, \ldots, a_m)$ for $a_1\colon A_1, \ldots, a_m\colon A_m$. Formulate the Formation, Introduction, Elimination and Computation rules for m-ary product types, and starting from these to show that if $A_1, \ldots, A_m$ are n-types then so is $\mathsf{Product}_m(A_1, \ldots, A_m)$.

(b) (Bonus challenge: only for fun!) The notion of a W-type is defined in [Uni13, p.154]. Show that if $A, B(x) \mid x\colon A$ are n-types for $n \geqslant -1$, then so is $\mathsf{W}_{(x\colon A)} B(x)$.

References

[Uni13] The Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations of Mathematics*. `https://homotopytypetheory.org/book`, Institute for Advanced Study, 2013.