Cubical Syntax for Reflection-Free Extensional Equality

FSCD 2019

Jonathan Sterling[1] Carlo Angiuli[1] Daniel Gratzer[2]

[1]Carnegie Mellon University

[2]Aarhus University

dependent type theory

... is about *families*!

dependent type theory

… is about *families*!

$$x : A \vdash B(x) \ \textit{type}$$

dependent type theory

... is about *families*!

$$x : A \vdash B(x) \textit{ type}$$

$$x : A \vdash M(x) : B(x)$$

dependent type theory

... is about *families*!

$$x : A \vdash B(x) \ \textit{type}$$

$$x : A \vdash M(x) : B(x)$$

need to consider equality of types: if $A = B$ *type*, then elements of A should be elements of B.

dependent type theory

... is about *families*!

$$x : A \vdash B(x) \; \textit{type}$$

$$x : A \vdash M(x) : B(x)$$

need to consider equality of types: if $A = B$ *type*, then elements of A should be elements of B.

types depend on elements, so equality of elements necessary too. not all equations can be made automatic, so a language of proofs must account for *coercions*.

equality in type theory

what equations can be made automatic? surely $\alpha/\delta/\beta$, and type theorists also know how to automate $\eta/\xi/\nu/ \dots$

equality in type theory

what equations can be made automatic? surely $\alpha/\delta/\beta$, and type theorists also know how to automate $\eta/\xi/\nu/\ldots$

$$x : A \times B \vdash M : F(x)$$

iff

$$x : A \times B \vdash M : F(\langle x.1, x.2 \rangle)$$

equality in type theory

other equations may require explicit coercion.[1] consider a family $n : \mathbf{nat} \vdash F(n)$ *type*...

[1]"equality reflection" just pushes the problem elsewhere and makes it worse! unlike many, I speak from experience.

equality in type theory

other equations may require explicit coercion.[1] consider a family $n : \textbf{nat} \vdash F(n)$ *type*...

$$n : \textbf{nat} \vdash M : F(n+1) \quad \textbf{iff???} \quad n : \textbf{nat} \vdash M : F(1+n)$$

[1]"equality reflection" just pushes the problem elsewhere and makes it worse! unlike many, I speak from experience.

equality in type theory

other equations may require explicit coercion.[1] consider a family $n : \mathbf{nat} \vdash F(n)$ *type*...

$$n : \mathbf{nat} \vdash M : F(n+1) \quad \textbf{iff} \quad n : \mathbf{nat} \vdash \mathbf{coe}_{F(-)}(P, M) : F(1+n)$$

[1] "equality reflection" just pushes the problem elsewhere and makes it worse! unlike many, I speak from experience.

equality in type theory

other equations may require explicit coercion.[1] consider a family $n : \textbf{nat} \vdash F(n)$ *type*...

$$n : \textbf{nat} \vdash M : F(n+1) \quad \textbf{iff} \quad n : \textbf{nat} \vdash \textbf{coe}_{F(-)}(P, M) : F(1+n)$$

1. is M equal to $\textbf{coe}_{F(-)}(P, M)$? yes, up to a coercion

[1] "equality reflection" just pushes the problem elsewhere and makes it worse! unlike many, I speak from experience.

equality in type theory

other equations may require explicit coercion.[1] consider a family $n : \mathbf{nat} \vdash F(n)\ \mathit{type}$...

$$n : \mathbf{nat} \vdash M : F(n+1) \quad \textbf{iff} \quad n : \mathbf{nat} \vdash \mathbf{coe}_{F(-)}(P, M) : F(1+n)$$

1. is M equal to $\mathbf{coe}_{F(-)}(P, M)$? yes, up to a coercion
2. is $\mathbf{coe}_{F(-)}(P, M)$ equal to $\mathbf{coe}_{F(-)}(Q, M)$? maybe

[1] "equality reflection" just pushes the problem elsewhere and makes it worse! unlike many, I speak from experience.

Observational Type Theory

Altenkirch and McBride [AM06]. ***Towards Observational Type Theory*****.**

Altenkirch, McBride, and Swierstra [AMS07]. "Observational Equality, Now!"

Observational Type Theory

Altenkirch and McBride [AM06]. *Towards Observational Type Theory*.

Altenkirch, McBride, and Swierstra [AMS07]. "Observational Equality, Now!"

- hierarchy of *closed*/*inductive* universes of sets, props

Observational Type Theory

Altenkirch and McBride [AM06]. *Towards Observational Type Theory*.
Altenkirch, McBride, and Swierstra [AMS07]. "Observational Equality, Now!"

- hierarchy of *closed*/*inductive* universes of sets, props
- heterogeneous equality type $\mathbf{Eq}(M : A, N : B)$ defined as *generic program*, by recursion on type codes A, B

$$\mathbf{Eq}(F_0 : A_0 \to B_0, F_1 : A_1 \to B_1) = (x_0 : A_0)(x_1 : A_1)(\tilde{x} : \mathbf{Eq}(x_0 : A_0, x_1 : A_1)) \to \mathbf{Eq}(F_0(x_0) : B_0, F_1(x_1) : B_1) \tag{funext}$$

Observational Type Theory

Altenkirch and McBride [AM06]. *Towards Observational Type Theory*.
Altenkirch, McBride, and Swierstra [AMS07]. "Observational Equality, Now!"

- hierarchy of *closed*/*inductive* universes of sets, props
- heterogeneous equality type $\mathbf{Eq}(M : A, N : B)$ defined as *generic program*, by recursion on type codes A, B

$$\mathbf{Eq}(F_0 : A_0 \to B_0, F_1 : A_1 \to B_1) = (x_0 : A_0)(x_1 : A_1)(\tilde{x} : \mathbf{Eq}(x_0 : A_0, x_1 : A_1)) \to \mathbf{Eq}(F_0(x_0) : B_0, F_1(x_1) : B_1) \qquad \text{(funext)}$$

- judgmental UIP (proof irrelevance): always have $P = Q : \mathbf{Eq}(M_0 : A_0, M_1 : A_1)$

Observational Type Theory

Altenkirch and McBride [AM06]. *Towards Observational Type Theory*.
Altenkirch, McBride, and Swierstra [AMS07]. "Observational Equality, Now!"

- hierarchy of *closed/inductive* universes of sets, props
- heterogeneous equality type $\mathbf{Eq}(M : A, N : B)$ defined as *generic program*, by recursion on type codes A, B

$$\begin{aligned} &\mathbf{Eq}(F_0 : A_0 \to B_0, F_1 : A_1 \to B_1) = \\ &\quad (x_0 : A_0)(x_1 : A_1)(\tilde{x} : \mathbf{Eq}(x_0 : A_0, x_1 : A_1)) \\ &\quad \to \mathbf{Eq}(F_0(x_0) : B_0, F_1(x_1) : B_1) \end{aligned} \qquad \text{(funext)}$$

- judgmental UIP (proof irrelevance): always have $P = Q : \mathbf{Eq}(M_0 : A_0, M_1 : A_1)$
- many primitives: reflexivity, respect, coercion, coherence, heterogeneous irrelevance (see Altenkirch, McBride, and Swierstra [AMS07])

cubical reconstruction: XTT

goal: find smaller set of primitives which systematically generate (something in the spirit of) **OTT**

idea: start with Cartesian cubical type theory [ABCFHL], restrict to *Bishop sets* à la Coquand [Coq17]

the **XTT** paper

Sterling, Angiuli, and Gratzer [SAG19]. “Cubical Syntax for Reflection-Free Extensional Equality”. *Formal Structures for Computation and Deduction (FSCD 2019).*

XTT: equality using the interval

rather than defining heterogeneous equality by recursion on type structure, define *dependent equality* all at once using a formal interval:

$$\frac{}{0, 1 : \mathbb{I}}$$

EQ FORMATION

$$\frac{i : \mathbb{I} \vdash A : \textbf{Type} \qquad M : A[0] \qquad N : A[1]}{\textbf{Eq}_{i.A[i]}(M, N) : \textbf{Type}}$$

EQ INTRODUCTION

$$\frac{i : \mathbb{I} \vdash M[i] : A[i] \qquad M[0] = N_0 : A[0] \qquad M[1] = N_1 : A[1]}{\lambda i.M[i] : \textbf{Eq}_{i.A[i]}(N_0, N_1)}$$

EQ ELIMINATION

$$\frac{M : \textbf{Eq}_{i.A[i]}(N_0, N_1) \qquad r : \mathbb{I}}{M(r) : A[r] \qquad M(0) = N_0 : A[0] \qquad M(1) = N_1 : A[1]}$$

(along with more β, η rules, etc.)

function extensionality in XTT

we have function extensionality by swapping quantifiers:

$$\dfrac{F_0, F_1 : A \to B \qquad Q : (x : A) \to \mathbf{Eq}_{\_.B}(F_0(x), F_1(x))}{\lambda i.\lambda x.Q(x)(i) : \mathbf{Eq}_{\_.A \to B}(F_0, F_1)}$$

generalized coercion: coercion, coherence, and more

given a cube $Q : \mathbf{Eq}_{\_.\mathsf{Type}}(A, B)$, we can *coerce* from A to B:

$$\frac{Q : \mathbf{Eq}_{\_.\mathsf{Type}}(A, B) \qquad M : A}{[i.Q(i)] \downarrow_1^0 M : B}$$

generalized coercion: coercion, coherence, and more

given a cube $Q : \mathbf{Eq}_{\_.\mathsf{Type}}(A, B)$, we can *coerce* from A to B:

$$\frac{i : \mathbb{I} \vdash C[i] : \mathbf{Type} \qquad M : C[0]}{[i.C[i]] \downarrow_1^0 M : C[1]}$$

generalized coercion: coercion, coherence, and more

given a cube $Q : \mathbf{Eq}_{\_.\mathbf{Type}}(A, B)$, we can *coerce* from A to B:

$$\frac{i : \mathbb{I} \vdash C[i] : \mathbf{Type} \qquad M : C[0]}{[i.C[i]] \downarrow_1^0 M : C[1]}$$

but how is M related to $[i.C[i]] \downarrow_1^0 M$?

generalized coercion: coercion, coherence, and more

given a cube $Q : \textbf{Eq}_{\_.\textsf{Type}}(A, B)$, we can *coerce* from A to B:

$$\frac{r, r' : \mathbb{I} \qquad i : \mathbb{I} \vdash C[i] : \textbf{Type} \qquad M : C[r]}{[i.C[i]] \downarrow^{r}_{r'} M : C[r']}$$

but how is M related to $[i.C[i]] \downarrow^{0}_{1} M$? by another equation.

generalized coercion: coercion, coherence, and more

given a cube $Q : \mathbf{Eq}_{\_.\mathsf{Type}}(A, B)$, we can *coerce* from A to B:

$$\frac{r, r' : \mathbb{I} \qquad i : \mathbb{I} \vdash C[i] : \mathsf{Type} \qquad M : C[r]}{[i.C[i]] \downarrow_{r'}^{r} M : C[r']}$$

but how is M related to $[i.C[i]] \downarrow_{1}^{0} M$? by another equation.

$$\frac{i : \mathbb{I} \vdash C[i] : \mathsf{Type} \qquad M : C[0]}{\lambda j.[i.C[i]] \downarrow_{j}^{0} M : \mathbf{Eq}_{i.C[i]}(M, [i.C[i]] \downarrow_{1}^{0} M)}$$

generalized coercion: attaching faces

allow either zero or two faces to be attached:

$$\frac{r, r', s : \mathbb{I} \qquad i : \mathbb{I} \vdash A[i] : \textbf{Type} \qquad M : A[r] \qquad \begin{array}{l} s = 0, j : \mathbb{I} \vdash N_0 : A[r'] \\ s = 1, j : \mathbb{I} \vdash N_1 : A[r'] \end{array}}{[i.A[i]] \downarrow^{r}_{r'} M \; [s = 0 \to j.N_0 \mid s = 1 \to j.N_1] : A[r']}$$

generalized coercion: attaching faces

allow either zero or two faces to be attached:

$$\frac{r, r', s : \mathbb{I} \qquad i : \mathbb{I} \vdash A[i] : \textbf{Type} \qquad M : A[r] \qquad \begin{array}{l} s = 0, j : \mathbb{I} \vdash N_0 : A[r'] \\ s = 1, j : \mathbb{I} \vdash N_1 : A[r'] \end{array}}{[i.A[i]] \downarrow^{r}_{r'} M \; [s = 0 \to j.N_0 \mid s = 1 \to j.N_1] : A[r']}$$

implements symmetry, transitivity, coercion in $\textbf{Eq}_{i.A}(M, N)$

$\Longrightarrow$ generalizes several primitives of **OTT** simultaneously

like **OTT**, deciding equality of coercions requires *inductive-recursive* universe

judgmental UIP via *boundary separation*

in **OTT**, we always have $Q_0 = Q_1 : \mathbf{Eq}(M : A, N : B)$; we achieve this modularly using a *boundary separation*[2] rule:

$$\frac{r : \mathbb{I} \qquad r = 0 \vdash M = N : A \qquad r = 1 \vdash M = N : A}{M = N : A}$$

(does not mention equality type!!)

given $Q_0, Q_1 : \mathbf{Eq}_{i.A}(M, N)$, we have $Q_0 = Q_1 : \mathbf{Eq}_{i.A}(M, N)$ by the β, η, ξ rules of the equality type, together with boundary separation.

[2](it is a presheaf separation condition for a certain coverage on the category of contexts)

5 second coffee break

subjective metatheory: counting grains of sand

we used to study the metatheory of *presentations* of type theories, not of type theories.

subjective metatheory: counting grains of sand

we used to study the metatheory of *presentations* of type theories, not of type theories.

1. "raw" terms, "raw" substitution, insufficient annotations (*a priori* no determinate notion of model, nor interpretation)
2. ???
3. interpretation into models???

subjective metatheory: counting grains of sand

we used to study the metatheory of *presentations* of type theories, not of type theories.

1. "raw" terms, "raw" substitution, insufficient annotations (*a priori* no determinate notion of model, nor interpretation)
2. prove normalization for raw syntax (but without using model theory!)
3. interpretation into models???

subjective metatheory: counting grains of sand

we used to study the metatheory of *presentations* of type theories, not of type theories.

1. "raw" terms, "raw" substitution, insufficient annotations (*a priori* no determinate notion of model, nor interpretation)
2. prove normalization for raw syntax (but without using model theory!)
   2.1 operational semantics
   2.2 PER "model" of type theory
   2.3 logical relation between syntax and PER "model"

   (~ 200 pages of work)
3. interpretation into models???

subjective metatheory: counting grains of sand

we used to study the metatheory of *presentations* of type theories, not of type theories.

1. "raw" terms, "raw" substitution, insufficient annotations (*a priori* no determinate notion of model, nor interpretation)
2. prove normalization for raw syntax (but without using model theory!)
   - 2.1 operational semantics
   - 2.2 PER "model" of type theory
   - 2.3 logical relation between syntax and PER "model"

   (~ 200 pages of work)
3. sound & complete interpretation (~ 100 more pages of work)

subjective metatheory: counting grains of sand

we used to study the metatheory of *presentations* of type theories, not of type theories.

1. "raw" terms, "raw" substitution, insufficient annotations (*a priori* no determinate notion of model, nor interpretation)
2. prove normalization for raw syntax (but without using model theory!)
   2.1 operational semantics
   2.2 PER "model" of type theory
   2.3 logical relation between syntax and PER "model"

   (~ 200 pages of work)
3. sound & complete interpretation (~ 100 more pages of work)

actually this is totally intractable to do more than once! let's bootstrap it a different way.

objective metatheory and categorical gluing

a new (old) syntax-invariant approach to metatheory

[3] See also Coquand, Huber, and Sattler [CHS19], Kaposi, Huber, and Sattler [KHS19], and Shulman [Shu15].

objective metatheory and categorical gluing

a new (old) syntax-invariant approach to metatheory

1. type theory is essentially algebraic (insist on it!) [Car86; ACD08; Awo18; Uem19]; presentations considered up to isomorphism

---

[3] See also Coquand, Huber, and Sattler [CHS19], Kaposi, Huber, and Sattler [KHS19], and Shulman [Shu15].

objective metatheory and categorical gluing

a new (old) syntax-invariant approach to metatheory

1. type theory is essentially algebraic (insist on it!) [Car86; ACD08; Awo18; Uem19]; presentations considered up to isomorphism
2. each type theory $\mathbb{T}$ *automatically* induces a category of algebras with initial object (soundness and completeness); initial algebra is covered by *fully-annotated* De Bruijn syntax (but this doesn't matter)

---

[3] See also Coquand, Huber, and Sattler [CHS19], Kaposi, Huber, and Sattler [KHS19], and Shulman [Shu15].

objective metatheory and categorical gluing

a new (old) syntax-invariant approach to metatheory

1. type theory is essentially algebraic (insist on it!) [Car86; ACD08; Awo18; Uem19]; presentations considered up to isomorphism
2. each type theory $\mathbb{T}$ *automatically* induces a category of algebras with initial object (soundness and completeness); initial algebra is covered by *fully-annotated* De Bruijn syntax (but this doesn't matter)
3. easily prove canonicity, normalization, decidability of type checking for initial $\mathbb{T}$-algebra using categorical gluing/logical families [Coq18][3]

[3] See also Coquand, Huber, and Sattler [CHS19], Kaposi, Huber, and Sattler [KHS19], and Shulman [Shu15].

objective metatheory and categorical gluing

a new (old) syntax-invariant approach to metatheory

1. type theory is essentially algebraic (insist on it!) [Car86; ACD08; Awo18; Uem19]; presentations considered up to isomorphism
2. each type theory $\mathbb{T}$ *automatically* induces a category of algebras with initial object (soundness and completeness); initial algebra is covered by *fully-annotated* De Bruijn syntax (but this doesn't matter)
3. easily prove canonicity, normalization, decidability of type checking for initial $\mathbb{T}$-algebra using categorical gluing/logical families [Coq18][3]
4. relate "informal" & unannotated syntax to initial $\mathbb{T}$-algebra by elaboration (using the above)

[3] See also Coquand, Huber, and Sattler [CHS19], Kaposi, Huber, and Sattler [KHS19], and Shulman [Shu15].

objective metatheory and categorical gluing

a new (old) syntax-invariant approach to metatheory

1. type theory is essentially algebraic (insist on it!) [Car86; ACD08; Awo18; Uem19]; presentations considered up to isomorphism
2. each type theory $\mathbb{T}$ *automatically* induces a category of algebras with initial object (soundness and completeness); initial algebra is covered by *fully-annotated* De Bruijn syntax (but this doesn't matter)
3. easily prove canonicity, normalization, decidability of type checking for initial $\mathbb{T}$-algebra using categorical gluing/logical families [Coq18][3]
4. relate "informal" & unannotated syntax to initial $\mathbb{T}$-algebra by elaboration (using the above)

the language of category theory makes each of the preceding steps "easy", and independent of syntax / representation details. no raw terms, no PERs.

[3]See also Coquand, Huber, and Sattler [CHS19], Kaposi, Huber, and Sattler [KHS19], and Shulman [Shu15].

cubical gluing: canonicity for XTT

to warm up, we proved canonicity for **XTT** using a cubical gluing technique (independently proposed by Awodey).

cubical gluing: canonicity for XTT

to warm up, we proved canonicity for **XTT** using a cubical gluing technique (independently proposed by Awodey).

Theorem (Canonicity)

*In the initial **XTT**-algebra, if $M \in \mathbf{El}(\diamond, \mathbf{bool})$ then either $M = \mathbf{tt}$ or $M = \mathbf{ff}$.*

use "cubical version" of global sections functor (cubical nerve). first we need to understand **XTT**'s structure.

$$\Psi \mid \Gamma \vdash M : A$$

$$\Psi \mid \Gamma \vdash M : A$$

Ψ *cube*$_+$

$\square_+ :$ **Cat**

$$\Psi \mid \Gamma \vdash M : A$$

Ψ *cube*$_+$ $\qquad$ $\Psi \mid \Gamma$ *ctx*

$\square_+ : \mathbf{Cat}$ $\qquad$ $\mathbb{C} \xrightarrow{\mathfrak{u}} \square_+$

$$\Psi \mid \Gamma \vdash M : A$$

Ψ *cube*$_+$	$\Psi \mid \Gamma$ *ctx*	$\Psi \mid \Gamma \vdash A$ *type*
$\square_+ : \mathbf{Cat}$	$\mathbb{C} \xrightarrow{\mathfrak{u}} \square_+$	$\mathbb{C}^{\mathsf{op}} \xrightarrow{\mathsf{Ty}} \mathbf{Set}$

$$\boxed{\Psi \mid \Gamma \vdash M : A}$$

$$\mathbf{El} \longrightarrow \mathbf{Ty} : \mathbf{Pr}(\mathbb{C})$$

$\Psi\ \mathit{cube}_+$	$\Psi \mid \Gamma\ \mathit{ctx}$	$\Psi \mid \Gamma \vdash A\ \mathit{type}$
$\square_+ : \mathbf{Cat}$	$\mathbb{C} \xrightarrow{\mathfrak{u}} \square_+$	$\mathbb{C}^{\mathsf{op}} \xrightarrow{\mathbf{Ty}} \mathbf{Set}$

computability families and the *cubical nerve*

idea: consider empty Γ, arbitrary Ψ; "cubical" version of closed terms

$$\boxed{\Psi \mid \diamond \vdash M : A}$$

[4]proposed by Awodey in 2015; analogous to Fiore's *relative hom functor* in NbE (2002)

computability families and the *cubical nerve*

idea: consider empty Γ, arbitrary Ψ; "cubical" version of closed terms

$$\Psi \mid \diamond \vdash M : A$$

define for each $\Psi \mid \diamond \vdash A$ *type* a family of "computability proofs" over each $M : A$; must live in **Pr**$(\square_+)$.

$$\begin{array}{ccc} \square_+ & \xrightarrow{\mathfrak{i}} & \mathbb{C} \\ \Psi & \longmapsto & \Psi \mid \diamond \end{array}$$

[4] proposed by Awodey in 2015; analogous to Fiore's *relative hom functor* in NbE (2002)

computability families and the *cubical nerve*

idea: consider empty Γ, arbitrary Ψ; "cubical" version of closed terms

$$\boxed{\Psi \mid \diamond \vdash M : A}$$

define for each $\Psi \mid \diamond \vdash A$ *type* a family of "computability proofs" over each $M : A$; must live in **Pr**$(\square_+)$.

$$\begin{array}{ccc} \square_+ & \xrightarrow{\mathfrak{i}} & \mathbb{C} \\ \Psi & \longmapsto & \Psi \mid \diamond \end{array}$$

"cubical global sections functor" is a nerve $\mathbb{C} \xrightarrow{\mathbf{N}} \mathbf{Pr}(\square_+)$,[4] restricting the Yoneda embedding to purely cubical contexts:

$$\mathbf{N}(\Gamma) = \mathbb{C}[\mathfrak{i}(-), \Gamma]$$

[4]proposed by Awodey in 2015; analogous to Fiore's *relative hom functor* in NbE (2002)

a flavor of cubical gluing

- a glued context $\widetilde{\Gamma} : \widetilde{\mathbb{C}}$ is a context $\Gamma : \mathbb{C}$ together with a computability family $\Gamma^\bullet : \mathbf{N}(\Gamma) \to \mathbf{U}$ internal to $\mathbf{Pr}(\square_+)$.

[5] more complicated, because **XTT** needs inductive-recursive universe; just for intuition!

a flavor of cubical gluing

- a glued context $\widetilde{\Gamma} : \widetilde{\mathbb{C}}$ is a context $\Gamma : \mathbb{C}$ together with a computability family $\Gamma^\bullet : \mathbf{N}(\Gamma) \to \mathbf{U}$ internal to $\mathbf{Pr}(\Box_+)$.
- a glued substitution $\widetilde{\Delta} \xrightarrow{\widetilde{\gamma}} \widetilde{\Gamma}$ is a substitution $\Delta \xrightarrow{\gamma} \Gamma$ together with a realizer $\gamma^\bullet : \prod_{\delta : \mathbf{N}(\Delta)} \prod_{\delta^\bullet : \Delta^\bullet \delta} \Gamma^\bullet(\gamma\delta)$.

[5] more complicated, because **XTT** needs inductive-recursive universe; just for intuition!

a flavor of cubical gluing

- a glued context $\widetilde{\Gamma} : \widetilde{\mathbb{C}}$ is a context $\Gamma : \mathbb{C}$ together with a computability family $\Gamma^\bullet : \mathbf{N}(\Gamma) \to \mathbf{U}$ internal to $\mathbf{Pr}(\square_+)$.
- a glued substitution $\widetilde{\Delta} \xrightarrow{\widetilde{\gamma}} \widetilde{\Gamma}$ is a substitution $\Delta \xrightarrow{\gamma} \Gamma$ together with a realizer $\gamma^\bullet : \prod_{\delta:\mathbf{N}(\Delta)} \prod_{\delta^\bullet:\Delta^\bullet\delta} \Gamma^\bullet(\gamma\delta)$.
- a glued type $\widetilde{A} \in \mathbf{Ty}(\widetilde{\Gamma})$ has a type $A \in \mathbf{Ty}(\Gamma)$ together with a computability family $A^\bullet : \prod_{\gamma:\mathbf{N}(\Gamma)} \prod_{\gamma^\bullet:\Gamma^\bullet} \mathbf{N}(A\gamma) \to \mathbf{U}$.[5]

[5] more complicated, because **XTT** needs inductive-recursive universe; just for intuition!

a flavor of cubical gluing

- a glued context $\widetilde{\Gamma} : \widetilde{\mathbb{C}}$ is a context $\Gamma : \mathbb{C}$ together with a computability family $\Gamma^\bullet : \mathbf{N}(\Gamma) \to \mathbf{U}$ internal to $\mathbf{Pr}(\square_+)$.
- a glued substitution $\widetilde{\Delta} \xrightarrow{\widetilde{\gamma}} \widetilde{\Gamma}$ is a substitution $\Delta \xrightarrow{\gamma} \Gamma$ together with a realizer $\gamma^\bullet : \prod_{\delta:\mathbf{N}(\Delta)} \prod_{\delta^\bullet:\Delta^\bullet\delta} \Gamma^\bullet(\gamma\delta)$.
- a glued type $\widetilde{A} \in \mathbf{Ty}(\widetilde{\Gamma})$ has a type $A \in \mathbf{Ty}(\Gamma)$ together with a computability family $A^\bullet : \prod_{\gamma:\mathbf{N}(\Gamma)} \prod_{\gamma^\bullet:\Gamma^\bullet} \mathbf{N}(A\gamma) \to \mathbf{U}$.[5]
- a glued element $\widetilde{M} \in \mathbf{El}(\widetilde{\Gamma}, \widetilde{A})$ is an element $M \in \mathbf{El}(\Gamma, A)$ together with a realizer $M^\bullet : \prod_{\gamma:\mathbf{N}(\Gamma)} \prod_{\gamma^\bullet:\Gamma^\bullet} A^\bullet\gamma\gamma^\bullet(M\gamma)$.

[5] more complicated, because **XTT** needs inductive-recursive universe; just for intuition!

a flavor of cubical gluing

- a glued context $\widetilde{\Gamma} : \widetilde{\mathbb{C}}$ is a context $\Gamma : \mathbb{C}$ together with a computability family $\Gamma^\bullet : \mathbf{N}(\Gamma) \to \mathbf{U}$ internal to $\mathbf{Pr}(\square_+)$.
- a glued substitution $\widetilde{\Delta} \xrightarrow{\widetilde{\gamma}} \widetilde{\Gamma}$ is a substitution $\Delta \xrightarrow{\gamma} \Gamma$ together with a realizer $\gamma^\bullet : \prod_{\delta:\mathbf{N}(\Delta)} \prod_{\delta^\bullet:\Delta^\bullet\delta} \Gamma^\bullet(\gamma\delta)$.
- a glued type $\widetilde{A} \in \mathbf{Ty}(\widetilde{\Gamma})$ has a type $A \in \mathbf{Ty}(\Gamma)$ together with a computability family $A^\bullet : \prod_{\gamma:\mathbf{N}(\Gamma)} \prod_{\gamma^\bullet:\Gamma^\bullet} \mathbf{N}(A\gamma) \to \mathbf{U}$.[5]
- a glued element $\widetilde{M} \in \mathbf{El}(\widetilde{\Gamma}, \widetilde{A})$ is an element $M \in \mathbf{El}(\Gamma, A)$ together with a realizer $M^\bullet : \prod_{\gamma:\mathbf{N}(\Gamma)} \prod_{\gamma^\bullet:\Gamma^\bullet} A^\bullet\gamma\gamma^\bullet(M\gamma)$.

intuition: "realizers" are semantic whnfs, but *intrinsic*. what remains is the pure essence of operational-style techniques [Hub18; AFH17].

[5]more complicated, because **XTT** needs inductive-recursive universe; just for intuition!

proving canonicity

Theorem (Canonicity)

*In the initial **XTT**-algebra, if* $M \in \mathbf{El}(\diamond, \mathbf{bool})$ *then either* $M = \mathbf{tt}$ *or* $M = \mathbf{ff}$.

proving canonicity

Theorem (Canonicity)

*In the initial **XTT**-algebra, if* $M \in \mathbf{El}(\diamond, \mathbf{bool})$ *then either* $M = \mathbf{tt}$ *or* $M = \mathbf{ff}$.

we choose a computability family for **bool** which forces this to be true!

$$\widetilde{\mathbf{bool}} \in \mathbf{Ty}(\tilde{\diamond})$$
$$\widetilde{\mathbf{bool}} = \left(\mathbf{bool}, \mathbf{?} : \textstyle\prod_{\epsilon:\mathbf{N}(\diamond)} \prod_{\epsilon^\bullet : \diamond^\bullet \epsilon} \mathbf{U}\right)$$

proving canonicity

Theorem (Canonicity)

*In the initial **XTT**-algebra, if $M \in \mathbf{El}(\diamond, \mathbf{bool})$ then either $M = \mathbf{tt}$ or $M = \mathbf{ff}$.*

we choose a computability family for **bool** which forces this to be true!

$$\widetilde{\mathbf{bool}} \in \mathbf{Ty}(\tilde{\diamond})$$
$$\widetilde{\mathbf{bool}} = (\mathbf{bool}, \lambda \epsilon \epsilon^{\bullet} M.(M = \mathbf{tt}) + (M = \mathbf{ff}))$$

idea: the "realizer" of any closed boolean reveals whether it is tt or ff. abstract operational semantics!

proving canonicity

Theorem (Canonicity)

*In the initial **XTT**-algebra, if* $M \in \mathbf{El}(\diamond, \mathbf{bool})$ *then either* $M = \mathbf{tt}$ *or* $M = \mathbf{ff}$.

we choose a computability family for **bool** which forces this to be true!

$$\widetilde{\mathbf{bool}} \in \mathbf{Ty}(\tilde{\diamond})$$
$$\widetilde{\mathbf{bool}} = (\mathbf{bool}, \lambda\epsilon\epsilon^{\bullet}M.(M = \mathbf{tt}) + (M = \mathbf{ff}))$$

$$\widetilde{\mathbf{tt}}, \widetilde{\mathbf{ff}} \in \mathbf{El}(\tilde{\diamond}, \widetilde{\mathbf{bool}})$$
$$\widetilde{\mathbf{tt}} = (\mathbf{tt}, \lambda\epsilon\epsilon^{\bullet}.\mathbf{inl}(\mathbf{refl}_{\mathbf{tt}}))$$
$$\widetilde{\mathbf{ff}} = (\mathbf{ff}, \lambda\epsilon\epsilon^{\bullet}.\mathbf{inr}(\mathbf{refl}_{\mathbf{ff}}))$$

idea: the "realizer" of any closed boolean reveals whether it is tt or ff. abstract operational semantics!

canonicity, delivered

Theorem (Canonicity)

*In the initial **XTT**-algebra, if $M \in \mathbf{El}(\diamond, \mathbf{bool})$ then either $M = \mathbf{tt}$ or $M = \mathbf{ff}$.*

Proof.

If $M \in \mathbf{El}(\diamond, \mathbf{bool})$ in the initial **XTT**-algebra $\mathbb{C}$, then by the universal property of $\mathbb{C}$, there exists $\widetilde{N} = (N, N^\bullet) \in \mathbf{El}(\tilde{\diamond}, \widetilde{\mathbf{bool}})$ such that $N = M$ and $N^\bullet \in \mathbf{bool}^\bullet N$. Proceed by case:

1. if $N^\bullet = \mathbf{inl}(\mathbf{refl}_{\mathbf{tt}})$, then $M = N = \mathbf{tt}$
2. if $N^\bullet = \mathbf{inr}(\mathbf{refl}_{\mathbf{ff}})$, then $M = N = \mathbf{ff}$ $\square$

outlook

we contributed a (Cartesian) cubical reconstruction of OTT, and took a first step toward objective metatheory (gluing) for cubical type theory. what's next?

- can we overcome inductive-recursive universes?
- can we add propositions with function comprehension (AUC)?
- can we add effective quotients?

judgmental boundary separation most likely too strict for any of the above; **XTT** could be extended to a language for quasitoposes (not toposes). programming applications hoped for!

References I

[ABCFHL] Carlo Angiuli, Guillaume Brunerie, Thierry Coquand, Kuen-Bang Hou (Favonia), Robert Harper, and Daniel R. Licata. “Syntax and Models of Cartesian Cubical Type Theory”. Preprint. Feb. 2019. URL: https://github.com/dlicata335/cart-cube (cit. on p. 19).

[ACD08] Andreas Abel, Thierry Coquand, and Peter Dybjer. “On the Algebraic Foundation of Proof Assistants for Intuitionistic Type Theory”. In: *Functional and Logic Programming*. Ed. by Jacques Garrigue and Manuel V. Hermenegildo. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 3–13. ISBN: 978-3-540-78969-7 (cit. on pp. 37–42).

[AFH17] Carlo Angiuli, Kuen-Bang Hou (Favonia), and Robert Harper. *Computational Higher Type theory III: Univalent Universes and Exact Equality*. 2017. arXiv: 1712.01800 (cit. on pp. 53–57).

References II

[AK16a] Thorsten Altenkirch and Ambrus Kaposi. “Normalisation by Evaluation for Dependent Types”. In: *1st International Conference on Formal Structures for Computation and Deduction (FSCD 2016)*. Ed. by Delia Kesner and Brigitte Pientka. Vol. 52. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2016, 6:1–6:16. ISBN: 978-3-95977-010-1. DOI: 10.4230/LIPIcs.FSCD.2016.6.

[AK16b] Thorsten Altenkirch and Ambrus Kaposi. “Type Theory in Type Theory Using Quotient Inductive Types”. In: *Proceedings of the 43rd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*. POPL ’16. St. Petersburg, FL, USA: ACM, 2016, pp. 18–29. ISBN: 978-1-4503-3549-2. DOI: 10.1145/2837614.2837638.

References III

[AM06] Thorsten Altenkirch and Conor McBride. *Towards Observational Type Theory*. 2006. URL: www.strictlypositive.org/ott.pdf (cit. on pp. 14–18).

[AMB13] Guillaume Allais, Conor McBride, and Pierre Boutillier. “New Equations for Neutral Terms: A Sound and Complete Decision Procedure, Formalized”. In: *Proceedings of the 2013 ACM SIGPLAN Workshop on Dependently-typed Programming*. DTP ’13. Boston, Massachusetts, USA: ACM, 2013, pp. 13–24. ISBN: 978-1-4503-2384-0. DOI: 10.1145/2502409.2502411.

[AMS07] Thorsten Altenkirch, Conor McBride, and Wouter Swierstra. “Observational Equality, Now!” In: *Proceedings of the 2007 Workshop on Programming Languages Meets Program Verification*. PLPV ’07. Freiburg, Germany: ACM, 2007, pp. 57–68. ISBN: 978-1-59593-677-6 (cit. on pp. 14–18).

References IV

[Awo18] Steve Awodey. "Natural models of homotopy type theory". In: *Mathematical Structures in Computer Science* 28.2 (2018), pp. 241–286. DOI: 10.1017/S0960129516000268 (cit. on pp. 37–42).

[BD08] Alexandre Buisse and Peter Dybjer. "Towards formalizing categorical models of type theory in type theory". In: *Electronic Notes in Theoretical Computer Science* 196 (2008), pp. 137–151.

[Car86] John Cartmell. "Generalised algebraic theories and contextual categories". In: *Annals of Pure and Applied Logic* 32 (1986), pp. 209–243. ISSN: 0168-0072 (cit. on pp. 37–42).

[CCD17] Simon Castellan, Pierre Clairambault, and Peter Dybjer. "Undecidability of Equality in the Free Locally Cartesian Closed Category (Extended version)". In: *Logical Methods in Computer Science* 13.4 (2017).

References V

[CCHM17] Cyril Cohen, Thierry Coquand, Simon Huber, and Anders Mörtberg. "Cubical Type Theory: a constructive interpretation of the univalence axiom". In: *IfCoLog Journal of Logics and their Applications* 4.10 (Nov. 2017), pp. 3127–3169. URL: `http://www.collegepublications.co.uk/journals/ifcolog/?00019`.

[CFM18] James Chapman, Fredrik Nordvall Forsberg, and Conor McBride. "The Box of Delights (Cubical Observational Type Theory)". Unpublished note. Jan. 2018. URL: `https://github.com/msp-strath/platypus/blob/master/January18/doc/CubicalOTT/CubicalOTT.pdf`.

[CHS19] Thierry Coquand, Simon Huber, and Christian Sattler. "Homotopy canonicity for cubical type theory". In: *Proceedings of the 4th International Conference on Formal Structures for Computation and Deduction (FSCD 2019)*. Ed. by Herman Geuvers. Vol. 131. 2019 (cit. on pp. 37–42).

References VI

[Coq17] Thierry Coquand. *Universe of Bishop sets.* Feb. 2017. URL: http://www.cse.chalmers.se/~coquand/bishop.pdf (cit. on p. 19).

[Coq18] Thierry Coquand. *Canonicity and normalization for Dependent Type Theory.* Oct. 2018. arXiv: 1810.09367 (cit. on pp. 37–42).

[Fio02] Marcelo Fiore. "Semantic Analysis of Normalisation by Evaluation for Typed Lambda Calculus". In: *Proceedings of the 4th ACM SIGPLAN International Conference on Principles and Practice of Declarative Programming.* PPDP '02. Pittsburgh, PA, USA: ACM, 2002, pp. 26–37. ISBN: 1-58113-528-9. DOI: 10.1145/571157.571161 (cit. on pp. 50–52).

[Hub18] Simon Huber. "Canonicity for Cubical Type Theory". In: *Journal of Automated Reasoning* (June 13, 2018). ISSN: 1573-0670. DOI: 10.1007/s10817-018-9469-1 (cit. on pp. 53–57).

References VII

[JT93] Achim Jung and Jerzy Tiuryn. “A new characterization of lambda definability”. In: *Typed Lambda Calculi and Applications*. Ed. by Marc Bezem and Jan Friso Groote. Berlin, Heidelberg: Springer Berlin Heidelberg, 1993, pp. 245–257. ISBN: 978-3-540-47586-6.

[KHS19] Ambrus Kaposi, Simon Huber, and Christian Sattler. “Gluing for type theory”. In: *Proceedings of the 4th International Conference on Formal Structures for Computation and Deduction (FSCD 2019)*. Ed. by Herman Geuvers. Vol. 131. 2019 (cit. on pp. 37–42).

[KKA19] Ambrus Kaposi, András Kovács, and Thorsten Altenkirch. “Constructing Quotient Inductive-inductive Types”. In: *Proc. ACM Program. Lang.* 3.POPL (Jan. 2019), 2:1–2:24. ISSN: 2475-1421. DOI: 10.1145/3290315.

References VIII

[ML75a] Per Martin-Löf. “About Models for Intuitionistic Type Theories and the Notion of Definitional Equality”. In: *Proceedings of the Third Scandinavian Logic Symposium*. Ed. by Stig Kanger. Vol. 82. Studies in Logic and the Foundations of Mathematics. Elsevier, 1975, pp. 81–109.

[ML75b] Per Martin-Löf. “An Intuitionistic Theory of Types: Predicative Part”. In: *Logic Colloquium ’73*. Ed. by H. E. Rose and J. C. Shepherdson. Vol. 80. Studies in Logic and the Foundations of Mathematics. Elsevier, 1975, pp. 73–118. DOI: 10.1016/S0049-237X(08)71945-1.

[MS93] John C. Mitchell and Andre Scedrov. “Notes on sconing and relators”. In: *Computer Science Logic*. Ed. by E. Börger, G. Jäger, H. Kleine Büning, S. Martini, and M. M. Richter. Berlin, Heidelberg: Springer Berlin Heidelberg, 1993, pp. 352–378. ISBN: 978-3-540-47890-4.

References IX

[SAG19] Jonathan Sterling, Carlo Angiuli, and Daniel Gratzer. “Cubical Syntax for Reflection-Free Extensional Equality”. In: *Proceedings of the 4th International Conference on Formal Structures for Computation and Deduction (FSCD 2019)*. Ed. by Herman Geuvers. Vol. 131. 2019. DOI: 10.4230/LIPIcs.FSCD.2019.32. arXiv: 1904.08562 (cit. on p. 19).

[Shu06] Michael Shulman. *Scones, Logical Relations, and Parametricity*. Blog. 2006. URL: https://golem.ph.utexas.edu/category/2013/04/scones_logical_relations_and_p.html.

[Shu15] Michael Shulman. “Univalence for inverse diagrams and homotopy canonicity”. In: *Mathematical Structures in Computer Science* 25.5 (2015), pp. 1203–1277. DOI: 10.1017/S0960129514000565 (cit. on pp. 37–42).

[SS18] Jonathan Sterling and Bas Spitters. *Normalization by gluing for free λ-theories*. Sept. 2018. arXiv: 1809.08646 [cs.LO].

References X

[Ste18] Jonathan Sterling. *Algebraic Type Theory and Universe Hierarchies*. Dec. 2018. arXiv: `1902.08848 [math.LO]`.

[Str91] Thomas Streicher. *Semantics of Type Theory: Correctness, Completeness, and Independence Results*. Cambridge, MA, USA: Birkhauser Boston Inc., 1991. ISBN: 0-8176-3594-7.

[Str94] Thomas Streicher. *Investigations Into Intensional Type Theory*. Habilitationsschrift, Universität München. 1994.

[Str98] Thomas Streicher. "Categorical intuitions underlying semantic normalisation proofs". In: *Preliminary Proceedings of the APPSEM Workshop on Normalisation by Evaluation*. Ed. by O. Danvy and P. Dybjer. Department of Computer Science, Aarhus University, 1998.

[Uem19] Taichi Uemura. *A General Framework for the Semantics of Type Theory*. 2019. arXiv: `1904.04097` (cit. on pp. 37–42).

References XI

[Voe16] Vladimir Voevodsky. *Mathematical theory of type theories and the initiality conjecture.* Research proposal to the Templeton Foundation for 2016-2019, project description. Apr. 2016. URL: http://www.math.ias.edu/Voevodsky/other/Voevodsky%20Templeton%20proposal.pdf.