Sets in HoTT

Egbert Rijke Bas Spitters

Radboud University Nijmegen

April 23rd, 2013

Challenges of current type theories

For finitary mathematics (Coq) type theory works very well.
The extension to infinitary mathematics is challenging.
No: quotients, functional extensionality, subset types...
Univalence Axiom to the rescue!

Setting

- Univalent foundations of mathematics.

Setting

- Univalent foundations of mathematics.
  - We work with a univalent universe U.

Setting

- Univalent foundations of mathematics.
  - We work with a univalent universe U.
- Types in U have the structure of weak ∞-groupoids.

Setting

- Univalent foundations of mathematics.
  - We work with a univalent universe U.
- Types in U have the structure of weak ∞-groupoids.
- Sets (Set) in U are types for which *UIP* is valid.

Setting

- Univalent foundations of mathematics.
  - We work with a univalent universe U.
- Types in U have the structure of weak ∞-groupoids.
- Sets (Set) in U are types for which *UIP* is valid.
- Mere propositions (Prop) in U are types with proof irrelevance.

Aim of the current work

- To understand the category Set in a univalent universe.

Aim of the current work

- To understand the category Set in a univalent universe.
  - Initial question: Is Set a predicative topos?

Aim of the current work

- To understand the category Set in a univalent universe.
  - Initial question: Is Set a predicative topos?
    Do we have

Aim of the current work

- To understand the category Set in a univalent universe.
  - Initial question: Is Set a predicative topos?
    Do we have
    - (Stable) image factorization,

Aim of the current work

- To understand the category Set in a univalent universe.
  - Initial question: Is Set a predicative topos?
    Do we have
    - (Stable) image factorization,
    - quotients,

Aim of the current work

- To understand the category Set in a univalent universe.
  - Initial question: Is Set a predicative topos?
    Do we have
    - (Stable) image factorization,
    - quotients,
    - a (large) subobject classifier,

Aim of the current work

- To understand the category Set in a univalent universe.
  - Initial question: Is Set a predicative topos?
    Do we have
    - (Stable) image factorization,
    - quotients,
    - a (large) subobject classifier,
    - an object classifier,

Aim of the current work

- To understand the category Set in a univalent universe.
  - Initial question: Is Set a predicative topos?
    Do we have
    - (Stable) image factorization,
    - quotients,
    - a (large) subobject classifier,
    - an object classifier,
    - the collection axiom?

Aim of the current work

- To understand the category Set in a univalent universe.
  - Initial question: Is Set a predicative topos?
    Do we have
    - (Stable) image factorization,
    - quotients,
    - a (large) subobject classifier,
    - an object classifier,
    - the collection axiom?
- We will use ideas from

Aim of the current work

- To understand the category Set in a univalent universe.
  - Initial question: Is Set a predicative topos?
    Do we have
    - (Stable) image factorization,
    - quotients,
    - a (large) subobject classifier,
    - an object classifier,
    - the collection axiom?
- We will use ideas from
  - Algebraic Set Theory (Joyal, Moerdijk).

Aim of the current work

- To understand the category Set in a univalent universe.
  - Initial question: Is Set a predicative topos?
    Do we have
    - (Stable) image factorization,
    - quotients,
    - a (large) subobject classifier,
    - an object classifier,
    - the collection axiom?
- We will use ideas from
  - Algebraic Set Theory (Joyal, Moerdijk).
  - Higher category theory (Lurie, Rezk)

Squash

- The inclusion of Prop in U has a left adjoint called (-1)-truncation.

Squash

- The inclusion of Prop in U has a left adjoint called (-1)-truncation.

$$\| - \|_{-1} : \mathsf{U} \to \mathsf{Prop}.$$

Squash

- The inclusion of Prop in U has a left adjoint called (-1)-truncation.

$$\| - \|_{-1} : \mathsf{U} \to \mathsf{Prop}.$$

With unit

$$| - |_{-1} : A \to \|A\|$$

for every $A : \mathsf{U}$.

Squash

- The inclusion of Prop in U has a left adjoint called (-1)-truncation.

  $$\| - \|_{-1} : \mathsf{U} \to \mathsf{Prop}.$$

  With unit

  $$| - |_{-1} : A \to \|A\|$$

  for every $A : \mathsf{U}$.
- The universal property of (-1)-truncation is that
  - For every type $A : \mathsf{U}$,

Squash

- The inclusion of Prop in U has a left adjoint called (-1)-truncation.

  $$\| - \|_{-1} : \mathsf{U} \to \mathsf{Prop}.$$

  With unit

  $$| - |_{-1} : A \to \|A\|$$

  for every $A : \mathsf{U}$.
- The universal property of (-1)-truncation is that
  - For every type $A : \mathsf{U}$,
  - For every family $P : \|A\|_{-1} \to \mathsf{Prop}$ of mere propositions over $\|A\|_{-1}$,

Squash

- The inclusion of Prop in U has a left adjoint called (-1)-truncation.

  $$\|-\|_{-1} : \mathsf{U} \to \mathsf{Prop}.$$

  With unit

  $$|-|_{-1} : A \to \|A\|$$

  for every $A : \mathsf{U}$.
- The universal property of (-1)-truncation is that
  - For every type $A : \mathsf{U}$,
  - For every family $P : \|A\|_{-1} \to \mathsf{Prop}$ of mere propositions over $\|A\|_{-1}$,
  - The *pre-composition* function

    $$\lambda s.\, \lambda a.\, s(|a|_{-1}) : \prod_{x:\|A\|_{-1}} P(x) \to \prod_{a:A} P(|a|_{-1})$$

    is an equivalence.

0-truncation

- The inclusion of Set in U has a left adjoint called 0-truncation.

0-truncation

- The inclusion of Set in U has a left adjoint called 0-truncation.

$$\| - \|_0 : \mathsf{U} \to \mathsf{Set}.$$

0-truncation

- The inclusion of Set in U has a left adjoint called 0-truncation.

$$\| - \|_0 : \mathsf{U} \to \mathsf{Set}.$$

With unit

$$| - |_0 : A \to \|A\|_0$$

for every $A : \mathsf{U}$.

0-truncation

- The inclusion of Set in U has a left adjoint called 0-truncation.

$$\| - \|_0 : \mathsf{U} \to \mathsf{Set}.$$

With unit

$$| - |_0 : A \to \|A\|_0$$

for every $A : \mathsf{U}$.

- The universal property of 0-truncation is that
  - For every type $A : \mathsf{U}$,

0-truncation

- The inclusion of Set in U has a left adjoint called 0-truncation.

$$\| - \|_0 : \mathsf{U} \to \mathsf{Set}.$$

With unit

$$| - |_0 : A \to \|A\|_0$$

for every $A : \mathsf{U}$.

- The universal property of 0-truncation is that
  - For every type $A : \mathsf{U}$,
  - For every family $P : \|A\|_0 \to$ Set of sets over $\|A\|_0$,

0-truncation

- The inclusion of Set in U has a left adjoint called 0-truncation.

$$\| - \|_0 : \mathsf{U} \to \mathsf{Set}.$$

With unit

$$| - |_0 : A \to \|A\|_0$$

for every $A : \mathsf{U}$.

- The universal property of 0-truncation is that
  - For every type $A : \mathsf{U}$,
  - For every family $P : \|A\|_0 \to \mathsf{Set}$ of sets over $\|A\|_0$,
  - The *pre-composition* function

$$\lambda s.\, \lambda a.\, s(|a|_0) : \prod_{x:\|A\|_0} P(x) \to \prod_{a:A} P(|a|_0)$$

is an equivalence.

Higher inductive types

- The truncations are implemented as higher inductive types (Brunerie).

Higher inductive types

- The truncations are implemented as higher inductive types (Brunerie).
- We use higher inductive types to implement colimits.

Higher inductive types

- The truncations are implemented as higher inductive types (Brunerie).
- We use higher inductive types to implement colimits.
- In particular we will get:

Higher inductive types

- The truncations are implemented as higher inductive types (Brunerie).
- We use higher inductive types to implement colimits.
- In particular we will get:
  - Coequalizers

Higher inductive types

- The truncations are implemented as higher inductive types (Brunerie).
- We use higher inductive types to implement colimits.
- In particular we will get:
  - Coequalizers
  - Pushouts

Higher inductive types

- The truncations are implemented as higher inductive types (Brunerie).
- We use higher inductive types to implement colimits.
- In particular we will get:
  - Coequalizers
  - Pushouts
  - Quotients

Higher inductive types

- The truncations are implemented as higher inductive types (Brunerie).
- We use higher inductive types to implement colimits.
- In particular we will get:
  - Coequalizers
  - Pushouts
  - Quotients (only when we work with sets)

Quotients as HIT's

Suppose A is a type and $R : A \to A \to \mathrm{Prop}$ is an equivalence relation over A.

Quotients as HIT's

Suppose A is a type and $R : A \to A \to \mathrm{Prop}$ is an equivalence relation over A. We define A/R as a Higher Inductive Type with basic constructors:

Quotients as HIT's

Suppose A is a type and $R : A \to A \to \text{Prop}$ is an equivalence relation over A. We define A/R as a Higher Inductive Type with basic constructors:

$$\pi : A \to A/R$$
$$\text{paths} : \prod_{(x,y:A)} R(x,y) \to (\pi(x) = \pi(y))$$

and

$$\text{isSet} : \prod_{(x,y:A/R)} \prod_{(p,q:x=y)} (p = q)$$

Quotients as HIT's

Suppose A is a type and $R : A \to A \to \text{Prop}$ is an equivalence relation over A. We define A/R as a Higher Inductive Type with basic constructors:

$$\pi : A \to A/R$$
$$\text{paths} : \prod_{(x,y:A)} R(x,y) \to (\pi(x) = \pi(y))$$

and

$$\text{isSet} : \prod_{(x,y:A/R)} \prod_{(p,q:x=y)} (p = q)$$

To find a map out of A/R into a set X, the induction principle tells us that it suffices to find

$$f : A \to X$$
$$H : \prod_{(x,y:A)} R(x,y) \to (f(x) = f(y)).$$

Regularity

Definition

For a function $f : A \to B$, we define

$$\mathrm{im}(f) :\equiv \sum_{b:B} \|\mathrm{fib}_f(b)\|$$

Regularity

Definition

For a function $f : A \to B$, we define

$$\mathrm{im}(f) :\equiv \sum_{b:B} \|\mathrm{fib}_f(b)\|$$

Theorem

For any function $f : A \to B$ between sets, $\mathrm{im}(f)$ is the coequalizer of

$$\pi_1, \pi_2 : \sum_{x,y:A} (f(x) = f(y)) \to A.$$

Sketch of the proof

- First prove the Principle of Unique Choice.

Sketch of the proof

- First prove the Principle of Unique Choice.
- Also show that a function is an epimorphism if and only if it is surjective.

Sketch of the proof

- First prove the Principle of Unique Choice.
- Also show that a function is an epimorphism if and only if it is surjective. (predicative, using univalence)

Sketch of the proof

- First prove the Principle of Unique Choice.
- Also show that a function is an epimorphism if and only if it is surjective. (predicative, using univalence)
- The function $A \to \mathrm{im}(f)$ is surjective.

Sketch of the proof

- First prove the Principle of Unique Choice.
- Also show that a function is an epimorphism if and only if it is surjective. (predicative, using univalence)
- The function $A \to \mathrm{im}(f)$ is surjective.
- Hence it suffices to show that

Sketch of the proof

- First prove the Principle of Unique Choice.
- Also show that a function is an epimorphism if and only if it is surjective. (predicative, using univalence)
- The function $A \to \mathrm{im}(f)$ is surjective.
- Hence it suffices to show that

  for any function $g : A \to C$

Sketch of the proof

- First prove the Principle of Unique Choice.
- Also show that a function is an epimorphism if and only if it is surjective. (predicative, using univalence)
- The function $A \to \mathrm{im}(f)$ is surjective.
- Hence it suffices to show that

  for any function $g : A \to C$
  and any homotopy $g \circ \pi_1 \sim g \circ \pi_2$

Sketch of the proof

- First prove the Principle of Unique Choice.
- Also show that a function is an epimorphism if and only if it is surjective. (predicative, using univalence)
- The function $A \to \mathrm{im}(f)$ is surjective.
- Hence it suffices to show that

  for any function $g : A \to C$
  and any homotopy $g \circ \pi_1 \sim g \circ \pi_2$
  there is a function $h : \mathrm{im}(f) \to C$

Sketch of the proof

- First prove the Principle of Unique Choice.
- Also show that a function is an epimorphism if and only if it is surjective. (predicative, using univalence)
- The function $A \to \mathrm{im}(f)$ is surjective.
- Hence it suffices to show that

  for any function $g : A \to C$
  and any homotopy $g \circ \pi_1 \sim g \circ \pi_2$
  there is a function $h : \mathrm{im}(f) \to C$
  and a homotopy $h \circ f \sim g$.

Sketch of the proof

- First prove the Principle of Unique Choice.
- Also show that a function is an epimorphism if and only if it is surjective. (predicative, using univalence)
- The function $A \to \mathrm{im}(f)$ is surjective.
- Hence it suffices to show that

  for any function $g : A \to C$
  and any homotopy $g \circ \pi_1 \sim g \circ \pi_2$
  there is a function $h : \mathrm{im}(f) \to C$
  and a homotopy $h \circ f \sim g$.
- Define the function h with the Principle of Unique Choice.

Quotients

Definition

Suppose $R : A \to A \to \text{Prop}$ is a mere relation on A. We define $c_R : A \to A/R$ to be the coequalizer (in Set) of the pair

$$\pi_1, \pi_2 : \sum_{x,y:A} R(x, y) \to A.$$

Quotients

Definition

Suppose $R : A \to A \to \mathrm{Prop}$ is a mere relation on A. We define $c_R : A \to A/R$ to be the coequalizer (in Set) of the pair

$$\pi_1, \pi_2 : \sum_{x,y:A} R(x,y) \to A.$$

Definition

A mere relation $R : A \to A \to \mathrm{Prop}$ is said to be effective if

$$\begin{array}{ccc} \sum_{(x,y:A)} R(x,y) & \xrightarrow{\pi_1} & A \\ \downarrow{\scriptstyle \pi_2} & & \downarrow{\scriptstyle c_R} \\ A & \xrightarrow[c_R]{} & A/R \end{array}$$

is a pullback square.

Theorem

The mere effective relations are precisely the mere equivalence relations.

Theorem

The mere effective relations are precisely the mere equivalence relations.

Theorem

Set *is a* ΠW*-pretopos.*

(Sub)object classifier

Without any further assumptions, we have *no proof* that Set is a predicative topos. However, we can prove that

(Sub)object classifier

Without any further assumptions, we have *no proof* that Set is a predicative topos. However, we can prove that

Theorem

For any type $B : \mathsf{U}$*, there is an equivalence*

$$(B \to \mathsf{U}) \simeq \sum_{A:\mathsf{U}} (A \to B)$$

Sketch of the proof

- Define $\varphi : (B \to \mathsf{U}) \to \sum_{(A:\mathsf{U})}(A \to B)$ by

$$\varphi :\equiv \lambda P.\, \langle \textstyle\sum_{(b:B)} P(b), \mathsf{pr}_1 \rangle$$

Sketch of the proof

- Define $\varphi : (B \to \mathsf{U}) \to \sum_{(A:\mathsf{U})}(A \to B)$ by

$$\varphi :\equiv \lambda P.\, \langle \textstyle\sum_{(b:B)} P(b), \mathsf{pr}_1 \rangle$$

- Define the function $\psi : \sum_{(A:\mathsf{U})}(A \to B) \to (B \to \mathsf{U})$ by

$$\psi :\equiv \lambda \langle A, f \rangle.\, \lambda b.\, \mathsf{fib}_f(b)$$

Sketch of the proof

- Define $\varphi : (B \to \mathsf{U}) \to \sum_{(A:\mathsf{U})}(A \to B)$ by

$$\varphi :\equiv \lambda P.\, \langle \textstyle\sum_{(b:B)} P(b), \mathsf{pr}_1 \rangle$$

- Define the function $\psi : \sum_{(A:\mathsf{U})}(A \to B) \to (B \to \mathsf{U})$ by

$$\psi :\equiv \lambda \langle A, f \rangle.\, \lambda b.\, \mathsf{fib}_f(b)$$

- Show that these functions are each other's inverses (using Univalence)

In fact we can show:

Theorem

The function $\mathrm{pr}_1 : \sum_{(X:\mathsf{U})} X \to \mathsf{U}$ *is the object classifier for* U.

In fact we can show:

Theorem

The function $\mathrm{pr}_1 : \sum_{(X:\mathsf{U})} X \to \mathsf{U}$ *is the object classifier for* U.

Theorem

The function $\mathrm{pr}_1 : \sum_{(X:n\text{-}\mathrm{Type}_{\mathsf{U}})} X \to n\text{-}\mathrm{Type}_{\mathsf{U}}$ *is the object classifier for* $n\text{-}\mathrm{Type}_{\mathsf{U}}$.

In fact we can show:

Theorem

The function $\mathrm{pr}_1 : \sum_{(X:\mathsf{U})} X \to \mathsf{U}$ *is the object classifier for* U.

Theorem

The function $\mathrm{pr}_1 : \sum_{(X:n\text{-}\mathrm{Type}_{\mathsf{U}})} X \to n\text{-}\mathrm{Type}_{\mathsf{U}}$ *is the object classifier for* $n\text{-}\mathrm{Type}_{\mathsf{U}}$.

And in particular

Theorem

The function $\mathrm{pr}_1 : \sum_{(X:\mathsf{Set})} X \to \mathsf{Set}$ *is the object classifier for* Set.
The function $\mathrm{pr}_1 : \sum_{(X:\mathsf{Prop})} X \to \mathsf{Prop}$ *is the subobject classifier for* Set.

Using $\mathrm{pr}_1 : \sum_{(X:\mathrm{Set})} X \to \mathrm{Set}$, we obtain a class $\mathcal{S}$ of small maps, by the methods of algebraic set theory.

Using $\mathrm{pr}_1 : \sum_{(X:\mathrm{Set})} X \to \mathrm{Set}$, we obtain a class $\mathcal{S}$ of small maps, by the methods of algebraic set theory.
Writing $\mathcal{S}(f)$ if f is in $\mathcal{S}$, the class $\mathcal{S}$ satisfies:

Using $\mathrm{pr}_1 : \sum_{(X:\mathrm{Set})} X \to \mathrm{Set}$, we obtain a class $\mathcal{S}$ of small maps, by the methods of algebraic set theory.
Writing $\mathcal{S}(f)$ if f is in $\mathcal{S}$, the class $\mathcal{S}$ satisfies:

- The pullback of a map in $\mathcal{S}$ is again in $\mathcal{S}$.

Using $\mathrm{pr}_1 : \sum_{(X:\mathrm{Set})} X \to \mathrm{Set}$, we obtain a class $\mathcal{S}$ of small maps, by the methods of algebraic set theory.
Writing $\mathcal{S}(f)$ if f is in $\mathcal{S}$, the class $\mathcal{S}$ satisfies:

- The pullback of a map in $\mathcal{S}$ is again in $\mathcal{S}$.
- When g is the pullback of f along a surjective map, then $\mathcal{S}(g) \to \mathcal{S}(f)$.

Using $\mathrm{pr}_1 : \sum_{(X:\mathrm{Set})} X \to \mathrm{Set}$, we obtain a class $\mathcal{S}$ of small maps, by the methods of algebraic set theory.
Writing $\mathcal{S}(f)$ if f is in $\mathcal{S}$, the class $\mathcal{S}$ satisfies:

- The pullback of a map in $\mathcal{S}$ is again in $\mathcal{S}$.
- When g is the pullback of f along a surjective map, then $\mathcal{S}(g) \to \mathcal{S}(f)$.
- $\mathcal{S}(f) \to \mathcal{S}(g) \to \mathcal{S}(f + g)$.

Using $\mathrm{pr}_1 : \sum_{(X:\mathrm{Set})} X \to \mathrm{Set}$, we obtain a class $\mathcal{S}$ of small maps, by the methods of algebraic set theory.
Writing $\mathcal{S}(f)$ if f is in $\mathcal{S}$, the class $\mathcal{S}$ satisfies:

- The pullback of a map in $\mathcal{S}$ is again in $\mathcal{S}$.
- When g is the pullback of f along a surjective map, then $\mathcal{S}(g) \to \mathcal{S}(f)$.
- $\mathcal{S}(f) \to \mathcal{S}(g) \to \mathcal{S}(f + g)$.
- $\mathcal{S}$ is locally full.

Using $\mathrm{pr}_1 : \sum_{(X:\mathrm{Set})} X \to \mathrm{Set}$, we obtain a class $\mathcal{S}$ of small maps, by the methods of algebraic set theory.
Writing $\mathcal{S}(f)$ if f is in $\mathcal{S}$, the class $\mathcal{S}$ satisfies:

- The pullback of a map in $\mathcal{S}$ is again in $\mathcal{S}$.
- When g is the pullback of f along a surjective map, then $\mathcal{S}(g) \to \mathcal{S}(f)$.
- $\mathcal{S}(f) \to \mathcal{S}(g) \to \mathcal{S}(f + g)$.
- $\mathcal{S}$ is locally full.

However, we have no proof of the collection axiom for $\mathcal{S}$.

Using $\mathrm{pr}_1 : \sum_{(X:\mathrm{Set})} X \to \mathrm{Set}$, we obtain a class $\mathcal{S}$ of small maps, by the methods of algebraic set theory.
Writing $\mathcal{S}(f)$ if f is in $\mathcal{S}$, the class $\mathcal{S}$ satisfies:

- The pullback of a map in $\mathcal{S}$ is again in $\mathcal{S}$.
- When g is the pullback of f along a surjective map, then $\mathcal{S}(g) \to \mathcal{S}(f)$.
- $\mathcal{S}(f) \to \mathcal{S}(g) \to \mathcal{S}(f + g)$.
- $\mathcal{S}$ is locally full.

However, we have no proof of the collection axiom for $\mathcal{S}$.
Conjecture: higher inductive types can be used instead.
Theorem: We do have collection in the Aczel encoding of sets.

Conclusion

Sets form a ΠW-pretopos with an object classifier:
well-behaved:

- class of proof irrelevant types (Prop)
- squash types
- quotient types
- equivalence between Fam and Pow