Internal parametricity, without an interval

Ambrus Kaposi

Eötvös Loránd University, Budapest, Hungary

j.w.w. Thorsten Altenkirch, Yorgo Chamoun and Michael Shulman

POPL
London
19 January 2024

Identity type in type theory

$$? : \mathsf{Id}_{\mathbb{N}}\,(1 + 1)\,2$$

Identity type in type theory

$$? : \mathsf{Id}_{\mathbb{N}} \quad 2 \quad 2$$

Identity type in type theory

$$\mathsf{refl}_2 : \mathsf{Id}_{\mathbb{N}}\,(1+1)\,2$$

Identity type in type theory

$$\mathsf{refl}_2 : \mathsf{Id}_{\mathbb{N}}\,(1+1)\,2$$

$$? : \mathsf{Id}_{\mathbb{N}}\,(x+0)\,x$$

Identity type in type theory

$$\mathsf{refl}_2 : \mathsf{Id}_{\mathbb{N}}\,(1+1)\,2$$

$$? : \mathsf{Id}_{\mathbb{N}}\,(x+0)\,x$$

$$\begin{aligned} 0 \qquad + b &:= b \\ \mathsf{suc}\,a + b &:= \mathsf{suc}\,(a+b) \end{aligned}$$

Identity type in type theory

$$\mathsf{refl}_2 : \mathsf{Id}_{\mathbb{N}}\,(1+1)\,2$$

$$\mathsf{ind}_{\mathbb{N}}\,(\ldots)\,x : \mathsf{Id}_{\mathbb{N}}\,(x+0)\,x$$

$$\begin{aligned} 0 \qquad + b &:= b \\ \mathsf{suc}\,a + b &:= \mathsf{suc}\,(a+b) \end{aligned}$$

Identity type in type theory

$$\mathsf{refl}_2 : \mathsf{Id}_{\mathbb{N}}\,(1+1)\,2$$

$$\mathsf{ind}_{\mathbb{N}}\,(\dots)\,x : \mathsf{Id}_{\mathbb{N}}\,(x+0)\,x$$

$$\begin{aligned} 0 \quad\quad + b &:= b \\ \mathsf{suc}\,a + b &:= \mathsf{suc}\,(a+b) \end{aligned}$$

In general:

$$\frac{A : \mathsf{Type}}{\mathsf{Id}_A : A \to A \to \mathsf{Type}} \qquad \frac{a : A}{\mathsf{refl}_a : \mathsf{Id}_A\,a\,a} \qquad \dots$$

What kind of type is Id?

- Per Martin-Löf: inductively, eliminator J

What kind of type is Id?

- Per Martin-Löf: inductively, eliminator J
  - no element of $\mathsf{Id}_{\mathbb{N}\to\mathbb{N}}\,(\lambda x.x+0)\,(\lambda x.x)$

What kind of type is Id?

- Per Martin-Löf: inductively, eliminator J
  - no element of $\mathsf{Id}_{\mathbb{N}\to\mathbb{N}}\,(\lambda x.x+0)\,(\lambda x.x)$
- Cubical type theory: $\mathsf{Id}_A\,a\,b := (f : \mathbb{I} \to A) \times (f\,0 = a) \times (f\,1 = b)$

What kind of type is Id?

- Per Martin-Löf: inductively, eliminator J
  - no element of $\mathrm{Id}_{\mathbb{N}\to\mathbb{N}}\,(\lambda x.x+0)\,(\lambda x.x)$
- Cubical type theory: $\mathrm{Id}_A\,a\,b := (f : \mathbb{I} \to A) \times (f\,0 = a) \times (f\,1 = b)$
- Observational type theory: by computation

What kind of type is Id?

- Per Martin-Löf: inductively, eliminator J
  - no element of $\mathsf{Id}_{\mathbb{N}\to\mathbb{N}}\,(\lambda x.x+0)\,(\lambda x.x)$
- Cubical type theory: $\mathsf{Id}_A\,a\,b := (f : \mathbb{I} \to A) \times (f\,0 = a) \times (f\,1 = b)$
- Observational type theory: by computation

$$\begin{aligned}
\mathsf{Id}_{\mathbb{N}}\,0 \quad & 0 && := \top \\
\mathsf{Id}_{\mathbb{N}}\,(\mathsf{suc}\,m)\,& (\mathsf{suc}\,n) && := \mathsf{Id}_N\,m\,n \\
\mathsf{Id}_{\mathbb{N}}\,0 \quad & (\mathsf{suc}\,n) && := \bot \\
\mathsf{Id}_{\mathbb{N}}\,(\mathsf{suc}\,n)\,& 0 && := \bot
\end{aligned}$$

What kind of type is Id?

- Per Martin-Löf: inductively, eliminator J
  - no element of $\mathsf{Id}_{\mathbb{N}\to\mathbb{N}}\,(\lambda x.x+0)\,(\lambda x.x)$
- Cubical type theory: $\mathsf{Id}_A\,a\,b :\equiv (f : \mathbb{I} \to A) \times (f\,0 = a) \times (f\,1 = b)$
- Observational type theory: by computation

$$\mathsf{Id}_{A\times B}\,(a_0, b_0)\,(a_1, b_1) :\equiv \mathsf{Id}_A\,a_0\,a_1 \times \mathsf{Id}_B\,b_0\,b_1$$

What kind of type is Id?

- Per Martin-Löf: inductively, eliminator J
  - no element of $\mathsf{Id}_{\mathbb{N}\to\mathbb{N}}\,(\lambda x.x+0)\,(\lambda x.x)$
- Cubical type theory: $\mathsf{Id}_A\,a\,b :\equiv (f : \mathbb{I} \to A) \times (f\,0 = a) \times (f\,1 = b)$
- Observational type theory: by computation

$$\mathsf{Id}_{A\to B}\,f_0\,f_1 :\equiv \forall a_0\,a_1\,.\,\mathsf{Id}_A\,a_0\,a_1 \to \mathsf{Id}_B\,(f_0\,a_0)\,(f_1\,a_0)$$

What kind of type is Id?

- Per Martin-Löf: inductively, eliminator J
  - no element of $\mathsf{Id}_{\mathbb{N}\to\mathbb{N}}\,(\lambda x.x+0)\,(\lambda x.x)$
- Cubical type theory: $\mathsf{Id}_A\,a\,b := (f : \mathbb{I} \to A) \times (f\,0 = a) \times (f\,1 = b)$
- Observational type theory: by computation
- Higher Observational type theory (WORK IN PROGRESS): by computation and
$$\mathsf{Id}_{\mathsf{Type}}\,A\,B := (A \simeq B)$$

What kind of type is Id?

- Per Martin-Löf: inductively, eliminator J
  - no element of $\mathrm{Id}_{\mathbb{N}\to\mathbb{N}}\,(\lambda x.x+0)\,(\lambda x.x)$
- Cubical type theory: $\mathrm{Id}_A\,a\,b := (f : \mathbb{I} \to A) \times (f\,0 = a) \times (f\,1 = b)$
- Observational type theory: by computation
- Higher Observational type theory (WORK IN PROGRESS): by computation and

$$\mathrm{Id}_{\mathsf{Type}}\,A\,B := (A \simeq B)$$

  Promises:

  - explainable: no interval, only low dimensional operations
  - computational univalence (unlike C.T.T.)
  - efficient (?): a simple extension of Martin-Löf's type theory

This paper: a baby version of H.O.T.T.

- Not identity, only logical relation

This paper: a baby version of H.O.T.T.

- Not identity, only logical relation
  - a reflexive Id type which is a congruence, but not symmetric or transitive

This paper: a baby version of H.O.T.T.

- Not identity, only logical relation
  - a reflexive Id type which is a congruence, but not symmetric or transitive
  - Voevodsky's univalence: everything preserves isomorphisms
  - Reynolds' parametricity: everything preserves relations

This paper: a baby version of H.O.T.T.

- Not identity, only logical relation
  - a reflexive Id type which is a congruence, but not symmetric or transitive
  - Voevodsky's univalence: everything preserves isomorphisms
  - Reynolds' parametricity: everything preserves relations
- not everything is computational

This paper: a baby version of H.O.T.T.

- Not identity, only logical relation
  - a reflexive Id type which is a congruence, but not symmetric or transitive
  - Voevodsky's univalence: everything preserves isomorphisms
  - Reynolds' parametricity: everything preserves relations
- not everything is computational
  - "$\mathsf{Id}_{A\to B} \cong \mathsf{Id}_A \to \mathsf{Id}_B$"

This paper: a baby version of H.O.T.T.

- Not identity, only logical relation
  - a reflexive Id type which is a congruence, but not symmetric or transitive
  - Voevodsky's univalence: everything preserves isomorphisms
  - Reynolds' parametricity: everything preserves relations
- not everything is computational
  - "$\mathrm{Id}_{A \to B} \cong \mathrm{Id}_A \to \mathrm{Id}_B$"
  - $\mathrm{Id}_{\mathrm{Type}}\, A\, B \ncong (A \to B \to \mathrm{Type})$, only up to section-retraction

This paper: a baby version of H.O.T.T.

- Not identity, only logical relation
  - a reflexive Id type which is a congruence, but not symmetric or transitive
  - Voevodsky's univalence: everything preserves isomorphisms
  - Reynolds' parametricity: everything preserves relations
- not everything is computational
  - "$\mathrm{Id}_{A\to B} \cong \mathrm{Id}_A \to \mathrm{Id}_B$"
  - $\mathrm{Id}_{\mathrm{Type}}\, A\, B \ncong (A \to B \to \mathrm{Type})$, only up to section-retraction
- not logical relation, only logical span

This paper: a baby version of H.O.T.T.

- Not identity, only logical relation
  - a reflexive Id type which is a congruence, but not symmetric or transitive
  - Voevodsky's univalence: everything preserves isomorphisms
  - Reynolds' parametricity: everything preserves relations
- not everything is computational
  - "$\mathsf{Id}_{A \to B} \cong \mathsf{Id}_A \to \mathsf{Id}_B$"
  - $\mathsf{Id}_{\mathsf{Type}}\, A\, B \ncong (A \to B \to \mathsf{Type})$, only up to section-retraction
- not logical relation, only logical span
  - instead of $\mathsf{Id}_A : A \to A \to \mathsf{Type}$ we have $A \xleftarrow{0_A} \forall A \xrightarrow{1_A} A$

This paper: a baby version of H.O.T.T.

- Not identity, only logical relation
  - a reflexive Id type which is a congruence, but not symmetric or transitive
  - Voevodsky's univalence: everything preserves isomorphisms
  - Reynolds' parametricity: everything preserves relations
- not everything is computational
  - "$\mathrm{Id}_{A\to B} \cong \mathrm{Id}_A \to \mathrm{Id}_B$"
  - $\mathrm{Id}_{\mathrm{Type}}\, A\, B \ncong (A \to B \to \mathrm{Type})$, only up to section-retraction
- not logical relation, only logical span
  - instead of $\mathrm{Id}_A : A \to A \to \mathrm{Type}$ we have $A \xleftarrow{0_A} \forall A \xrightarrow{1_A} A$
- For now we see things in a glass, darkly; but then face to face. (1 Cor 13:12)

This paper: a baby version of H.O.T.T.

- Not identity, only logical relation
  - a reflexive Id type which is a congruence, but not symmetric or transitive
  - Voevodsky's univalence: everything preserves isomorphisms
  - Reynolds' parametricity: everything preserves relations
- not everything is computational
  - "$\mathsf{Id}_{A\to B} \cong \mathsf{Id}_A \to \mathsf{Id}_B$"
  - $\mathsf{Id}_{\mathsf{Type}}\, A\, B \ncong (A \to B \to \mathsf{Type})$, only up to section-retraction
- not logical relation, only logical span
  - instead of $\mathsf{Id}_A : A \to A \to \mathsf{Type}$ we have $A \xleftarrow{0_A} \forall A \xrightarrow{1_A} A$
- For now we see things in a glass, darkly; but then face to face. (1 Cor 13:12)
  - explainability, computation, simple extension

Semantics

The semantics is Bezem-Coquand-Huber cubes

Semantics

The semantics is Bezem-Coquand-Huber cubes

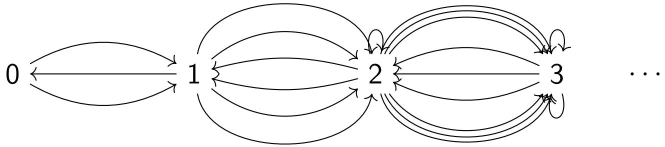
- the first constructive model of univalence (TYPES 2013)
- no cubical type theory based on it: the interval is substructural

Semantics

The semantics is Bezem-Coquand-Huber cubes

- the first constructive model of univalence (TYPES 2013)
- no cubical type theory based on it: the interval is substructural

The category of cubes:

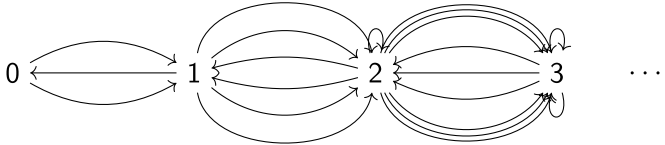


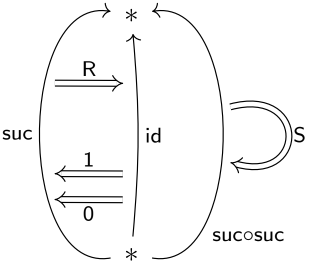
Semantics

The semantics is Bezem-Coquand-Huber cubes

- the first constructive model of univalence (TYPES 2013)
- no cubical type theory based on it: the interval is substructural

The category of cubes:



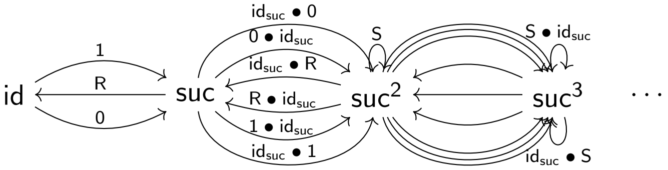


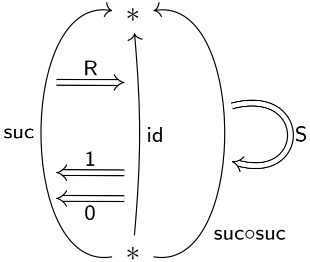
Semantics

The semantics is Bezem-Coquand-Huber cubes

- the first constructive model of univalence (TYPES 2013)
- no cubical type theory based on it: the interval is substructural

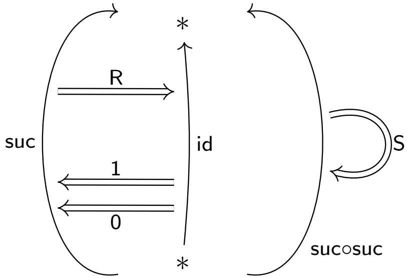
The category of cubes:





Syntax from semantics

The cube category $\square$:



Syntax from semantics

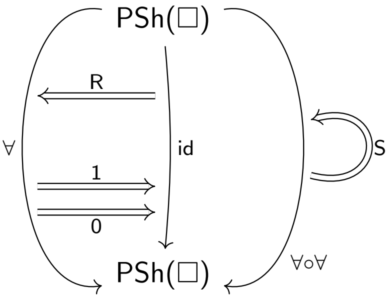
Structure on presheaves
over $\square$:

$$
\begin{tikzcd}[row sep=huge, column sep=huge]
\mathrm{PSh}(\square) \arrow[d, bend right=80, "\mathrm{suc}^*"'] \arrow[d, "\mathrm{id}"] \arrow[d, bend left=80, "\mathrm{suc}^* \circ \mathrm{suc}^*"] \arrow[d, phantom, shift right=6ex, "\overset{R}{\Leftarrow} \; \overset{1}{\Rightarrow} \; \underset{0}{\Rightarrow}"] \\
\mathrm{PSh}(\square)
\end{tikzcd}
$$

S

Syntax from semantics

Structure on presheaves over $\square$:



A new kind of modal theory (substructural):

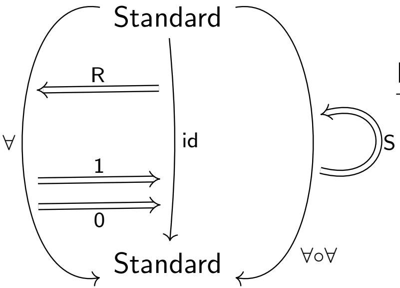
$$\frac{\vdash \Gamma}{\vdash \forall\Gamma} \qquad \frac{\sigma : \Delta \Rightarrow \Gamma}{\forall\sigma : \forall\Delta \Rightarrow \forall\Gamma}$$

$$\frac{\Gamma \vdash A}{\forall\Gamma \vdash \forall A} \qquad \frac{\Gamma \vdash t : A}{\forall\Gamma \vdash \forall t : \forall A}$$

$$\frac{\vdash \Gamma}{\begin{array}{c} \mathsf{R}_\Gamma : \Gamma \Rightarrow \forall\Gamma \\ \mathsf{k}_\Gamma : \forall\Gamma \Rightarrow \Gamma \\ \mathsf{S}_\Gamma : \forall\forall\Gamma \Rightarrow \forall\forall\Gamma \end{array}}$$

Syntax from semantics

Structure on the standard model internal to PSh(□):



Our final theory (structural):

$$\frac{\Gamma \vdash A}{\Gamma \vdash \forall A} \qquad \frac{\Gamma \vdash f : A \to B}{\Gamma \vdash \mathrm{ap}\, f : \forall A \to \forall B}$$

$$\frac{\Gamma, x : A \vdash B \quad a_2 : \forall A}{\Gamma \vdash \forall\!\mathrm{d}(x.B)\, a_2} \qquad \frac{\Gamma \vdash t : \Pi(x : A).B}{\Gamma \vdash \mathrm{apd}\, t : \Pi(a_2 : \forall A).\forall\!\mathrm{d}(x.B)\, a_2}$$

$$\frac{\Gamma \vdash A}{\begin{array}{c}\Gamma \vdash \mathrm{R_A} : A \to \forall A \\ \Gamma \vdash \mathrm{k_A} : \forall A \to A \\ \Gamma \vdash \mathrm{S_A} : \forall\!\!\forall A \to \forall\!\!\forall A\end{array}}$$

Summary

- We defined a type theory with internal parametricity

Summary

- We defined a type theory with internal parametricity
- Applications:
  - polymorphic identity function example
  - Church encoded naturals have support induction

Summary

- We defined a type theory with internal parametricity
- Applications:
  - polymorphic identity function example
  - Church encoded naturals have support induction
- First structural type theory for BCH-cubes.

Summary

- We defined a type theory with internal parametricity
- Applications:
  - polymorphic identity function example
  - Church encoded naturals have support induction
- First structural type theory for BCH-cubes.
- Geometry is emergent, rather than built-in.

Summary

- We defined a type theory with internal parametricity
- Applications:
  - polymorphic identity function example
  - Church encoded naturals have support induction
- First structural type theory for BCH-cubes.
- Geometry is emergent, rather than built-in.
- We proved canonicity: every closed boolean is convertible to true or false.

Summary

- We defined a type theory with internal parametricity
- Applications:
  - polymorphic identity function example
  - Church encoded naturals have support induction
- First structural type theory for BCH-cubes.
- Geometry is emergent, rather than built-in.
- We proved canonicity: every closed boolean is convertible to true or false.
- Ongoing and future work:
  - Prove normalisation
  - Replace spans by relations (Reedy fibrancy)
  - Add Kan operations = transport rule = symmetry, transitivity of Id
  - Implementation