Synthetic homotopy theory and higher inductive types

Ulrik Buchholtz, CMU

FoMUS, Bielefeld, July, 2016

Outline

1. Synthetic Homotopy Theory
2. Truncation
3. The fundamental group of the circle
4. Higher inductive types
5. More homotopy theory
6. More HITs

As a first approximation consider the analogy:

synthetic geometry	:	analytic geometry
synthetic homotopy theory	:	classical homotopy theory

As a first approximation consider the analogy:

synthetic geometry	:	analytic geometry
synthetic homotopy theory	:	classical homotopy theory

Spaces, points, paths, homotopies are *basic notions* given directly in terms of the identity type. Sometimes this leads to *new proofs*.

It's not a new idea to consider abstract homotopy theory: this goes back at least 50 years:

- Edgar Brown's abstract homotopy theory (1965)
- Quillen model categories (1967)
- Kenneth Brown's fibration categories (1973)
- Waldhausen categories (1983)
- Grothendieck's derivators (1990)
- etc.

In this way, HoTT is part of a long tradition in homotopy theory.

Homotopy type theory provides another way to do abstract homotopy theory. It feels even more synthetic because the framework ensures that everything is invariant under equivalence. *This is new.*

Some have complained about the term *synthetic homotopy theory* for this reason. Perhaps better would be *type-theoretic homotopy theory* or *univalent homotopy theory*.

What we'll do in this workshop is to see how this works for a few basic results (kind of like browsing through Book 1 of Euclid's *Elements*), and along the way we'll be acquainted with a key tool: higher inductive types.

Outline

Let us recall the basic dictionary of homotopy type theory:

A	type	space	∞-groupoid
$a : A$	term	point	object
$p : a =_A b$	identification	path	arrow
$f : A \to B$	function	continuous map	homomorphism
$C : A \to \mathrm{Type}$	dep. type	fibration	fibration
$f : \Pi(x : A)C(x)$	dep. function	section	section
$s : \Sigma(x : A)C(x)$	dep. pair	point in total space	

Truncation levels

Recall that a type A may be *truncated* at a finite level:

Level	Predicate	Name
-2	$\mathrm{isContr}(A) := \Sigma(x : A)\Pi(y : A)(x = y)$	contractible
-1	$\mathrm{isProp}(A) := \Pi(x, y : A)\mathrm{isContr}(x = y)$	proposition
0	$\mathrm{isSet}(A) := \Pi(x, y : A)\mathrm{isProp}(x = y)$	set
1	$\mathrm{isGpd}(A) := \Pi(x, y : A)\mathrm{isSet}(x = y)$	groupoid
…	…	…
$n+1$	$\mathrm{isTrunc}_{n+1}(A) := \Pi(x, y : A)\mathrm{isTrunc}_n(x = y)$	$n+1$-groupoid

These predicates are themselves propositions and we have the equivalence

$$\mathrm{isProp}(A) \simeq \Pi(x, y : A)(x = y)$$

Recall that the identity types $a = b$, for $a, b : A$ can be thought of as inductively defined by an element $\mathrm{idp} : a = a$.

The corresponding induction principle is called *path induction*: If $C : \Pi(x : A)((a = x) \to \mathrm{Type})$, and we have some $c : C(a, \mathrm{idp})$, then we have a section

$$J(C, c) : \Pi(x : A)(p : a = x).\ C(x, p)$$

We have $J(C, c)(a, \mathrm{idp}) = c$.

A fundamental example of a fibration is the *path fibration*: Given a type A and a point $a : A$, have $P : A \to \mathrm{Type}$ with $P(x) := (a = x)$.

Exercise: prove $\mathrm{isContr}(\Sigma(x : A)P(x))$.

A fundamental example of a fibration is the *path fibration*: Given a type A and a point $a : A$, have $P : A \to \mathrm{Type}$ with $P(x) := (a = x)$.

Exercise: prove $\mathrm{isContr}(\Sigma(x : A)P(x))$.

Center of contraction: $\langle a, \mathrm{idp} \rangle$.

Use path induction to show $\langle a, \mathrm{idp} \rangle = \langle b, p \rangle$ for any $b : A$, $p : a = b$.

We need to know that there is a way for any A and $n \geq -2$ to make a "best approximation" $\|A\|_n$ of A that is an n-truncated type. It comes with a map $|-|_n : A \to \|A\|_n$.

The *universal property* of the truncation is this: If B is any n-truncated type, then the following map is an equivalence:

$$(\|A\|_n \to B) \to (A \to B)$$
$$g \mapsto g \circ |-|_n$$

Later we'll see how to construct $\|A\|_n$ as a higher inductive type.

The fundamental group

The *fundamental group* of a pointed space A, written $\pi_1(A, \star)$, has as underlying set

$$\pi_1(A, \star) :\equiv \|\Omega(A, \star)\|_0,$$

where $\Omega(A, \star) :\equiv (\star =_A \star)$ is the *loop space*.

The group operation is path concatenation, inverses are given by path reversal, and the neutral element is the reflexivity path.

We're going to calculate the fundamental group of the circle $\mathbb{S}^1$.

We introduce the circle $\mathbb{S}^1$ as motivating example of a higher inductive type. Recall two examples of ordinary inductive types:

The booleans $\mathbb{B}$, generated by:

- Two points $\mathsf{true}, \mathsf{false} : \mathbb{B}$.

The natural numbers $\mathbb{N}$, generated by:

- A point $0 : \mathbb{N}$, and
- a function $\mathsf{S} : \mathbb{N} \to \mathbb{N}$.

The circle is generated by:

- A point $\mathsf{base} : \mathbb{S}^1$, and
- A path $\mathsf{loop} : \mathsf{base} =_{\mathbb{S}^1} \mathsf{base}$.

The circle continued

Let's see what the elimination principle for the circle should be:

- We define $f : \mathbb{B} \to Z$ by recursion by giving $f(\mathrm{true}) : Z$ and $f(\mathrm{false}) : Z$.
- We define $f : \mathbb{N} \to Z$ by recursion by giving $f(0) : Z$ and for each $n : \mathbb{N}$, $f(\mathsf{S}\, n) : Z$, assuming the value $f(n) : Z$ known.
- We define $f : \mathbb{S}^1 \to Z$ by recursion by giving $f(\mathrm{base}) : Z$ and $\mathrm{ap}_f(\mathrm{loop})$, a path of type $f(\mathrm{base}) =_Z f(\mathrm{base})$.

Induction principles not only tell us how to construction functions out of an inductive type, they also tell us more generally how to construct sections of fibrations over them.

- We define a section $f : \Pi(b : \mathbb{B})P(b)$ by induction by giving $f(\mathrm{true}) : P(\mathrm{true})$ and $f(\mathrm{false}) : P(\mathrm{false})$.
- We define $f : \Pi(n : \mathbb{N})P(n)$ by induction by giving $f(0) : P(0)$ and for each $n : \mathbb{N}$, $f(\mathrm{S}\,n) : P(\mathrm{S}\,n)$, assuming the value $f(n) : P(n)$ known.
- We define $f : \Pi(x : \mathbb{S}^1)P(x)$ by induction by giving $f(\mathrm{base}) : P(\mathrm{base})$ and $\mathrm{apd}_f(\mathrm{loop})$, a dependent path (a *pathover*) of type $f(\mathrm{base}) =^P_{\mathrm{loop}} f(\mathrm{base})$.

We now prove that $\pi_1(\mathbb{S}^1) \simeq \mathbb{Z}$. Define $\mathrm{code} : \mathbb{S}^1 \to \mathrm{Type}$ by recursion:

$$\mathrm{code}(\mathrm{base}) :\equiv \mathbb{Z}$$
$$\mathrm{ap}_{\mathrm{code}}(\mathrm{loop}) := \mathrm{ua}(\mathrm{succ})$$

(Note the use of univalence!)

We then give a fiber-wise equivalence
$\Pi(x : \mathbb{S}^1)(\mathrm{code}(x) \simeq (\mathrm{base} = x))$.

It's possible to emulate the traditional proof in HoTT, proving that a fiber-wise map gives an equivalence on total spaces:
$\Sigma(x : \mathbb{S}^1)\mathrm{code}(x) \simeq \Sigma(x : \mathbb{S}^1)(\mathrm{base} = x) \simeq 1$.

Here is a more type-theoretic proof: First define $\mathrm{encode} : \Pi(x : \mathbb{S}^1)((\mathrm{base} = x) \to \mathrm{code}(x))$ by

$$\mathrm{encode}(x) :\equiv \lambda p : \mathrm{base} = x.\ \mathrm{transport}^{\mathrm{code}}(p, 0)$$

and $\mathrm{decode} : \Pi(x : \mathbb{S}^1)(\mathrm{code}(x) \to (\mathrm{base} = x))$ by circle induction:

$$\mathrm{decode}(\mathrm{base}) :\equiv \lambda z : \mathbb{Z}.\ \mathrm{loop}^z$$
$$\mathrm{apd}_{\mathrm{decode}}(\mathrm{loop}) :\equiv ?$$

Here is a more type-theoretic proof: First define $\mathrm{encode} : \Pi(x : \mathbb{S}^1)((\mathrm{base} = x) \to \mathrm{code}(x))$ by

$$\mathrm{encode}(x) :\equiv \lambda p : \mathrm{base} = x.\ \mathrm{transport}^{\mathrm{code}}(p, 0)$$

and $\mathrm{decode} : \Pi(x : \mathbb{S}^1)(\mathrm{code}(x) \to (\mathrm{base} = x))$ by circle induction:

$$\mathrm{decode}(\mathrm{base}) :\equiv \lambda z : \mathbb{Z}.\ \mathrm{loop}^z$$
$$\mathrm{apd}_{\mathrm{decode}}(\mathrm{loop}) :\equiv ?$$

Need lemma: if $B, C : A \to \mathrm{Type}$, $p : a =_A a'$, $f : B\,a \to C\,a$, $g : B\,a' \to C\,a'$ and $\Pi(b : B\,a)(f\,b =^C_p g\,(\mathrm{transport}^B(p, b)))$, then $f =^{\lambda x:A.\,B\,x \to C\,x}_p g$.

Need to show for all $z : \mathbb{Z}$: $\mathsf{loop}^z =^{\lambda x:\mathbb{S}^1.\,\mathsf{base}=x}_{\mathsf{loop}} \mathsf{loop}^{z+1}$.

Need another lemma: if $f, g : A \to B$, $p : a =_A a'$, $q : f\,a = g\,a$, $r : f\,a' = g\,a'$ and s fills a square:

$$\begin{array}{ccc} f\,a & \overset{q}{=\!=\!=} & g\,a \\ {\scriptstyle \mathsf{ap}_f\,p}\Big\| & & \Big\|{\scriptstyle \mathsf{ap}_g\,p} \\ f\,a' & \underset{r}{=\!=\!=} & g\,a', \end{array}$$

then $q =^{\lambda x:A.\,f\,x=g\,x}_{p} r$.

The encode-decode method, cont.

Still need to show for all $z : \mathbb{Z}$: $\mathrm{loop}^z =^{\lambda x:\mathbb{S}^1.\,\mathrm{base}=x}_{\mathrm{loop}} \mathrm{loop}^{z+1}$. By the lemma, it suffices to fill the square:

$$\begin{array}{ccc} \mathrm{base} & \overset{\mathrm{loop}^z}{=\!=\!=} & \mathrm{base} \\ \mathrm{idp}\Big\| & & \Big\|\mathrm{loop} \\ \mathrm{base} & \underset{\mathrm{loop}^{z+1}}{=\!=\!=} & \mathrm{base} \end{array}$$

This we can easily do. Thus we can *define* the function $\mathrm{decode} : \Pi(x : \mathbb{S}^1)(\mathrm{code}(x) \to (\mathrm{base} = x))$.

It remains to show that encode and decode are fiberwise mutually inverse.

Lemma 1: For all $x : \mathbb{S}^1$ and $p : \mathrm{base} = x$, $\mathrm{decode}(x)(\mathrm{encode}(x)(p)) = p$.
Proof by path induction: $\mathrm{decode}(\mathrm{base})(\mathrm{encode}(\mathrm{base})(\mathrm{idp})) = \mathrm{decode}(\mathrm{base})(0) = \mathrm{loop}^0 = \mathrm{idp}$. $\square$

Lemma 2: For all $x : \mathbb{S}^1$ and $z : \mathrm{code}(x)$, $\mathrm{encode}(x)(\mathrm{decode}(x)(z)) = z$.
Proof by circle induction: Suffices (since $\mathbb{Z}$ is a set) to do the base case: $\mathrm{encode}(\mathrm{base})(\mathrm{decode}(\mathrm{base})(z)) = \mathrm{transport}^{\mathrm{code}}(\mathrm{loop}^z, 0) = z$ (by induction on $z : \mathbb{Z}$). $\square$.

This completes the proof that for $x : \mathbb{S}^1$, $\mathrm{code}(x) \simeq (\mathrm{base} = x)$. In particular, $\mathbb{Z} \simeq (\mathrm{base} = \mathrm{base})$. Hence also $\mathbb{Z} \simeq \pi_1(\mathbb{S}^1)$. $\square$

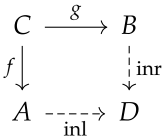
Higher inductive types (HITs)

Having seen an example of a higher inductive type and how it used in synthetic homotopy theory, let us look at more higher inductive types, and make some general remarks.

- A higher inductive type includes ordinary point constructors, but also *path constructors* (with given source and target), and possibly *higher* path constructors.
- Path constructors give *new* elements of identity types (just like univalence does).
- The resulting type is a freely generated ∞-groupoid. But by including higher path constructors we can impose “relations” (will return to this).
- If a path constructor has an argument of type A, then n-paths in A give rise to $n+1$-paths in the generated type.
- *I should note that we don’t have a general schema for HITs yet – but we’re making progress*

The pushout

Given $f : C \to A$ and $g : C \to B$ (forming a *span*), the pushout is a type D fitting into a diagram:



It has point constructors inl and inr, and a path constructor

$$\mathrm{glue} : \Pi(x : C)(\mathrm{inl}(f\,x) = \mathrm{inr}(g\,x)).$$

The non-dependent elimination principle is simply the universal property.

Elimination principle for pushouts

Let D be the pushout of the span consisting of $f : C \to A$ and $g : C \to B$. Let $P : D \to \mathrm{Type}$ be given. We can define a section $s : \Pi(x : D)P(x)$ by giving:

$$
\begin{aligned}
s(\mathrm{inl}\, a) &: P(\mathrm{inl}\, a) && \text{for } a : A \\
s(\mathrm{inr}\, b) &: P(\mathrm{inl}\, b) && \text{for } b : B \\
\mathrm{apd}_s(\mathrm{glue}(x)) &: s(\mathrm{inl}(f\, x)) =^{P}_{\mathrm{glue}(x)} s(\mathrm{inr}(g\, x)) && \text{for } x : C
\end{aligned}
$$

Coequalizer Q, given $f, g : A \rightrightarrows B$: point constructor $q : B \to Q$;
path constructor $r : \Pi(x : A)(q(f(x)) = q(g(x)))$.

Interval $\mathbb{I}$: point constructors $0, 1 : \mathbb{I}$;
path constructor $\mathrm{seg} : 0 = 1$.

Suspension $\mathrm{susp}(A)$: point constructors $\mathrm{N}, \mathrm{S} : \mathrm{susp}(A)$;
path constructor $\mathrm{merid} : A \to \mathrm{N} = \mathrm{S}$.

Join $A * B$: point constructors $\mathrm{inl} : A \to A * B$, $\mathrm{inr}\, B \to A * B$
path constructor $\mathrm{glue} : \Pi(a : A)(b : B).\ \mathrm{inl}\, a = \mathrm{inr}\, b$.

Torus T^2: point constructor $\mathrm{base} : T^2$;
path constructors $p, q : \mathrm{base} = \mathrm{base}$;
2-path constructor: $s : p \cdot q = q \cdot p$

These already suffice to do a lot of homotopy theory. They are all definable using pushouts (and standard type operations).

A glimpse of the menagerie

Coequalizer Q, given $f, g : A \rightrightarrows B$: point constructor $q : B \to Q$;
path constructor $r : \Pi(x : A)(q(f(x)) = q(g(x)))$.

Interval $\mathbb{I}$: point constructors $0, 1 : \mathbb{I}$;
path constructor $\mathrm{seg} : 0 = 1$.

Suspension $\mathrm{susp}(A)$: point constructors $\mathrm{N}, \mathrm{S} : \mathrm{susp}(A)$;
path constructor $\mathrm{merid} : A \to \mathrm{N} = \mathrm{S}$.

Join $A * B$: point constructors $\mathrm{inl} : A \to A * B$, $\mathrm{inr}\, B \to A * B$
path constructor $\mathrm{glue} : \Pi(a : A)(b : B).\ \mathrm{inl}\, a = \mathrm{inr}\, b$.

Torus T^2: point constructor $\mathrm{base} : T^2$;
path constructors $p, q : \mathrm{base} = \mathrm{base}$;
2-path constructor: $s : p \cdot q = q \cdot p$

These already suffice to do a lot of homotopy theory. They are all definable using pushouts (and standard type operations).

Exercise: Prove that $\mathbb{I}$ is contractible.

Hubs and spokes

The torus used a 2-path constructor. It turns out that higher path constructors can always be avoided via the hubs-and-spokes method. For T^2, instead of the 2-path constructor we could add another point constructor $h : T^2$ (the hub) and a path constructor $s : \Pi(x : \mathbb{S}^1)(f\,x = h)$, where $f : \mathbb{S}^1 \to T^2\restriction$ is defined by circle-induction, mapping base to base and loop to $p \cdot q \cdot p^{-1} \cdot q^{-1}$. (*Drawing on blackboard*)

This is exactly the same principle as when we glue in a higher cell along a map f using a pushout:

$$
\begin{array}{ccc}
\mathbb{S}^n & \xrightarrow{f} & X \\
\downarrow & & \downarrow \\
1 & \longrightarrow & X'
\end{array}
$$

We return to constructing the truncations $\|A\|_n$ for $n \geq -2$.
Let $\|A\|_{-2} := 1$.

The *propositional truncation* $\|A\| = \|A\|_{-1}$ is the higher inductive type with: point constructor $|-| : A \to \|A\|$; and
path constructor $p : \Pi(x, y : \|A\|)(x = y)$.

$\|A\|$ is freely generated by a function from A and the fact that it should be a proposition.

This is our first example of a *recursive* HIT. The universal property follows from the recursion principle (exercise!).

Fact: A is n-truncated iff $\Omega^{n+1}(A, a)$ ($\simeq \mathrm{Map}_*(\mathbb{S}^{n+1}, (A, a))$) is contractible for all $a : A$.

This suggests the following description of $\|A\|_n$ as a HIT, generated by

- a function $|-| : A \to \|A\|_n$; and
- for each $r : \mathbb{S}^{n+1} \to \|A\|_n$, a hub point $h(r) : \|A\|_n$; and
- for each $r : \mathbb{S}^{n+1} \to \|A\|_n$, and each $x : \mathbb{S}^{n+1}$, a spoke $s_r(x) : h(x) = r(x)$.

Fact: This gives the right universal property (cf. HoTT book).

Truncations, reduced

It turns out that the truncations are definable in terms of pushouts!

- For propositional truncation, this is due to Floris van Doorn and Nicolai Kraus.
- For higher truncations, this is due to Egbert Rijke.

Suppose A : Set and $R : A \to A \to \text{Prop}$. Then we can form the quotient A/R as the set-coequalizer of the two projections

(*) $$\Sigma(a, b : A) R(a, b) \rightrightarrows A.$$

(This is the set-truncation of the type-coequalizer.)

In fact it can be useful in the general case of A : Type and $R : A \to A \to \text{Type}$ to form the coequalizer (*), as a *type-quotient*. This is a built-in HIT in the Lean proof assistant, generated by

- a function $q : A \to A/R$;
- for each $a, b : A$ and each $r : R(a, b)$ a path $e(r) : q(a) = q(b)$.

Higher inductive types can also be used to construct free algebras. For instance, if A : Set, we can construct the free group on A, $F(A)$, as generated by:

- A function $\eta : A \to F(A)$;
- A function $m : F(A) \times F(A) \to F(A)$;
- An element $e : F(A)$;
- A function $i : F(A) \to F(A)$;
- For each $x, y, z : F(A)$ a path $m(x, m(y, z)) = m(m(x, y), z)$;
- For each $x : F(A)$ paths $m(x, e) = x = m(e, x)$;
- For each $x : F(A)$ paths $m(x, i(x)) = e = m(i(x), x)$;
- The 0-truncation constructor.

Outline

We can define the higher homotopy groups of a pointed type A as $\pi_n(A,\star) := \|\Omega^n(A,\star)\|_0 = \pi_1(\Omega^{n-1}(A,\star))$. For $n \geq 2$, this is an abelian group (Eckmann-Hilton argument).

	$\mathbb{S}^0$	$\mathbb{S}^1$	$\mathbb{S}^2$	$\mathbb{S}^3$	$\mathbb{S}^4$	$\mathbb{S}^5$	$\mathbb{S}^6$	$\mathbb{S}^7$	$\mathbb{S}^8$
π_1	0	$\mathbb{Z}$	0	0	0	0	0	0	0
π_2	0	0	$\mathbb{Z}$	0	0	0	0	0	0
π_3	0	0	$\mathbb{Z}$	$\mathbb{Z}$	0	0	0	0	0
π_4	0	0	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}$	0	0	0	0
π_5	0	0	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}$	0	0	0
π_6	0	0	$\mathbb{Z}_{12}$	$\mathbb{Z}_{12}$	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}$	0	0
π_7	0	0	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}\times\mathbb{Z}_{12}$	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}$	0
π_8	0	0	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}_2^2$	$\mathbb{Z}_{24}$	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}$

A type A may be more or less *connected*:

$$\mathrm{isConn}_n(A) := \mathrm{isContr}(\|A\|_n).$$

-1-connected=inhabited, 0-connected=connected,
1-connected=simply-connected.

We can prove that suspension increases connectivity:

$$\mathrm{isConn}_n(A) \to \mathrm{isConn}_{n+1}(\mathrm{susp}\, A)$$

Thus, the n-sphere $\mathbb{S}^n$ is $n-1$-connected (what is the base case?).

Truncatedness, connectedness and homotopy groups

Lemma 1: If A is n-truncated and $a : A$, then $\pi_k(A, a) = 1$ for $k > n$.

Lemma 2: If A is n-connected and $a : A$, then $\pi_k(A, a) = 1$ for $k \leq n$.

Corollary: $\pi_k(\mathbb{S}^n) = 1$ for $k < n$.

Classifying types of groups

Recall that we can think of types as ∞-groupoids. A pointed, connected type represents an ∞-group.

What is the connection between discrete groups and 1-truncated ∞-groups?

Answer: Given BG, a pointed, connected, 1-truncated type, $G := \Omega BG$ is a discrete group.

Conversely: Given a discrete group G, we can construct a pointed, connected, 1-truncated type BG with $G = \Omega BG$ as a HIT!
What are the constructors?

The same procedure produces a univalent category given a set-presented precategory.

Covering spaces

In traditional homotopy theory it is relatively complicated to define covering spaces (e.g., using local homeomorphisms).

In HoTT, a covering space of A is simply a map $C : A \to \mathrm{Set}$, where $\mathrm{Set} := \Sigma(X : \mathrm{Type})\mathrm{isSet}(\mathrm{X})$.

Given $C : A \to \mathrm{Set}$, we get for any $a : A$ a $\pi_1(A, a)$-set (using that Set is 1-truncated. Favonia showed that you can go back: there is an equivalence between $\pi_1(A, a)$-sets and A-covering spaces when A is connected (recovering a classical theorem).

Infinity group actions

Given a pointed connected type BG, thought of as an ∞-group G, an action of G is just a dependent type $X : BG \to \mathrm{Type}$.

The type acted on is the fiber $X(\star)$, and the action is by transport. The quotient is just the dependent sum $\Sigma(x : BG)X(x)$!

The path fibration $BG \to \mathrm{Type}$, $\lambda x.\ \star = x$ corresponds to the right action of G on itself. The quotient is contractible.

The real projective spaces $\mathbb{RP}^n$ are traditionally quotients of $\mathbb{S}^n$ by the antipodal map.

Using the HIT BC_2 corresponding to the 2-element group C_2, note that the iterated joins $M^n : BC_2 \to \mathrm{Type}, \lambda x.\ (\star = x)^{*(n+1)}$, are the antipodal actions on the spheres $\mathbb{S}^n$, so we can define $\mathbb{RP}^n := \Sigma(x : BC_2)M^n(x)$.

A similar construction defined complex projective spaces.

(This construction is due to Egbert Rijke and myself.)

Eilenberg-MacLane spaces and cohomology

If G is abelian, we can form higher versions of BG, usually called $K(G,n)$. These are $(n-1)$-connected, n-truncated types with $\pi_n(K(G,n)) = G$.

$K(G,0) := G$ and $K(G,1) := BG$.

Cohomology is now simply defined as $H^n(A;G) := \|A \to K(G,n)\|_0$.

Outline

In HoTT we can construct the Cauchy-complete reals (without assuming dependent choice), as follows:

$$\begin{aligned} \mathrm{rat} &: \mathbb{Q} \to \mathbb{R} \\ \lim &: (x : \mathbb{Q}_+ \to \mathbb{R}) \to (\forall \delta, \varepsilon : \mathbb{Q}_+, x_\delta \sim_{\delta+\varepsilon} x_\varepsilon) \to \mathbb{R} \\ \mathrm{eq} &: (u, v : \mathbb{R}) \to (\forall \varepsilon : \mathbb{Q}_+, u \sim_\varepsilon v) \to u =_{\mathbb{R}} v \end{aligned}$$

eliding clauses for $\sim_\varepsilon$, and set truncation.

Complicated induction principle!

The cumulative set hierarchy

In HoTT, we can construct the cumulative hierarchy V as a HIT generated by:

$$\begin{aligned}
\mathrm{set} &: (A : \mathrm{Type}) \to (f : A \to V) \to V \\
\mathrm{eq} &: (A, B : \mathrm{Type}) \to (f : A \to V) \to (g : B \to V) \\
&\quad \to (\forall a : A, \exists b : B, f\,a =_V g\,b) \to (\forall b : B, \exists a : A, f\,a =_V g\,b) \\
&\quad \to \mathrm{set}(A, f) =_V \mathrm{set}(B, g)
\end{aligned}$$

plus a constructor making V 0-truncated.

Again, complicated induction principle!

Where to go from here: HoTT book and . . .

- Licata-Shulman, *Calculating the Fundamental Group of the Circle in Homotopy Type Theory*, 2013.
- Licata-Brunerie, $\pi_n(S^n)$ *in Homotopy Type Theory*, 2013.
- Favonia-Harper, *Covering Spaces in Homotopy Type Theory*, 2014.
- Licata-Finster, *Eilenberg-MacLane Spaces in Homotopy Type Theory*, 2014.
- Cavallo, *Synthetic Cohomology in Homotopy Type Theory*, MSc thesis, 2015.
- Favonia-Finster-Licata-Lumsdaine, *A mechanization of the Blakers-Massey connectivity theorem in Homotopy Type Theory*, preprint, 2016.
- Brunerie, *On the homotopy groups of spheres in homotopy type theory*, PhD thesis, 2016.
- Lumsdaine-Shulman, *Semantics and syntax of higher inductive types*, slides, 2016.
- you?