Recent progress in Homotopy type theory

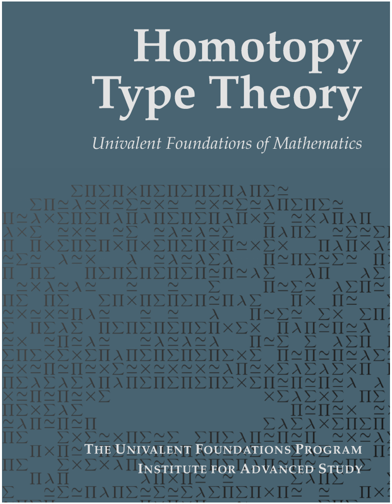
Egbert Rijke Bas Spitters

Radboud University Nijmegen

June 19th, 2013

Supported by EU FP7 STREP FET-open ForMATH

Most of the presentation is based on the forthcoming book:



Homotopy type theory

Collaborative effort lead by Awodey, Coquand, Voevodsky
at Institute for Advanced Study
Forthcoming book, library of formal proofs.

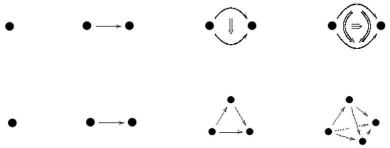
Towards a new practical foundation for mathematics.
Closer to mathematical practice, inherent treatment of equivalences.

Towards a new design of proof assistants:
Proof assistant with a clear semantics,
guiding the addition of new features.

Concise computer proofs.

Two generalizations of Sets

To keep track of isomorphisms we want to generalize sets to groupoids (categories with all morphisms invertible),
2-groupoids (add coherence conditions for associativity),
..., ∞-groupoids
∞-groupoids are modeled by Kan simplicial sets.
(Grothendieck homotopy hypothesis)



Topos theory

[Picture of blind men and the elephant.]

Topos theory

A topos is like:

- a category of sheaves on a site
- a category with finite limits and power-objects
- a generalized space
- a semantics for intuitionistic formal systems

Type theory

Type theory is

- a foundation for constructive mathematics, an abstract set theory ($\Pi\Sigma$).
- a calculus for proofs
- an abstract programming language
- a system for developing computer proofs

Examples: nat, $nat \times nat$, $vector :\equiv \Sigma_{n:nat}\mathrm{list}(n)$, $x = y$.

Higher topos theory

A higher topos is like:

- a model category which is Quillen equivalent to simplicial $PSh(C)_S$ for some model site (C, S).
- a generalized space (presented by homotopy types)
- a semantics for Martin-Löf type theory with univalence and higher inductive types ??
- a place for abstract homotopy theory
- a place for abstract algebraic topology

Envisioned applications

Type theory with univalence and higher inductive types as the internal language for higher topos theory?

- higher categorical foundation of mathematics
- framework for formalization of mathematics
  internalizes reasoning with isomorphisms
- expressive programming language
- language for synthetic pre-quantum physics (like Bohrification)
  Schreiber/Shulman

Here: develop mathematics in this framework.

Homotopy Type Theory

The homotopical interpretation of type theory is that we think of:

- types as spaces
- dependent types as fibrations (continuous families of types)
- identity types as path spaces

We define homotopy between functions $A \to B$ by:
$f \sim g :\equiv \prod_{(x:A)} f(x) =_B g(x)$.
The function extensionality principle asserts that the canonical function $(f =_{A \to B} g) \to (f \sim g)$ is an equivalence.

(homotopy type) theory = homotopy (type theory)

The hierarchy of complexity

Definition

We say that a type A is contractible if there is an element of type

$$\mathsf{isContr}(A) :\equiv \sum_{(x:A)} \prod_{(y:A)} x =_A y$$

Contractible types are said to be of level -2.

Definition

We say that a type A is a mere proposition if there is an element of type

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

Mere propositions are said to be of level -1.

The hierarchy of complexity

Definition

We say that a type A is a set if there is an element of type

$$\mathrm{isSet}(A) :\equiv \prod_{x,y:A} \mathrm{isProp}(x =_A y)$$

Sets are said to be of level 0.

Definition

Let A be a type. We define

$$\begin{aligned}\text{is-}(-2)\text{-type}(A) &:\equiv \mathrm{isContr}(A) \\ \text{is-}(n+1)\text{-type}(A) &:\equiv \prod_{x,y:A} \text{is-}n\text{-type}(x =_A y)\end{aligned}$$

Equivalence

A good (homotopical) definition of equivalence is:

$$\prod_{b:B} \mathsf{isContr}\left(\sum\nolimits_{(a:A)}(f(a) =_B b)\right)$$

The identity type of the universe

The univalence axiom describes the identity type of the universe Type. Recall that there is a canonical function

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

The univalence axiom: this function is an equivalence.

- The univalence axiom formalizes the informal practice of substituting a structure for an isomorphic one.
- It implies function extensionality
- It is used to reason about higher inductive types

Voevodsky: The univalence axiom holds in Kan simplicial sets.

Direct consequences

Univalence implies:

- logically equivalent propositions are equal
- isomorphic Sets are equal
  all definable type theoretical constructions respect isomorphisms

Theorem (Structure invariance principle)

Isomorphic structures (monoids, groups,...) may be identified.

Informal in Bourbaki.

HITs

Higher inductive types were conceived by Bauer, Lumsdaine, Shulman and Warren.

The first examples of higher inductive types include:

- The interval
- The circle
- Propositional reflection

It was shown that:

- Having the interval implies function extensionality.
- The fundamental group of the circle is $\mathbb{Z}$.

Higher inductive types internalize colimits.

Ordinary inductive types are introduced with

1. basic constructors
2. from which we derive an induction principle.

The induction principle is formulated dependently:

1. it tells us under what condition there exists a term of type $\prod_{(x:W)} P(X)$ given a dependent type P over the inductively defined type W.
2. the dependency of the induction principle ensures the uniqueness part of the universal.

Higher inductive types

Inductive types/free algebra (nat, list) introduce new objects.

```
Inductive nat : Type :=
   O : nat
  | S : nat → nat
```

With higher inductive types, we allow paths among the basic constructors. For example:

- The interval I has basic constructors

$$0_I, 1_I : I \qquad \text{and} \qquad \mathsf{seg} : 0_I =_I 1_I.$$

- The circle $\mathbb{S}^1$ has basic constructors

$$\mathsf{base} : \mathbb{S}^1 \qquad \text{and} \qquad \mathsf{loop} : \mathsf{base} =_{\mathbb{S}^1} \mathsf{base}.$$

With paths among the basic constructors, the induction principle becomes more complicated.
The induction principle describes a condition under which we can prove a property $P(x)$ for all x in the inductively defined type.

Squash

Squash equates all terms in a type

Higher inductive definition:

```
Inductive squash (A : Type) : Type :=
  | inhab : A → squash A
  | inhab_path : forall (x y: squash A), x = y
```

Reflection into the mere propositions

Logic

Set theoretic foundation is formulated in first order logic.
In type theory logic can be defined, propositions as (-1)-types:

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

models constructive logic, not axiom of choice.

Lemma

Suppose $P : A \to \mathrm{Type}$ is a family of types, let $p : x =_A y$ and let $u : P(x)$. Then there is a term $p_*(u) : P(y)$, called the transportation of u along p.

Lemma

Suppose $f : \prod_{(x:A)} P(x)$ is a dependent function, and let $p : x =_A y$. Then there is a path $f(p) : p_*(f(x)) =_{P(y)} f(y)$.

In the case of the interval, we see that in order for a function $f : \prod_{(x:I)} P(x)$ to exist, we must have

$$\begin{aligned} f(0_I) &: P(0_I) \\ f(1_I) &: P(1_I) \\ f(\mathsf{seg}) &: \mathsf{seg}_*(f(0_I)) =_{P(1_I)} f(1_I) \end{aligned}$$

Induction with the interval

The induction principle for the interval is that for every $P : I \to \mathrm{Type}$, if there are

- $u : P(0_I)$ and $v : P(1_I)$
- $p : \mathrm{seg}_*(u) =_{P(1_I)} v$

then there is a function $f : \prod_{(x:I)} P(x)$ with

- $f(0_I) :\equiv u$ and $f(1_I) :\equiv v$
- $f(\mathrm{seg}) = p$.

Induction with the circle

The induction principle for the circle is that for every $P : \mathbb{S}^1 \to \mathrm{Type}$, if there are

- $u : P(\mathsf{base})$
- $p : \mathsf{loop}_*(u) =_{P(\mathsf{base})} u$

then there is a function $f : \prod_{(x:\mathbb{S}^1)} P(x)$ with

- $f(\mathsf{base}) :\equiv u$
- $f(\mathsf{loop}) = p$.

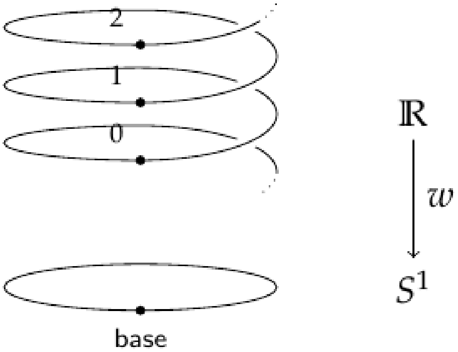
Using univalence to reason about HITs

How do we use univalence to reason about HITs?

- Suppose we have a HIT W.
- and we want to describe a property $P : W \to \mathrm{Type}$.
- for the point constructors of W we have to give types.
- for the path constructors of W we have to give paths between those types
- by univalence, it suffices to give equivalences between those types.

Suppose, in our inductive type W we have $p : x =_W y$ and $P(x) :\equiv A$ and $P(y) :\equiv B$ and to p we have assigned the equivalence $e : A \simeq B$. Then transporting along p computes as applying the equivalence e.

The universal cover, computing base $=_{\mathbb{S}^1}$ base



The universal cover, computing base $=_{\mathbb{S}^1}$ base

With this idea, we can construct the universal cover of the circle: $C : \mathbb{S}^1 \to \mathrm{Type}$. Our goal is to use C to show that

$$(\mathrm{base} =_{\mathbb{S}^1} \mathrm{base}) \simeq \mathbb{Z}.$$

We define $C : \mathbb{S}^1 \to \mathrm{Type}$ by:

- $C(\mathrm{base}) :\equiv \mathbb{Z}$
- To transport along loop we apply the equivalence $\mathrm{succ} : \mathbb{Z} \to \mathbb{Z}$.

Theorem

The cover C has the property that

$$\mathrm{isContr}\left(\sum_{(x:\mathbb{S}^1)} C(x)\right)$$

'$\mathbb{R}$ is contractible'

Before we prove the theorem let us indicate why it is useful.

- Suppose A, $a : A$ is a type and $P : A \to \mathrm{Type}$.
- there is a term of $P(a)$.
- and $\sum_{(x:A)} P(x)$ is contractible.

Note that

- $\sum_{(x:A)} x =_A a$ is contractible as well
- by the assumption $P(a)$, there exists a function

$$f(x) : (x =_A a) \to P(x)$$

for every $x : A$.

Theorem

If $f : \prod_{(x:A)} P(x) \to Q(x)$ *induces an equivalence*

$$(\textstyle\sum_{(x:A)} P(x)) \to (\textstyle\sum_{(x:A)} Q(x)),$$

then each $f(x) : P(x) \to Q(x)$ *is an equivalence.*

Hence under the above assumptions we obtain that

$$P(x) \simeq (x =_A a)$$

In particular, the theorem about the universal cover has the corollary that

$$C(x) \simeq (x =_{\mathbb{S}^1} \mathrm{base})$$

Theorem

The cover C has the property that

$$\mathrm{isContr}\left(\sum_{(x:\mathbb{S}^1)} C(x)\right)$$

$(\mathrm{base}; 0)$ is the center of contraction and

$$\alpha : \prod_{(k:\mathbb{Z})} \sum_{(p:\mathrm{base}=_{\mathbb{S}^1}\mathrm{base})} p_*(k) =_{\mathbb{Z}} 0.$$

With some calculations:

Theorem

$(\mathrm{base} =_{\mathbb{S}^1} \mathrm{base}) \simeq \mathbb{Z}$*.*

Fundamental group of the circle is $\mathbb{Z}$.
The proof is by induction on $\mathbb{S}^1$.

Formal proofs

This theorem has a concise computer proof.
Likewise, the following has been done:

- total space of Hopf fibration
- computing homotopy groups upto $\pi_4(S^3)$
- Freudenthal suspension theorem
- van Kampen theorem
- James construction
- ...

Most proofs are computer formalized, with short proofs.

Quotients

Towards sets in homotopy type theory.

Voevodsky: univalence provides quotients.

Quotients can also be defined as a higher inductive type

```
Inductive Quot (A : Type) (R:rel A) : Type :=
  | quot : A → Quot A
  | quot_path : forall x y, (R x y), quot x = quot y
  | _ :isset (Quot A).
```

We verified the universal properties of quotients.

Modelling set theory

Theorem (Rijke,S)

*0-*Type *is a* ΠW*-pretopos (constructive set theory).*

This is important for computer verification.
Assuming AC, we have a well-pointed boolean elementary topos with choice (Lawvere set theory).
Define the cumulative hierarchy $\emptyset, P(\emptyset), \ldots, P(V_\omega), \ldots,$ by higher induction. Then V is a model of constructive set theory.

Theorem

Assuming AC, V models ZFC.

We have retrieved the old foundation.

Subobject classifier

$$
\begin{tikzcd}
I \arrow[r, "!"] \arrow[d, "\alpha"] & 1 \arrow[d, "\mathsf{True}"'] \\
A \arrow[r, "P"] & \mathsf{Prop}
\end{tikzcd}
$$

Prop classifies monos into A
Equivalence between predicates and subsets.

Object classifier

$Fam(A) := \{(I, \alpha) \mid I : Type, \alpha : I \to A\}$ (slice cat)
$Fam(A) \cong A \to Type$
(Grothendieck construction, using univalence)

$$\begin{tikzcd} I \arrow[r, "i"] \arrow[d, "\alpha"] & \mathrm{Type}_\bullet \arrow[d, "\pi_1"'] \\ A \arrow[r, "P"] & \mathrm{Type} \end{tikzcd}$$

$\mathrm{Type}_\bullet = \{(B, x) \mid B : \mathrm{Type}, x : B\}$
Classifies *all* maps into A + group action of isomorphisms
Crucial construction in ∞-toposes.
Proper treatment of Grothendieck universes from set theory.

1-Category theory

Type of objects. Hom-set (0-Type) between any two elements.
Isomorphic objects objects are equal.
'Rezk complete categories.'

Theorem

$F : A \to B$ is an equivalence of categories iff it is an isomorphism.

Generalization of the Structure Identity Principle

Every pre-category has a Rezk completion.

Towards elementary higher topos theory

(Homotopy) limits can be defined.
We can define all 1-dimensional colimits using HITs.
Seems to generalize to n-dimensional colimits.
Rijke,S: 1-dimensional internal version of:
Left fibrations over a sSet S are equivalent to functors to the ∞-cat of spaces.
Using an internal model construction.

Theorem (Rijke,S)

Descent: For a diagram D, there is an equivalence:

$$\mathrm{equiFib(D)} \cong \mathrm{colim}(D) \to \mathrm{Type}$$

Towards elementary higher topos theory

Truncations, localization from higher topos theory can be captured as a modal operator $\circ$ from logic.
Natural generalization of n-truncations.

We want also: sheaf models for type theory and programming semantics.

Stable orthogonal factorization system

Every modality $\circ$ defines a stable factorization system $(\mathcal{E}_\circ, \mathcal{M}_\circ)$ with:

$$\mathcal{E}_\circ(f) \equiv \prod(b : B),\ \mathsf{isContr}(\circ\mathsf{hFiber}(f, b)).$$

and $\mathcal{M}_\circ$ the class of functions all of which fibers are modal, *modal functions* for short.

For -1-trunction, we get epi-mono-factorization.

For n-truncation, we show that there is a stable orthogonal factorization system $(\mathcal{E}, \mathcal{M})$ where $\mathcal{E}$ is the class of n-connected functions and $\mathcal{M}$ is the class of n-truncated functions.

Towards elementary higher topos theory

A modality $\circ$ is lex if the functor $\circ$ preserves pullbacks.
They correspond to reflective subtoposes.

Theorem (Shulman)

Every Lawvere-Tierney topology on Prop *extends to a (minimal) lex modality on* Type.

Conclusion

Forthcoming book, library of formal proofs.

Towards a new practical foundation for mathematics based on higher topos theory.
Closer to mathematical practice, less ad hoc encodings.

Towards a new design of proof assistants, programming languages:
Proof assistant with a clear semantics,
guiding the addition of new features.

`homotopytypetheory.org`