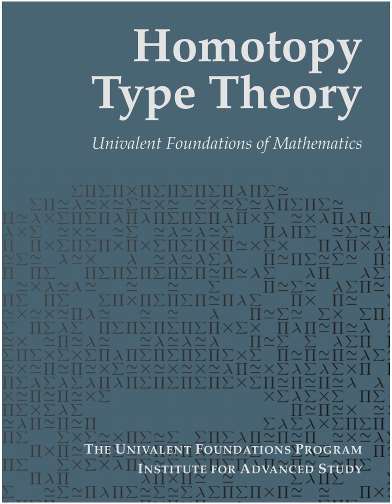
Sets in Homotopy type theory

Bas Spitters
Aarhus

About me

- PhD thesis on constructive analysis
- Connecting Bishop's pointwise mathematics w/topos theory (w/Coquand)
- Formalization of effective real analysis in Coq
  O'Connor's PhD
  part EU ForMath project
- Topos theory and quantum theory
- Univalent foundations as a combination of the strands
  co-author of the book and the Coq library
- guarded homotopy type theory: applications to CS

Most of the presentation is based on the book and Sets in HoTT (with Rijke).



Homotopy type theory

Towards a new practical foundation for mathematics.

- Modern ((higher) categorical) mathematics
- Formalization
- Constructive mathematics

Closer to mathematical practice, inherent treatment of equivalences.

Homotopy type theory

Towards a new practical foundation for mathematics.

- Modern ((higher) categorical) mathematics
- Formalization
- Constructive mathematics

Closer to mathematical practice, inherent treatment of equivalences.

Towards a new design of proof assistants:
Proof assistant with a clear (denotational) semantics, guiding the addition of new features.
E.g. guarded cubical type theory

Challenges

pre-HoTT:

Sets as Types no quotients (setoids), no unique choice (in Coq), ...

Types as Sets not fully abstract $\rightarrow$ Groupoid model

Towards a more symmetric treatment.

Challenges

pre-HoTT:

Sets as Types no quotients (setoids), no unique choice (in Coq), ...

Types as Sets not fully abstract $\rightarrow$ Groupoid model

Towards a more symmetric treatment.

Formalization of discrete mathematics: four color theorem, Feit Thompson, ... computational interpretation was crucial.
Can this be extended to non-discrete types?

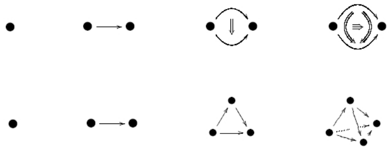
Two generalizations of Sets

To keep track of isomorphisms we generalize sets to
groupoids (proof relevant equivalence relations)
2-groupoids (add coherence conditions for associativity),
. . . , weak ∞-groupoids

Two generalizations of Sets

To keep track of isomorphisms we generalize sets to groupoids (proof relevant equivalence relations)
2-groupoids (add coherence conditions for associativity),
. . . , weak ∞-groupoids

Weak ∞-groupoids are modeled by Kan simplicial sets.
(Grothendieck homotopy hypothesis)



Topos theory

Wikimedia Commons

Parabel

Four blind men, who had been blind from birth, wanted to know what an elephant was like, so they asked an elephant-driver for information. He led them to an elephant, and invited them to examine it, so one man felt the elephant's leg, another its trunk, another its tail and the fourth its ear. Then they attempted to describe the elephant to one another. The first man said, 'The elephant is like a tree'. 'No,' said the second, 'the elephant is like a snake'. 'Nonsense' said the third, 'the elephant is like a broom'. 'You are all wrong,' said the fourth, 'the elephant is like a fan'. And so they went on arguing amongst themselves, while the elephant stood watching them quietly.

Topos theory

A topos is like:

- a semantics for intuitionistic formal systems
  model of intuitionistic higher order logic/type theory.
- a category of sheaves on a site (forcing)
- a category with finite limits and power-objects
- a generalized space

Topos theory

A topos is like:

- a semantics for intuitionistic formal systems
  model of intuitionistic higher order logic/type theory.
- a category of sheaves on a site (forcing)
- a category with finite limits and power-objects
- a generalized space

Free topos as a foundation of mathematics (Lambek and Scott):
Proposal to reconcile formalism, platonism and intuitionism

Higher topos theory

Combine these two generalizations of sets.

A higher topos is (represented by):

a model category which is Quillen equivalent to simplicial $Sh(C)_S$ for some model ∞-site (C, S)

Less precisely:

- a generalized space (presented by homotopy types)
- a place for abstract homotopy theory
- a place for abstract algebraic topology
- a semantics for Martin-Löf type theory with univalence and higher inductive types (Shulman/Cisinski).

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 large scale formalization of mathematics
- foundation for constructive mathematics
  e.g. type theory with the fan rule
- expressive programming language (e.g. `cubical`)
- free ∞-topos for a philosophy of maths?

[2] https://github.com/HoTT/HoTT/

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 large scale formalization of mathematics
- foundation for constructive mathematics
  e.g. type theory with the fan rule
- expressive programming language (e.g. `cubical`)
- free ∞-topos for a philosophy of maths?

Towards elementary ∞-topos theory.
Here: Develop mathematics in this framework

[2] https://github.com/HoTT/HoTT/

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 large scale formalization of mathematics
- foundation for constructive mathematics
  e.g. type theory with the fan rule
- expressive programming language (e.g. `cubical`)
- free ∞-topos for a philosophy of maths?

Towards elementary ∞-topos theory.
Here: Develop mathematics in this framework
Coq formalization[2]

[2] `https://github.com/HoTT/HoTT/`

Type theory

Type theory is another elephant

- 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

Homotopy Type Theory

The homotopical interpretation of type theory

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

(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

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

Sets are said to be of level 0.

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:

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

This is a mere proposition.

Functional extensionality

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.

Functional extensionality

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.

Direct consequences of Univalence

Univalence implies:

- functional extensionality
  Lemma ap10 {A B} (f g : A → B): (f=g → f == g).
  Lemma FunExt {A B}: forall f g, IsEquiv (ap10 f g).
- logically equivalent propositions are equal:
  Lemma uahp '{ua:Univalence}: forall P P': hProp, (P ↔ P')→ P = P'.
- isomorphic Sets may be identified equal
  all definable type theoretical constructions respect isomorphisms

Structure invariance principle

Theorem (Structure invariance principle)

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

Informal in Bourbaki. Formalized in agda (Coquand, Danielsson).

Proposition as types

$\vee$	$+$
$\wedge$	$\times$
$\rightarrow$	$\rightarrow$
$\forall$	Π
$\exists$	Σ

The classes of n-types are closed under

- dependent products
- dependent sums
- identity types
- Inductive types (W-types), when $n \geq -1$
- equivalences

The classes of n-types are closed under

- dependent products
- dependent sums
- identity types
- Inductive types (W-types), when $n \geq -1$
- equivalences

Thus, besides propositions as types we also get propositions as n-types for every $n \geq -2$.
Often, we will stick to 'propositions as types', but some mathematical concepts (e.g. the axiom of choice) are better interpreted using 'propositions as (-1)-types' (mere propositions).

Truncation

Higher inductive definition:

```
Inductive minus1Trunc (A : Type) : Type :=
  | min1 : A → minus1Trunc A
  | min1_path : forall (x y: minus1Trunc A), x = y
```

Reflection into the mere propositions

Awodey, Bauer [ ]-types as internal language for regular cats

Theorem

epi-mono factorization. Set is a regular category.

n-connected-*n*-truncated-factorization

space of factorizations is contractible

Logic

Set theoretic foundation is formulated in first order logic.

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.

Unique choice

```
Definition hexists {X} (P:X→ Type):=minus1Trunc (sig P).

Definition atmost1P {X} (P:X→ Type):=
    (forall x1 x2 :X, P x1 → P x2 → (x1 = x2 )).

Definition hunique {X} (P:X→ Type):=(hexists P) * (atmost1P P).

Lemma iota {X} (P:X→ Type):
  (forall x, IsHProp (P x)) → (hunique P) → sig P.
```

Direct from elimination principle of truncations.

Unique choice

```
Definition hexists {X} (P:X→ Type):=minus1Trunc (sig P).

Definition atmost1P {X} (P:X→ Type):=
    (forall x1 x2 :X, P x1 → P x2 → (x1  = x2 )).

Definition hunique {X} (P:X→ Type):=(hexists P) * (atmost1P P).

Lemma iota {X} (P:X→  Type):
  (forall x, IsHProp (P x)) → (hunique P) → sig P.
```

Direct from elimination principle of truncations.

In Coq we cannot escape Prop because we want program extraction.
Exact completion: freely add quotients to a category.
Similarly: Consider setoids $(T, \equiv)$ type with equivalence relation.
Spiwack: Setoids in Coq give a quasi-topos (topos without AC!).

Quotients

Towards sets in homotopy type theory.

Quotients

Towards sets in homotopy type theory.
Voevodsky: univalence provides (impredicative) quotient types.
Consider the type of equivalence classes.
Requires small power type: $A : U \vdash (A \to Prop_U) : U$.
Dependents on propositional univalence:
equivalent propositions are equal

Quotients

Towards sets in homotopy type theory.
Voevodsky: univalence provides (impredicative) quotient types.
Consider the type of equivalence classes.
Requires small power type: $A : U \vdash (A \to Prop_U) : U$.
Dependents on propositional univalence:
equivalent propositions are equal

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).
```

Truncated colimit.
We verified the universal properties of quotients (exactness).

Modelling set theory

pretopos: extensive exact category
(extensive: good coproducts, exact: good quotients)
ΠW-pretopos: pretopos with Π and W-types.

Theorem (Rijke,S)

*0-*Type *is a* Π*W-pretopos ('constructive set theory').*

This is important for computer verification.

Modelling set theory

pretopos: extensive exact category
(extensive: good coproducts, exact: good quotients)
ΠW-pretopos: pretopos with Π and W-types.

Theorem (Rijke,S)

*0-*Type *is a* Π*W-pretopos ('constructive set theory').*

This is important for computer verification.
Assuming AC, and thus classical logic and impredicativity,
we have a well-pointed boolean elementary topos with choice
(Lawvere structural set theory).

Modelling set theory

pretopos: extensive exact category
(extensive: good coproducts, exact: good quotients)
ΠW-pretopos: pretopos with Π and W-types.

Theorem (Rijke,S)

0-Type *is a* Π*W-pretopos ('constructive set theory').*

This is important for computer verification.
Assuming AC, and thus classical logic and impredicativity, we have a well-pointed boolean elementary topos with choice (Lawvere structural set theory).

Modelling set theory

Define the cumulative hierarchy

$$\emptyset, P(\emptyset), \ldots, P(V_\omega), \ldots$$

by higher induction induction.
Then V is a model of constructive set theory
(incl replacement, separation,
but no strong collection, subset collection)
Constructively, this is a higher inductive-inductive definition.

Theorem (Awodey)

Assuming AC, V models ZFC.

We have retrieved the old foundation.

Predicativity (some context)

Impredicative definition as specification (minimal such that)
Predicative definitions give a construction

Avoid Russell's paradox $Set : Set$.
Distinguish small and large objects.
By propositions as types, Prop(=Set) will not be small.
Martin-Löf type theory is predicative.

Proof theoretic strength of IZF equals ZF, but CZF is much lower

Predicativity (some context)

Impredicative definition as specification (minimal such that)
Predicative definitions give a construction

Avoid Russell's paradox $Set : Set$.
Distinguish small and large objects.
By propositions as types, Prop(=Set) will not be small.
Martin-Löf type theory is predicative.

Proof theoretic strength of IZF equals ZF, but CZF is much lower

Shulman: fits well with higher dimensional mathematics
Usually the $(n+1)$-category of all small n-categories is not small.
Why should we expect it for $n = -1$: the poset of all truth values.

Predicativity (some context)

Impredicative definition as specification (minimal such that)
Predicative definitions give a construction

Avoid Russell's paradox $Set : Set$.
Distinguish small and large objects.
By propositions as types, Prop(=Set) will not be small.
Martin-Löf type theory is predicative.

Proof theoretic strength of IZF equals ZF, but CZF is much lower

Shulman: fits well with higher dimensional mathematics
Usually the $(n+1)$-category of all small n-categories is not small.
Why should we expect it for $n = -1$: the poset of all truth values.

Voevodsky's resizing rules are a way of adding impredicativity to type theory. They have not been implemented yet. Also meta-theory has not been completely worked out.

Predicativity

In predicative topos theory: no subobject classifier/power set.
Algebraic Set Theory (AST) provides a framework for defining various classes predicative toposes/set theories.

Predicativity

In predicative topos theory: no subobject classifier/power set.
Algebraic Set Theory (AST) provides a framework for defining various classes predicative toposes/set theories.
Categorical treatment of set and class theories.
Two challenges:

- hSet does not (seem to) have the collection axiom from AST.
- The universe is not a set, but a groupoid!

Higher categorical version of AST?

Predicativity

In predicative topos theory: no subobject classifier/power set.
Algebraic Set Theory (AST) provides a framework for defining various classes predicative toposes/set theories.
Categorical treatment of set and class theories.
Two challenges:

- hSet does not (seem to) have the collection axiom from AST.
- The universe is not a set, but a groupoid!

Higher categorical version of AST?
Perhaps HoTT already provides this?...

Large subobject classifier

The subobject classifier lives in a higher universe.

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

With propositional univalence, hProp classifies monos into A.

$$A, I : U_i \qquad hProp_i := \Sigma_{B:U_i} isprop(B) \quad hProp_i : U_{i+1}$$

Equivalence between predicates and subsets.
Use universe polymorphism. Check that there is some way to satisfy the constraints. Tool support.
This correspondence is the crucial property of a topos.

Large subobject classifier

The subobject classifier lives in a higher universe.

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

With propositional univalence, hProp classifies monos into A.

$$A, I : U_i \qquad hProp_i := \Sigma_{B:U_i} isprop(B) \quad hProp_i : U_{i+1}$$

Equivalence between predicates and subsets.
Use universe polymorphism. Check that there is some way to satisfy the constraints. Tool support.
This correspondence is the crucial property of a topos.
Sanity check: epis are surjective (by universe polymorphism).

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.
Grothendieck universes from set theory by universal property

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.
Grothendieck universes from set theory by universal property
Accident: $hProp_\bullet \equiv 1$?

Object classifier

Theorem (Rijke/S)

Assuming funext, TFAE

1. *Univalence*
2. *Object classifier*
3. *Descent: Homotopy colimits defined by higher inductive types behave well.*

$2 \leftrightarrow 3$ is fundamental in higher topos theory.
Translate statement and give a direct proof.

Computational interpretation

Computational interpretation of univalence and some HITs build on the topos of cubical sets (Coquand).

Aarhus: Internal model construction in the logic of cubical sets.
Can be extended with guarded dependent types.
Solves problem from CS proof system for guarded recursive types.

Conclusion

- Practical foundation for mathematics
- HoTT generalizes the old foundation
- Can import proofs from iHOL (using universe polymorphism).
- Towards a proof assistant w/ denotational semantics (CTT,GCTT)
- Towards elementary higher topos theory
- Question: applications of (pre)sheaf models to type theory $\hat{\omega}$, modalities (RSS) for sheaf models.