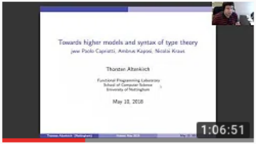
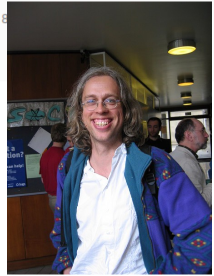
Towards the syntax and semantics of higher dimensional type theory

~~Thorsten Altenkirch~~
Nicolai Kraus
Oxford, HoTT/UF'18, 8 July



Thorsten Altenkirch, Towards higher models and syntax of type theory
Dan Christensen • 106 views • 1 month ago
Homotopy Type Theory Electronic Seminar Talks, 2018



[picture by Andrej Bauer, (CC BY-SA 2.5 SI)]

The goal: type theory in type theory

Plan: develop the metatheory of type theory *in* type theory
Why?

- A foundation should be able to model itself.
- "Template meta-programming", this problem is in some sense universal.
- Specify HITs.
- . . . ?

The goal: type theory in type theory

type signatures

HIIT

constructors

$$\mathrm{Con} : \mathcal{U}$$
$$\mathrm{Ty} : \quad \mathrm{Con} \to \mathcal{U}$$
$$\mathrm{Tm} : \quad \Pi\Gamma : \mathrm{Con}.\mathrm{Ty}(\Gamma) \to \mathcal{U}$$
$$\mathrm{Tms} : \mathrm{Con} \to \mathrm{Con} \to \mathcal{U}$$
$$\vdots$$
$$\mathrm{Pi} : \quad \Pi A : \mathrm{Ty}(\Gamma), B : \mathrm{Ty}(\Gamma.A).\mathrm{Ty}(\Gamma)$$
$$\vdots$$
$$\mathrm{lam} : \quad \mathrm{Tm}(\Gamma.A, B) \to \mathrm{Tm}(\Gamma, \mathrm{Pi}(A, B))$$
$$\mathrm{app} : \quad \mathrm{Tm}(\Gamma, \mathrm{Pi}(A, B)) \to \mathrm{Tm}(\Gamma.A, B)$$
$$\vdots$$
$$\beta : \quad \Pi t : \mathrm{Tm}(\Gamma.A, B).\mathrm{app}(\mathrm{lam}(t)) = t$$

Past work. . .

Altenkirch-Kaposi, POPL 2016:
Type theory in type theory using quotient inductive types

But this was done assuming UIP/K. How to do it in HoTT?

Past work...

Altenkirch-Kaposi, POPL 2016:
Type theory in type theory using quotient inductive types

But this was done assuming UIP/K. How to do it in HoTT?

Why not just set-truncate everything?

Breaks when we want to define the “standard model”, i.e. functions

$$\mathit{con} : \mathrm{Con} \to \mathcal{U}$$
$$\mathit{ty} : (\Gamma : \mathrm{Con}) \to \mathit{con}(\Gamma) \to \mathrm{Ty}(\gamma) \to \mathcal{U}$$
$$\mathit{tms} : \ldots$$
$$\mathit{tm} : \ldots$$

Categories with families

A category with families (CwF) is given by:

- A category of contexts and substitutions $\mathbf{Con}$.
- A presheaf of types $\mathbf{Ty} : \mathbf{Con}^{\mathrm{op}} \to \mathcal{U}$
- A presheaf of terms over contexts and types $\int \mathsf{Ty}^{\mathrm{op}} \to \mathcal{U}$
- ...

The "quotient inductive-inductive type" (QIIT) from before defines the initial CwF.

Categories with families

A category with families (CwF) is given by:

- A category of contexts and substitutions $\mathbf{Con}$.
- A presheaf of types $\mathbf{Ty} : \mathbf{Con}^{\mathrm{op}} \to \mathcal{U}$
- A presheaf of terms over contexts and types $\int \mathbf{Ty}^{\mathrm{op}} \to \mathcal{U}$
- ...

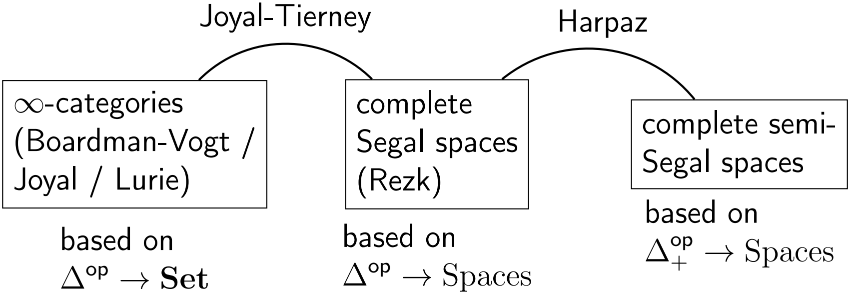
The "quotient inductive-inductive type" (QIIT) from before defines the initial CwF.

Thorsten's plan for the "HoTT in HoTT" problem:
Just replace "category" by "$(\infty, 1)$-category" and replace all notions by the relevant ∞-notions. The syntax will still be a set because the sytax will still have decidable equality. Done.

The End (of the part where I talk about Thorsten's ideas).

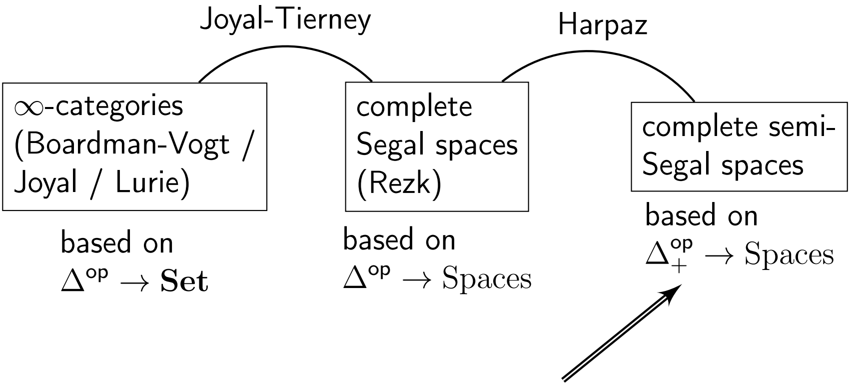
What are ∞-categories in HoTT?

Independently of type theory:



What are ∞-categories in HoTT?

Independently of type theory:



Works well in type theory!

Capriotti-Kraus, POPL 2018:
Higher univalent categories
via complete semi-Segal types

How do complete semi-Segal types work?

- First, we need $A : \Delta_+^{\mathsf{op}} \to \mathcal{U}$; encoding the *Reedy fibrant* ones is very natural in type theory (*semisimplicial types*):

  $A_0 : \mathcal{U}$

  $A_1 : A_0 \to A_0 \to \mathcal{U}$

  $A_2 : (x, y, z : A_0) \to A_1(x, y) \to A_1(y, z) \to A_1(x, z) \to \mathcal{U}$

How do complete semi-Segal types work?

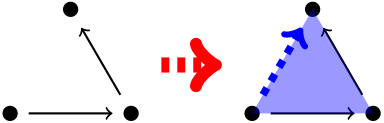
- First, we need $A : \Delta_+^{\mathsf{op}} \to \mathcal{U}$; encoding the *Reedy fibrant* ones is very natural in type theory (*semisimplicial types*):

  $A_0 : \mathcal{U}$

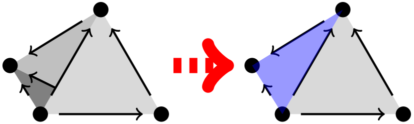
  $A_1 : A_0 \to A_0 \to \mathcal{U}$

  $A_2 : (x, y, z : A_0) \to A_1(x, y) \to A_1(y, z) \to A_1(x, z) \to \mathcal{U}$

- Add the *Segal condition*: For any inner horn, the type of fillers is contractible.



(composition!)



(associativity!)

How do complete semi-Segal types work? (2)

- Identities via Harpaz' (Lurie's) trick.

  Definition: $f : A_1(x, y)$ is an *equivalence* if $- \circ f$ and $f \circ -$ are equivalences.

  Condition: exactly one outgoing equivalence for every object.

$$\Pi x : A_0, \mathsf{isContr}(\Sigma(y : A_0), (e : A_1(x, y)), \mathsf{isequiv}(e))$$

How do complete semi-Segal types work? (2)

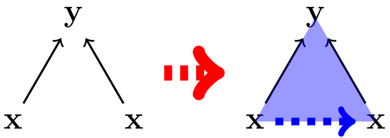
- Identities via Harpaz' (Lurie's) trick.

  Definition: $f : A_1(x, y)$ is an *equivalence* if $- \circ f$ and $f \circ -$ are equivalences.

  Condition: exactly one outgoing equivalence for every object.

  $$\Pi x : A_0, \mathsf{isContr}(\Sigma(y : A_0), (e : A_1(x, y)), \mathsf{isequiv}(e))$$

- Construct identities:



(identity found!)

How do complete semi-Segal types work? (2)

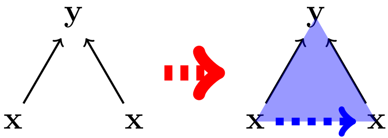
- Identities via Harpaz' (Lurie's) trick.

  Definition: $f : A_1(x, y)$ is an *equivalence* if $- \circ f$ and $f \circ -$ are equivalences.

  Condition: exactly one outgoing equivalence for every object.

$$\Pi x : A_0, \mathsf{isContr}(\Sigma(y : A_0), (e : A_1(x, y)), \mathsf{isequiv}(e))$$

- Construct identities:



(identity found!)

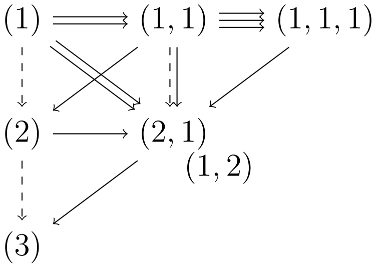
⇒ This gives **univalent** $(\infty, 1)$-categories. (Can remove univalence by removing isContr.)

What if we want an explicit identity structure?

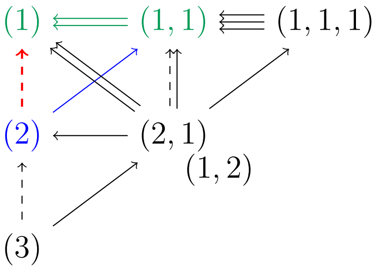
Try again to define *simplicial types*. Two possibilities are given in:

Kraus-Sattler 2017:
Space-valued diagrams, type-theoretically

Possibility 1: a *direct replacement* of Δ which is finite if restricted to finite levels.



Direct replacement of Δ



$A_{(1)} : \mathcal{U}$

$A_{(1,1)} : A_{(1)} \to A_{(1)} \to \mathcal{U}$

$A_{(1,1,1)} : (x, y, z : A_{(1)}) \to A_{(1,1)}(x, y)$

$\qquad \to A_{(1,1)}(y, z) \to A_{(1,1)}(x, z) \to \mathcal{U}$

$A_{(2)} : (x : A_{(1)}) \to A_{(1,1)}(x, x) \to \mathcal{U}$

$h_{(2)} : (x : A_{(1)}) \to \mathsf{isContr}(\Sigma(l : A_{(1,1)}(x, x), A_{(2)}(x, l)))$

$\ldots\ldots\ldots\ldots$

Homotopy coherent diagrams

Second possibility to get "simplicial types": Write down functors $\Delta^{\mathsf{op}} \to \mathcal{U}$ with **all** coherences.

Homotopy coherent diagrams

Second possibility to get "simplicial types": Write down functors $\Delta^{\mathsf{op}} \to \mathcal{U}$ with **all** coherences.

This can be done by looking at the nerve of Δ^{op}:

- a type for every $[n] : \Delta^{\mathsf{op}}$
- a function for every $[n] \xrightarrow{f} [m]$
- a commutative triangle for every $[n] \xrightarrow{f} [m] \xrightarrow{g} [k]$
- a tetrahedron for every $[n] \xrightarrow{f} [m] \xrightarrow{g} [k] \xrightarrow{h} [j]$
- ......

Note: Similar constructions have been used before for higher categories ("D construction"), e.g. Rădulescu-Banu'09, Szumiło'14.

Homotopy coherent diagrams

Second possibility to get "simplicial types": Write down functors $\Delta^{\mathsf{op}} \to \mathcal{U}$ with **all** coherences.

This can be done by looking at the nerve of Δ^{op}:

- a type for every $[n] : \Delta^{\mathsf{op}}$
- a function for every $[n] \xrightarrow{f} [m]$
- a commutative triangle for every $[n] \xrightarrow{f} [m] \xrightarrow{g} [k]$
- a tetrahedron for every $[n] \xrightarrow{f} [m] \xrightarrow{g} [k] \xrightarrow{h} [j]$
- ......

Note: Similar constructions have been used before for higher categories ("D construction"), e.g. Rădulescu-Banu'09, Szumiło'14.

- **plus**: every $[n] \xrightarrow{id} [n]$ is mapped to an equivalence

Higher categories without univalence

Result:
These two notions of *simplicial types* are equivalent.

We can use either of them to define $(\infty, 1)$-categories (without built-in univalence).

Higher categories without univalence

Result:
These two notions of *simplicial types* are equivalent.

We can use either of them to define $(\infty, 1)$-categories (without built-in univalence).

Now we can go back and attempt to construct what Thorsten suggested.

The End (of the talk).

References

Thorsten Altenkirch and Ambrus Kaposi.
Type theory in type theory using quotient inductive types.
POPL'16, 2016.

Danil Annenkov, Paolo Capriotti, and Nicolai Kraus.
Two-level type theory and applications.
ArXiv e-prints, 2017.

Paolo Capriotti and Nicolai Kraus.
Univalent higher categories via complete semi-segal types.
POPL'18, 2017.

Yonatan Harpaz.
Quasi-unital ∞–categories.
Algebraic & Geometric Topology, 2015.

André Joyal.
The theory of quasi-categories and its applications.
2008.

André Joyal and Myles Tierney.
Quasi-categories vs segal spaces.
Contemp. Math, 2006.

Nicolai Kraus and Christian Sattler.
Space-valued diagrams, type-theoretically.
ArXiv e-prints, 2017.

Jacob Lurie.
Higher Topos Theory.
2009.