On Computational Higher-Dimensional Type Theory

CIS 670 Project

Antoine Voizard and Pritam Choudhury

Department of Computer and Information Science

University of Pennsylvania

December 8, 2016

Motivation

Motivation

- The search for foundations of mathematics

Motivation

- The search for foundations of mathematics
- Type theory

Motivation

- The search for foundations of mathematics
- Type theory
- Austere notion of equality

Motivation

- The search for foundations of mathematics
- Type theory
- Austere notion of equality
- Machine-checked proofs

Motivation

- The search for foundations of mathematics
- Type theory
- Austere notion of equality
- Machine-checked proofs
- Connection between homotopy theory and type theory

Motivation

- The search for foundations of mathematics
- Type theory
- Austere notion of equality
- Machine-checked proofs
- Connection between homotopy theory and type theory
- Univalent foundations

Motivation

- The search for foundations of mathematics
- Type theory
- Austere notion of equality
- Machine-checked proofs
- Connection between homotopy theory and type theory
- Univalent foundations
- Computational Higher-Dimensional Type theory

The Homotopy connection

The Homotopy connection

The Homotopy connection

‘Portrait of Lotte’ by Frans Hofmeester

The Homotopy connection

- Types as spaces
- Equality as paths

The Homotopy connection

- Types as spaces
- Equality as paths
- Paths between paths?

The Homotopy connection

- Types as spaces
- Equality as paths
- Paths between paths?
- Paths between paths between paths?

Higher-Dimensional type theory

- Types have internal structure (Higher inductive types)

Higher-Dimensional type theory

- Types have internal structure (Higher inductive types)
- Type-theoretic operations respect the internal structures

From Martin-Löf type theory to Homotopy type theory

Judgements

- ϕ is a proposition.
- ϕ is true.
- ϕ_1 is true, ϕ_2 is true, ... , ϕ_n is true $\vdash \phi$ is a proposition.
- ϕ_1 is true, ϕ_2 is true, ... , ϕ_n is true $\vdash \phi$ is true.

Meaning of hypothetical judgements

- $$\frac{}{\phi_1 \text{ is true}, \phi_2 \text{ is true}, \ldots, \phi_i \text{ is true}, \ldots, \phi_n \text{ is true} \vdash \phi_i \text{ is true}}$$

- $$\frac{\Gamma \vdash \phi_1 \text{ is true} \qquad \Gamma, \phi_1 \text{ is true} \vdash \phi_2 \text{ is true}}{\Gamma \vdash \phi_2 \text{ is true}}$$

- $$\frac{\Gamma \vdash \phi_2 \text{ is true} \qquad \Gamma \vdash \phi_1 \text{ is a proposition}}{\Gamma, \phi_1 \text{ is true} \vdash \phi_2 \text{ is true}}$$

From Martin-Löf type theory to Homotopy type theory

Building complex propositions - True

- $\top$ is prop.
- $\Gamma \vdash \top$ is true

Building complex propositions - And

- $$\frac{\Gamma \vdash \phi_1 \text{ is prop} \quad \Gamma \vdash \phi_2 \text{ is prop}}{\Gamma \vdash \phi_1 \wedge \phi_2 \text{ is prop}}$$
- $$\frac{\Gamma \vdash \phi_1 \text{ is true} \quad \Gamma \vdash \phi_2 \text{ is true}}{\Gamma \vdash \phi_1 \wedge \phi_2 \text{ is true}}$$
- $$\frac{\Gamma \vdash \phi_1 \wedge \phi_2 \text{ is true}}{\Gamma \vdash \phi_1 \text{ is true}}$$
- $$\frac{\Gamma \vdash \phi_1 \wedge \phi_2 \text{ is true}}{\Gamma \vdash \phi_2 \text{ is true}}$$

Building complex propositions - Implication

- $$\frac{\Gamma \vdash \phi_1 \text{ is prop} \quad \Gamma, \phi_1 \text{ is true} \vdash \phi_2 \text{ is prop}}{\Gamma \vdash \phi_1 \implies \phi_2 \text{ is prop}}$$

- $$\frac{\Gamma, \phi_1 \text{ is true} \vdash \phi_2 \text{ is true}}{\Gamma \vdash \phi_1 \implies \phi_2 \text{ is true}}$$

- $$\frac{\Gamma \vdash \phi_1 \implies \phi_2 \text{ is true} \quad \Gamma \vdash \phi_1 \text{ is true}}{\Gamma \vdash \phi_2 \text{ is true}}$$

From Martin-Löf type theory to Homotopy type theory

Building complex propositions - Or

- $$\frac{\Gamma \vdash \phi_1 \text{ is prop} \quad \Gamma \vdash \phi_2 \text{ is prop}}{\Gamma \vdash \phi_1 \vee \phi_2 \text{ is prop}}$$

- $$\frac{\Gamma \vdash \phi_1 \text{ is true} \quad \Gamma \vdash \phi_2 \text{ is prop}}{\Gamma \vdash \phi_1 \vee \phi_2 \text{ is true}}$$

- $$\frac{\Gamma \vdash \phi_2 \text{ is true} \quad \Gamma \vdash \phi_1 \text{ is prop}}{\Gamma \vdash \phi_1 \vee \phi_2 \text{ is true}}$$

- $$\frac{\Gamma \vdash \phi_1 \vee \phi_2 \text{ true} \quad \Gamma, \phi_1 \text{ true} \vdash \phi \text{ true} \quad \Gamma, \phi_2 \text{ true} \vdash \phi \text{ true}}{\Gamma \vdash \phi \text{ true}}$$

Building complex propositions - False

- $\bot$ is prop.
- $$\frac{\Gamma \vdash \bot \text{ is true} \quad \Gamma, \bot \text{ is true} \vdash \phi \text{ is prop}}{\Gamma \vdash \phi \text{ is true}}$$

From Martin-Löf type theory to Homotopy type theory

What have we covered so for?

From Martin-Löf type theory to Homotopy type theory

What have we covered so for?

- $\top$ $\checkmark$

What have we covered so for?

- $\top$ ✓
- $\wedge$ ✓

What have we covered so for?

- $\top$ ✓
- $\wedge$ ✓
- $\Rightarrow$ ✓

What have we covered so for?

- $\top$ ✓
- $\wedge$ ✓
- $\Rightarrow$ ✓
- $\vee$ ✓

What have we covered so for?

- $\top$ ✓
- $\wedge$ ✓
- $\Rightarrow$ ✓
- $\vee$ ✓
- $\bot$ ✓

What have we covered so for?

- $\top$ ✓
- $\wedge$ ✓
- $\Rightarrow$ ✓
- $\vee$ ✓
- $\perp$ ✓

What's missing?

From Martin-Löf type theory to Homotopy type theory

What have we covered so for?

- $\top$ ✓
- $\wedge$ ✓
- $\Rightarrow$ ✓
- $\vee$ ✓
- $\bot$ ✓

What's missing?

- $\forall$
- $\exists$

What have we covered so for?

- $\top$ ✓
- $\wedge$ ✓
- $\Rightarrow$ ✓
- $\vee$ ✓
- $\perp$ ✓

What's missing?

- $\forall$
- $\exists$
- $=$

From Martin-Löf type theory to Homotopy type theory

- $$\frac{\Gamma, x : A \vdash B(x) \text{ is prop}}{\Gamma \vdash \forall(x : A).B(x) \text{ is prop}}$$

- $$\frac{\Gamma, x : A \vdash B(x) \text{ is prop}}{\Gamma \vdash \forall(x : A).B(x) \text{ is prop}}$$

- $$\frac{\Gamma, x : A \vdash p : B(x)}{\Gamma \vdash \lambda(x : A).p : \forall(x : A).B(x)}$$

- $$\frac{\Gamma, x : A \vdash B(x) \text{ is prop}}{\Gamma \vdash \forall(x : A).B(x) \text{ is prop}}$$

- $$\frac{\Gamma, x : A \vdash p : B(x)}{\Gamma \vdash \lambda(x : A).p : \forall(x : A).B(x)}$$

- $$\frac{\Gamma \vdash p : \forall(x : A).B(x) \quad \Gamma \vdash a : A}{\Gamma \vdash p(a) : B(a)}$$

- $$\frac{\Gamma, x : A \vdash B(x) \text{ is prop}}{\Gamma \vdash \exists(x : A).B(x) \text{ is prop}}$$

From Martin-Löf type theory to Homotopy type theory

- $$\frac{\Gamma, x : A \vdash B(x) \text{ is prop}}{\Gamma \vdash \exists(x : A).B(x) \text{ is prop}}$$

- $$\frac{\begin{array}{l}\Gamma, x : A \vdash B(x) \text{ is prop} \\ \Gamma \vdash a : A \qquad\qquad \Gamma \vdash b : B(a)\end{array}}{\Gamma \vdash (a, b) : \exists(x : A).B(x)}$$

From Martin-Löf type theory to Homotopy type theory

- $$\frac{\Gamma, x : A \vdash B(x) \text{ is prop}}{\Gamma \vdash \exists(x : A).B(x) \text{ is prop}}$$

- $$\frac{\begin{array}{l}\Gamma, x : A \vdash B(x) \text{ is prop} \\ \Gamma \vdash a : A \qquad \Gamma \vdash b : B(a)\end{array}}{\Gamma \vdash (a, b) : \exists(x : A).B(x)}$$

- $$\frac{\begin{array}{l}\Gamma, z : \exists(x : A).B(x) \vdash C(z) \text{ is prop} \\ \Gamma, x : A, p : B(x) \vdash r : C((x, p))\end{array}}{\Gamma, z : \exists(x : A).B(x) \vdash r[x := \pi_1 z][p := \pi_2 z] : C(z)}$$

From Martin-Löf type theory to Homotopy type theory

- $$\frac{\Gamma \vdash x : A \quad \Gamma \vdash y : A}{\Gamma \vdash x =_A y \text{ is prop}}$$

From Martin-Löf type theory to Homotopy type theory

- $$\frac{\Gamma \vdash x : A \quad \Gamma \vdash y : A}{\Gamma \vdash x =_A y \text{ is prop}}$$

- $$\frac{\Gamma \vdash a : A}{\Gamma \vdash \mathrm{refl}_a^A : a =_A a}$$

From Martin-Löf type theory to Homotopy type theory

- $$\frac{\Gamma \vdash x : A \quad \Gamma \vdash y : A}{\Gamma \vdash x =_A y \text{ is prop}}$$

- $$\frac{\Gamma \vdash a : A}{\Gamma \vdash \mathsf{refl}^A_a : a =_A a}$$

- $$\frac{\begin{array}{l}\Gamma, x : A, y : A, p : x =_A y \vdash C(x, y, p) \text{ is prop} \\ \Gamma, x : A \vdash d : C(x, x, \mathsf{refl}^A_x)\end{array}}{\Gamma, x : A, y : A, p : x =_A y \vdash d[x := x] : C(x, y, p)}$$

Non-Uniqueness of Identity proofs

Non-Uniqueness of Identity proofs

-

$$\frac{\begin{array}{c}\Gamma \vdash f, g : A \to B \\ \Gamma, x : A \vdash p : f(x) =_B g(x)\end{array}}{\Gamma \vdash \lambda(x : A).p : f =_{A \to B} g}$$

$$(f =_{A \to B} g) \triangleq \Pi_{(x:A)}(f(x) =_B g(x))$$

Non-Uniqueness of Identity proofs

- $$\frac{\Gamma \vdash f, g : A \to B \qquad \Gamma, x : A \vdash p : f(x) =_B g(x)}{\Gamma \vdash \lambda(x : A).p : f =_{A \to B} g}$$

  $$(f =_{A \to B} g) \triangleq \Pi_{(x:A)}(f(x) =_B g(x))$$

- $$\frac{\Gamma \vdash A, B : U \qquad \Gamma \vdash f : A \to B \qquad \Gamma \vdash g : B \to A \qquad \Gamma, x : A \vdash p_1 : g(f(x)) =_A x \qquad \Gamma, y : B \vdash p_2 : f(g(y)) =_B y}{\Gamma \vdash (f, g, (\lambda(x : A).p_1, \lambda(y : B).p_2)) : A =_U B}$$

  $$(A = B) \triangleq \Sigma_f \Sigma_g((\Pi_x(g(fx) = x)) \times (\Pi_y(f(gy) = y)))$$

Properties $=_{\_}$ must satisfy

- $$\frac{\Gamma \vdash a : A}{\Gamma \vdash r_a : a =_A a}$$

Properties $=_{\_}$ must satisfy

- $$\frac{\Gamma \vdash a : A}{\Gamma \vdash r_a : a =_A a}$$

- $$\frac{\Gamma \vdash p : a_1 =_A a_2}{\Gamma \vdash p^{-1} : a_2 =_A a_1}$$

Properties $=_{\_}$ must satisfy

- $$\frac{\Gamma \vdash a : A}{\Gamma \vdash r_a : a =_A a}$$

- $$\frac{\Gamma \vdash p : a_1 =_A a_2}{\Gamma \vdash p^{-1} : a_2 =_A a_1}$$

- $$\frac{\Gamma \vdash p : a_1 =_A a_2 \quad \Gamma \vdash q : a_2 =_A a_3}{\Gamma \vdash p \circ q : a_1 =_A a_3}$$

Properties $=_{\_}$ must satisfy

- $$\frac{\Gamma \vdash a : A}{\Gamma \vdash r_a : a =_A a}$$

- $$\frac{\Gamma \vdash p : a_1 =_A a_2}{\Gamma \vdash p^{-1} : a_2 =_A a_1}$$

- $$\frac{\Gamma \vdash p : a_1 =_A a_2 \quad \Gamma \vdash q : a_2 =_A a_3}{\Gamma \vdash p \circ q : a_1 =_A a_3}$$

- $$\frac{\Gamma \vdash P : \Pi_{(x:A)} B(x) \quad \Gamma \vdash q : a_1 =_A a_2}{\Gamma \vdash \mathrm{lift}_q^P : B(a_1) \to B(a_2)}$$

More properties

- $p \circ (q \circ r) =_{a_1 =_A a_4} (p \circ q) \circ r$

More properties

- $p \circ (q \circ r) =_{a_1 =_A a_4} (p \circ q) \circ r$
- $p \circ p^{-1} =_{a_1 =_A a_1} r_{a_1}$

More properties

- $p \circ (q \circ r) =_{a_1 =_A a_4} (p \circ q) \circ r$
- $p \circ p^{-1} =_{a_1 =_A a_1} r_{a_1}$
- $p^{-1} \circ p =_{a_2 =_A a_2} r_{a_2}$

More properties

- $p \circ (q \circ r) =_{a_1 =_A a_4} (p \circ q) \circ r$
- $p \circ p^{-1} =_{a_1 =_A a_1} r_{a_1}$
- $p^{-1} \circ p =_{a_2 =_A a_2} r_{a_2}$
- $p \circ r_{a_2} =_{a_1 =_A a_2} p$

More properties

- $p \circ (q \circ r) =_{a_1 =_A a_4} (p \circ q) \circ r$
- $p \circ p^{-1} =_{a_1 =_A a_1} r_{a_1}$
- $p^{-1} \circ p =_{a_2 =_A a_2} r_{a_2}$
- $p \circ r_{a_2} =_{a_1 =_A a_2} p$
- $r_{a_1} \circ p =_{a_1 =_A a_2} p$

Organizing the structure

$$x, x' : p =_{a_1 =_A a_2} p'$$

Organizing the structure

$$x, x' : p =_{a_1 =_A a_2} p'$$

$$x =_{p =_{a_1 =_A a_2} p'} x'$$

Organizing the structure

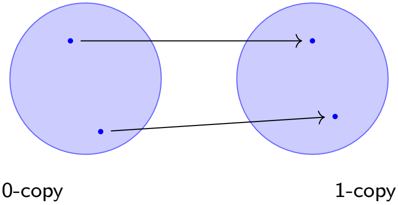
$$x, x' : p =_{a_1 =_A a_2} p'$$

$$x =_{p =_{a_1 =_A a_2} p'} x'$$

We need a way to organize the structure of paths.

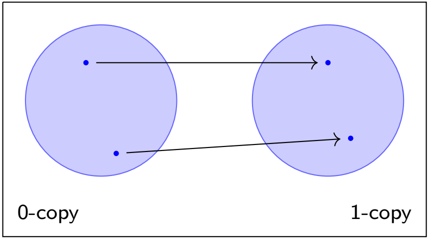
Organizing the structure

We need a way to organize the structure of paths.



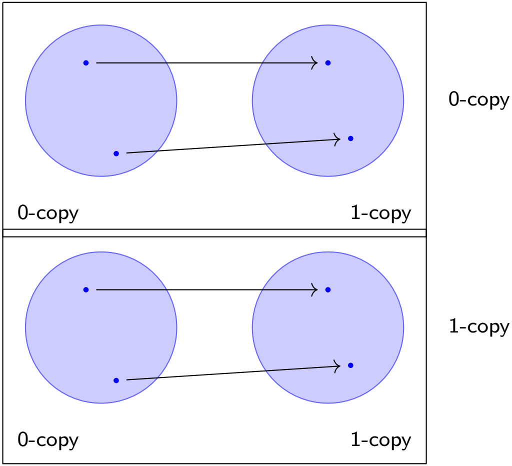
Organizing the structure

We need a way to organize the structure of paths.



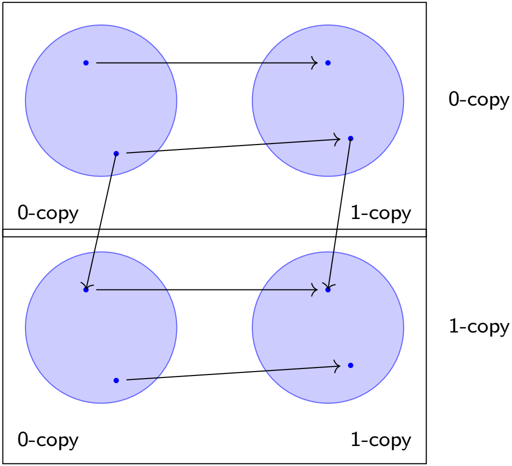
0-copy

We need a way to organize the structure of paths.



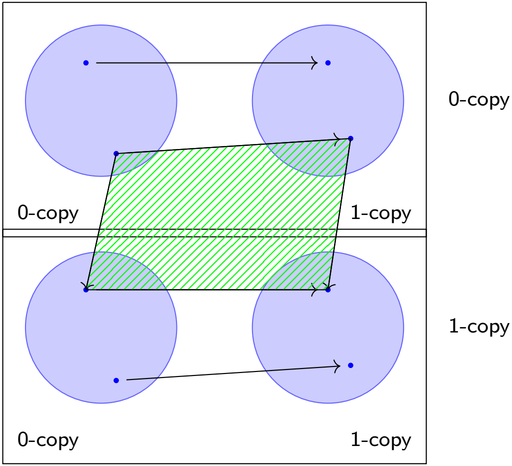
Organizing the structure

We need a way to organize the structure of paths.



Organizing the structure

We need a way to organize the structure of paths.



Cubical Type Theory

What is CTT?

Cubical Type Theory

What is CTT?

- Model of Homotopy Type Theory

Cubical Type Theory

What is CTT?

- Model of Homotopy Type Theory
- Based on cubical set

What is CTT?

- Model of Homotopy Type Theory
- Based on cubical set
- i.e., types with “primitive” internal structure

What is CTT?

- Model of Homotopy Type Theory
- Based on cubical set
- i.e., types with "primitive" internal structure

Essential concept: Dimensions

Cubical Type Theory

What is CTT?

- Model of Homotopy Type Theory
- Based on cubical set
- i.e., types with "primitive" internal structure

Essential concept: Dimensions

- Abstract interval

Cubical Type Theory

What is CTT?

- Model of Homotopy Type Theory
- Based on cubical set
- i.e., types with "primitive" internal structure

Essential concept: Dimensions

- Abstract interval
- Name

Cubical Type Theory

What is CTT?

- Model of Homotopy Type Theory
- Based on cubical set
- i.e., types with "primitive" internal structure

Essential concept: Dimensions

- Abstract interval
- Name
- Endpoints 0 and 1

Cubical Type Theory

What is CTT?

- Model of Homotopy Type Theory
- Based on cubical set
- i.e., types with “primitive” internal structure

Essential concept: Dimensions

- Abstract interval
- Name
- Endpoints 0 and 1
- Substitution

Cubical Types

What are cubical types?

Cubical Types

What are cubical types?

Objects at dimensions:

Cubical Types

What are cubical types?

Objects at dimensions:

- Dimension 0: usual points

Cubical Types

What are cubical types?

Objects at dimensions:

- Dimension 0: usual points
- Dimension $n + 1$: line between 2 objects of dimension n

Cubical Types

What are cubical types?

Objects at dimensions:

- Dimension 0: usual points
- Dimension $n + 1$: line between 2 objects of dimension n
- Types have terms at all dimensions

Cubical Types

What are cubical types?

Objects at dimensions:

- Dimension 0: usual points
- Dimension $n+1$: line between 2 objects of dimension n
- Types have terms at all dimensions

Types at dimensions:

Cubical Types

What are cubical types?

Objects at dimensions:

- Dimension 0: usual points
- Dimension $n+1$: line between 2 objects of dimension n
- Types have terms at all dimensions

Types at dimensions:

- Types of dim. 0: usual types

Cubical Types

What are cubical types?

Objects at dimensions:

- Dimension 0: usual points
- Dimension $n+1$: line between 2 objects of dimension n
- Types have terms at all dimensions

Types at dimensions:

- Types of dim. 0: usual types
- Types of dim. $n+1$: type line between n-types

Cubical Types

What are cubical types?

Objects at dimensions:

- Dimension 0: usual points
- Dimension $n+1$: line between 2 objects of dimension n
- Types have terms at all dimensions

Types at dimensions:

- Types of dim. 0: usual types
- Types of dim. $n+1$: type line between n-types
  - Elements: lines

Kan filling

What about open boxes?

Kan filling

What about open boxes?

- ($\rightarrow$ board)

Kan filling

What about open boxes?

- ($\rightarrow$ board)
- Must be defined <u>for every type</u>

Kan filling

What about open boxes?

- ($\rightarrow$ board)
- Must be defined <u>for every type</u>
  - ...depending on its shape

Kan filling

What about open boxes?

- ($\rightarrow$ board)
- Must be defined for every type
  - ...depending on its shape
- Explicit term hcom

Syntax

$$\begin{aligned} M := {} & (a{:}A) \to B \mid (a{:}A) \times B \mid \mathsf{Id}_{x.A}(M, N) \mid \mathsf{bool} \mid \mathsf{not}_r \mid \mathbb{S}^1 \\ & \mid \lambda a.M \mid \mathsf{app}(M, N) \mid \langle M, N \rangle \mid \mathsf{fst}(M) \mid \mathsf{snd}(M) \mid \langle x \rangle M \mid M@r \\ & \mid \mathsf{true} \mid \mathsf{false} \mid \mathsf{if}_{a.A}(M; N_1, N_2) \mid \mathsf{notel}_r(M) \\ & \mid \mathsf{base} \mid \mathsf{loop}_r \mid \mathbb{S}^1\text{-}\mathsf{elim}_{a.A}(M; N_1, x.N_2) \\ & \mid \mathsf{coe}_{x.A}^{r \rightsquigarrow r'}(M) \mid \mathsf{hcom}_A^{\overrightarrow{r_i}}(r \rightsquigarrow r', M; \overrightarrow{y.N_i^\varepsilon}) \end{aligned}$$

Notations:

a	Term variable
x	Dim. variable
r	Dim. term (0, 1, *x*)

Typing judgments

- Shape: $\Psi; \Gamma \vdash a : A$ ($\Psi; \Gamma \vdash A : \mathcal{U}$)
- $$\frac{\Psi; \Gamma \vdash A : \mathcal{U} \qquad \Psi; \Gamma, a : A \vdash B : \mathcal{U}}{\Psi; \Gamma \vdash (a : A) \to B : \mathcal{U}}$$

Language: typing

Typing judgments

- Shape: $\Psi; \Gamma \vdash a : A$ $(\Psi; \Gamma \vdash A : \mathcal{U})$
- $$\frac{\Psi; \Gamma \vdash A : \mathcal{U} \qquad \Psi; \Gamma, a : A \vdash B : \mathcal{U}}{\Psi; \Gamma \vdash (a : A) \to B : \mathcal{U}}$$
- $$\frac{\Psi; \Gamma, a : A \vdash M : B}{\Psi; \Gamma \vdash \lambda a.M : A}$$

Typing judgments

- Shape: $\Psi; \Gamma \vdash a : A$ ($\Psi; \Gamma \vdash A : \mathcal{U}$)
- $$\frac{\Psi; \Gamma \vdash A : \mathcal{U} \qquad \Psi; \Gamma, a : A \vdash B : \mathcal{U}}{\Psi; \Gamma \vdash (a : A) \to B : \mathcal{U}}$$
- $$\frac{\Psi; \Gamma, a : A \vdash M : B}{\Psi; \Gamma \vdash \lambda a.M : A}$$
- $$\frac{\Psi, x; \Gamma \vdash A : \mathcal{U} \qquad \Psi; \Gamma \vdash M : A\langle r/x \rangle}{\Psi; \Gamma \vdash \mathrm{coe}_{x.A}^{r \rightsquigarrow r'}(M) : A\langle r'/x \rangle}$$

Language: typing

Typing judgments

- Shape: $\Psi; \Gamma \vdash a : A$ ($\Psi; \Gamma \vdash A : \mathcal{U}$)
- $$\frac{\Psi; \Gamma \vdash A : \mathcal{U} \qquad \Psi; \Gamma, a : A \vdash B : \mathcal{U}}{\Psi; \Gamma \vdash (a : A) \to B : \mathcal{U}}$$
- $$\frac{\Psi; \Gamma, a : A \vdash M : B}{\Psi; \Gamma \vdash \lambda a.M : A}$$
- $$\frac{\Psi, x; \Gamma \vdash A : \mathcal{U} \qquad \Psi; \Gamma \vdash M : A\langle r/x\rangle}{\Psi; \Gamma \vdash \mathrm{coe}^{r \rightsquigarrow r'}_{x.A}(M) : A\langle r'/x\rangle}$$

Language: reduction

Interesting reduction rules

Language: reduction

Interesting reduction rules

$$\frac{}{\mathsf{loop}_x \ \mathsf{val}}$$

Language: reduction

Interesting reduction rules

$$\frac{}{\mathsf{loop}_x \ \mathsf{val}}$$

$$\frac{}{\mathsf{loop}_\varepsilon \longmapsto \mathsf{base}}$$

Language: reduction

Interesting reduction rules

$$\frac{}{\mathsf{loop}_x\ \mathsf{val}}$$

$$\frac{}{\mathsf{loop}_\varepsilon \longmapsto \mathsf{base}}$$

$$\frac{}{\mathsf{coe}^{r \rightsquigarrow r'}_{x.\mathsf{bool}}(M) \longmapsto M}$$

Language: reduction

Interesting reduction rules

$$\overline{\mathsf{loop}_x\ \mathsf{val}}$$

$$\overline{\mathsf{loop}_\varepsilon \longmapsto \mathsf{base}}$$

$$\overline{\mathsf{coe}^{r \rightsquigarrow r'}_{x.\mathsf{bool}}(M) \longmapsto M}$$

$$\overline{\mathsf{coe}^{r \rightsquigarrow r'}_{x.(a:A)\to B}(M) \longmapsto \lambda a.\mathsf{coe}^{r \rightsquigarrow r'}_{x.B[\mathsf{coe}^{r' \rightsquigarrow x}_{x.A}(a)/a]}(\mathsf{app}(M, \mathsf{coe}^{r' \rightsquigarrow r}_{x.A}(a)))}$$

Interesting reduction rules

$$\frac{}{\mathsf{loop}_x\ \mathsf{val}}$$

$$\frac{}{\mathsf{loop}_\varepsilon \longmapsto \mathsf{base}}$$

$$\frac{}{\mathsf{coe}^{r \rightsquigarrow r'}_{x.\mathsf{bool}}(M) \longmapsto M}$$

$$\frac{}{\mathsf{coe}^{r \rightsquigarrow r'}_{x.(a:A)\to B}(M) \longmapsto \lambda a.\mathsf{coe}^{r \rightsquigarrow r'}_{x.B[\mathsf{coe}^{r' \rightsquigarrow x}_{x.A}(a)/a]}(\mathsf{app}(M, \mathsf{coe}^{r' \rightsquigarrow r}_{x.A}(a)))}$$

- Are your ready for *hcom*?

Language: reduction on the loose

$$\frac{F = \mathsf{hcom}_A^{\overrightarrow{r_i}}(r \rightsquigarrow z, \mathsf{fst}(M); \overrightarrow{y.\mathsf{fst}(N_i^\varepsilon)})}{\begin{array}{c}\mathsf{hcom}_{(a:A)\times B}^{\overrightarrow{r_i}}(r \rightsquigarrow r', M; \overrightarrow{y.N_i^\varepsilon}) \\ \longmapsto \\ \langle \mathsf{hcom}_A^{\overrightarrow{r_i}}(r \rightsquigarrow r', \mathsf{fst}(M); \overrightarrow{y.\mathsf{fst}(N_i^\varepsilon)}), \mathsf{com}_{z.B[F/a]}^{\overrightarrow{r_i}}(r \rightsquigarrow r', \mathsf{snd}(M); \overrightarrow{y.\mathsf{snd}(N_i^\varepsilon)})\rangle\end{array}}$$

$$\frac{}{\mathsf{hcom}_{(a:A)\to B}^{\overrightarrow{r_i}}(r \rightsquigarrow r', M; \overrightarrow{y.N_i^\varepsilon}) \longmapsto \lambda a.\mathsf{hcom}_B^{\overrightarrow{r_i}}(r \rightsquigarrow r', \mathsf{app}(M, a); \overrightarrow{y.\mathsf{app}(N_i^\varepsilon, a)})}$$

And now?

Why are these systems desirable?

And now?

Why are these systems desirable?

- Much easier to use than plain HoTT

And now?

Why are these systems desirable?

- Much easier to use than plain HoTT
- More integrated

And now?

Why are these systems desirable?

- Much easier to use than plain HoTT
- More integrated
- For instance, no refl_t

And now?

Why are these systems desirable?

- Much easier to use than plain HoTT
- More integrated
- For instance, no refl_t
- Functions apply to paths automatically

And now?

Why are these systems desirable?

- Much easier to use than plain HoTT
- More integrated
- For instance, no refl_t
- Functions apply to paths automatically
- Shorter proofs

And now?

Why are these systems desirable?

- Much easier to use than plain HoTT
- More integrated
- For instance, no refl_t
- Functions apply to paths automatically
- Shorter proofs
- More advanced things that are beyond us :-)

And now?

Why are these systems desirable?

- Much easier to use than plain HoTT
- More integrated
- For instance, no refl_t
- Functions apply to paths automatically
- Shorter proofs
- More advanced things that are beyond us :-)

Examples

Some examples and ideas

Examples

Some examples and ideas

- Polar + cartesian coordinates

Examples

Some examples and ideas

- Polar + cartesian coordinates
- Modules

Examples

Some examples and ideas

- Polar + cartesian coordinates
- Modules
- Languages representations

Examples

Some examples and ideas

- Polar + cartesian coordinates
- Modules
- Languages representations
    - Compiler optimization

Summary

- Proof relevant notion of equality endows types with richer structure.

Summary

- Proof relevant notion of equality endows types with richer structure.
- We can work in such a type theory modulo an equivalence congruence relation weaker than strong equality.

Summary

- Proof relevant notion of equality endows types with richer structure.
- We can work in such a type theory modulo an equivalence congruence relation weaker than strong equality.
- Cubical sets provide us a model to work with such type theories.

Summary

- Proof relevant notion of equality endows types with richer structure.
- We can work in such a type theory modulo an equivalence congruence relation weaker than strong equality.
- Cubical sets provide us a model to work with such type theories.
- With this, it might be easier to express formally and constructively more of mathematics and other languages.

References

Carlo Angiuli, Robert Harper, and Todd Wilson, Computational higher dimensional type theory, Proceedings of the 44th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (New York, NY, USA), POPL '17, ACM, 2017.

Martin Hofmann and Thomas Streicher, The groupoid interpretation of type theory, Twenty-five years of constructive type theory (Venice, 1995), Oxford Logic Guides, vol. 36, Oxford Univ. Press, New York, 1998, pp. 83–111.

Per Martin-Löf, An intuitionistic theory of types, Twenty-five years of constructive type theory, Oxford Logic Guides, vol. 36, Oxford University Press, 1998, pp. 127–172.

The Univalent Foundations Program, Homotopy type theory: Univalent foundations of mathematics, `https://homotopytypetheory.org/book`, Institute for Advanced Study, 2013.

Questions?

Thank you.