A Cubical Type Theory

Simon Huber

(j.w.w. Cyril Cohen, Thierry Coquand, Anders Mörtberg)

University of Gothenburg

HoTT and UF – Mini-Symposium at DMV 2015

Hamburg, September 25, 2015

Cubical Type Theory: Overview

- Type theory where we can directly argue about *n*-dimensional cubes (points, lines, squares, cubes, ...).
- Based on a *constructive* model of type theory in cubical sets with connections and diagonals.
- Π, Σ, data types, U
- path types and identity types
- The Univalence Axiom and function extensionality are provable.
- Some higher inductive types with "good" definitional equalities

Basic Idea

Expressions may depend on *names* $i, j, k, \ldots$ ranging over an interval $\mathbb{I}$. E.g.,

$$x : A, i : \mathbb{I}, y : B(i, x) \vdash u(x, i) : C(x, i, y)$$

is a line connecting the two points

$$x : A, y : B(0, x) \vdash u(x, 0) : C(x, 0, y)$$
$$x : A, y : B(1, x) \vdash u(x, 1) : C(x, 1, y)$$

Each line $i : \mathbb{I} \vdash t(i) : A$ gives an equality

$$\vdash \langle i \rangle\, t(i) : \mathrm{Path}\, A\, t(0)\, t(1)$$

The Interval $\mathbb{I}$

- Given by $r, s ::= 0 \mid 1 \mid i \mid 1 - i \mid r \wedge s \mid r \vee s$
- i ranges over *names* or *symbols*
- Intuition: i an element of $[0, 1]$, $\wedge$ is min, and $\vee$ is max.
- Equality is the equality in the free bounded distributive lattice with generators i, $1 - i$.
- De Morgan algebra via

$$
\begin{aligned}
1 - 0 &= 1 & \qquad 1 - (r \wedge s) &= (1 - r) \vee (1 - s) \\
1 - 1 &= 0 & \qquad 1 - (r \vee s) &= (1 - r) \wedge (1 - s) \\
& & \qquad 1 - (1 - i) &= i
\end{aligned}
$$

NB: $i \wedge (1 - i) \neq 0$ and $i \vee (1 - i) \neq 1$!

Overview of the Syntax

$A, B, a, b, u, v ::=$	x	variables
	$\mid (x : A) \to B \mid \lambda x : A.\, u \mid u\, v$	Π-types
	$\mid (x : A) \times B \mid (u, v) \mid u.1 \mid v.2$	Σ-types
	$\mid \mathsf{U}$	universe
	$\mid \mathsf{Path}\, A\, a\, b$	path types
	$\mid \langle i \rangle u$	name abstraction
	$\mid u\, r$	interval application
	$\mid \mathsf{comp}^i\, A\, u\, \vec{u}$	composition
	$\mid \mathsf{Glue}\, A\, \vec{u} \mid (a, \vec{u})$	glueing
	$\mid \ldots$	data types. . .

Contexts and Substitutions

Contexts

$$\frac{}{() \vdash} \qquad \frac{\Gamma \vdash A}{\Gamma, x : A \vdash} \qquad \frac{\Gamma \vdash}{\Gamma, i : \mathbb{I} \vdash}$$

Substitutions are as usual but we also allow to assign an element in the interval to a name:

$$\frac{\sigma \colon \Delta \to \Gamma \qquad \Delta \vdash r : \mathbb{I}}{(\sigma, i = r) \colon \Delta \to \Gamma, i : \mathbb{I}}$$

Face Operations

Certain substitutions correspond to face operations. E.g.:

$$(x = x, i = 0, y = y)\colon (x : A, y : B(i = 0)) \to (x : A, i : \mathbb{I}, y : B)$$

In general a face operation are $\alpha\colon \Gamma\alpha \to \Gamma$ setting some names to 0 or 1 and otherwise the identity.

Faces are determined by all the assignments $i = b$, $b \in \{0, 1\}$; write

$$\alpha = (i_1 = b_1) \dots (i_n = b_n)$$

(Special case: $\alpha = \mathrm{id}$)

Basic Typing Rules

$$\frac{\Gamma \vdash}{\Gamma \vdash x : A} \quad (x : A \text{ in } \Gamma) \qquad\qquad \frac{\Gamma \vdash}{\Gamma \vdash i : \mathbb{I}} \quad (i : \mathbb{I} \text{ in } \Gamma)$$

$$\frac{\Gamma, x : A \vdash B}{\Gamma \vdash (x : A) \to B} \qquad\qquad \frac{\Gamma, x : A \vdash v : B}{\Gamma \vdash \lambda x : A.\, v : (x : A) \to B}$$

$$\frac{\Gamma \vdash w : (x : A) \to B \qquad \Gamma \vdash u : A}{\Gamma \vdash w\, u : B(x = u)}$$

Also: Sigma types and data types . . .

Path Types

$$\frac{\Gamma \vdash A \qquad \Gamma \vdash a : A \qquad \Gamma \vdash b : A}{\Gamma \vdash \mathsf{Path}\, A\, a\, b}$$

$$\frac{\Gamma \vdash A \qquad \Gamma, i : \mathbb{I} \vdash u : A}{\Gamma \vdash \langle i \rangle u : \mathsf{Path}\, A\, u(i = 0)\, u(i = 1)}$$

$$\frac{\Gamma \vdash w : \mathsf{Path}\, A\, a\, b \qquad \Gamma \vdash r : \mathbb{I}}{\Gamma \vdash w\, r : A}$$

$$(\langle i \rangle u)\, r = u(i = r)$$
$$\langle i \rangle\, u i = u$$

$$\frac{\Gamma \vdash w : \mathsf{Path}\, A\, a\, b}{\begin{array}{c}\Gamma \vdash w\, 0 = a : A \\ \Gamma \vdash w\, 1 = b : A\end{array}}$$

Path Type

- Reflexivity $a : A \vdash \mathsf{refl}\ a : \mathsf{Path}\, A\, a\, a$ is given by the constant path
$$\mathsf{refl}\, a = \langle i \rangle a$$
- Singletons are contractible: for $a : A$ and $S_a = (x : A) \times (\mathsf{Path}\, A\, a\, x)$ we have
$$\langle i \rangle (p\, i, \langle j \rangle p\, (i \wedge j)) : \mathsf{Path}\, S_a\, (a, \mathsf{refl}\, a)\, (x, p)$$
for $(x, p) : S_a$.

Function Extensionality

For f and g of type $C = (x : A) \to B$ and
$w : (x : A) \to \mathsf{Path}\, B\, (f\, x)\, (g\, x)$ we have

$$\langle i \rangle \lambda x : A.\, w\, x\, i \quad : \quad \mathsf{Path}\;\, C\, f\, g$$

Kan Operations

Given $i : \mathbb{I} \vdash A$ we want an equivalence between $A(i0)$ and $A(i1)$.

Require additional composition operations.

Refinement of Kan's extension condition (1955)

"Any open box can be filled"

Systems

A system

$$\vec{u} = [\alpha \mapsto u_\alpha]$$

for $\Gamma \vdash A$ is given by a family of *compatible* terms

$$\Gamma\alpha \vdash u_\alpha : A\alpha$$

(α ranging over a set of faces L, L downwards closed)

A system

$$\Gamma\alpha \vdash u_\alpha : A\alpha \quad (\alpha \in L)$$

can be considered as *partial element* of $\Gamma \vdash A$ with *extent* L. We call $\vec{u}$ *connected* if there is a $\Gamma \vdash u : A$ such that:

$$\Gamma\alpha \vdash u\alpha = u_\alpha : A\alpha$$

For example, if L is generated by the faces

$$(i = 0), (i = 1), (j = 0), (j = 1)$$

then a system corresponds to a boundary of a square. It is connected if the boundary of the square can be filled.

Compositions

$$\frac{\Gamma, i : \mathbb{I} \vdash A \qquad \Gamma \vdash u : A(i=0) \qquad \Gamma\alpha, i : \mathbb{I} \vdash u_\alpha : A\alpha \quad (\alpha \in L) \qquad \Gamma\alpha \vdash u\alpha = u_\alpha(i=0) : A\alpha(i=0)}{\Gamma \vdash \mathsf{comp}^i\, A\, u\, \vec{u} : A(i=1)}$$

$$(\mathsf{comp}^i\, A\, u\, \vec{u})\alpha = u_\alpha(i=1) \qquad \text{if } \alpha \in L$$

$$(\mathsf{comp}^i\, A\, u\, \vec{u})\sigma = \mathsf{comp}^j\, A(\sigma, i=j)\, u\sigma\, \vec{u}(\sigma, i=j)$$

Filling

There is also an operation

$$\Gamma, i : \mathbb{I} \vdash \mathsf{fill}^i\, A\, u\, \vec{u} : A$$

connecting u to $\mathsf{comp}^i\, A\, u\, \vec{u}$. This can be defined using compositions:

$$\mathsf{fill}^i\, A(i)\, u\, \vec{u}(i) = \mathsf{comp}^j\ A(i \wedge j)\ u\ [\alpha \mapsto u_\alpha(i \wedge j), (i = 0) \mapsto u]$$

Special case: path lifting property ($\vec{u} = [\,]$)

Composition

$\mathsf{comp}^i\ A\,u\,\vec{u}$ is defined by induction on the type A:

- Case $i : \mathbb{I} \vdash A = \mathsf{Path}\,B\,b_0\,b_1$.

$$\mathsf{comp}^i\ A\,u\,\vec{u} =$$
$$\langle j\rangle\ \mathsf{comp}^i\ B\,(u\,j)\,[\alpha \mapsto u_\alpha j, (j=0) \mapsto b_0, (j=1) \mapsto b_1]$$

- Case $i : \mathbb{I} \vdash A = (x : B) \to C$. For $b_1 : B(i1)$

$$\mathsf{comp}^i\ A\,f\,\vec{g}\,b_1 = \mathsf{comp}^j\ C(i=j, x=b)\,(f\,b_0)\,(\vec{g}(i=j)\,b)$$

with $b = \mathsf{fill}^{-j}\ B(i=j)\,b_1\,[\,]$ and $b_0 = b(j=0) : B(i=0)$.

Judgmental equalities are given by unfolding the definitions.

Glue

To justify composition for U and univalence we add *glueing*.

Given a system of equivalences on a type we introduce a new type:

$$\frac{\Gamma \vdash A \qquad \Gamma\alpha \vdash f_\alpha : \mathsf{Equiv}\ T_\alpha\ A\alpha \quad (\alpha \in L)}{\Gamma \vdash \mathsf{Glue}\ A\ \vec{f}}$$

$$\frac{\Gamma \vdash a : A \qquad \Gamma\alpha \vdash t_\alpha : T_\alpha \qquad \Gamma\alpha \vdash f_\alpha t_\alpha = a\alpha : A\alpha}{\Gamma \vdash (a, \vec{t}) : \mathsf{Glue}\ A\ \vec{f}}$$

$$(\mathsf{Glue}\ A\ \vec{f})\alpha = T_\alpha \qquad (a, \vec{t})\alpha = t_\alpha \qquad \text{if } \alpha \in L$$

$$(\mathsf{Glue}\ A\ \vec{f})\sigma = \mathsf{Glue}\ A\sigma\ \vec{f}\sigma \qquad (a, \vec{t})\sigma = (a\sigma, \vec{t}\sigma)$$

Compositions for the Universe

We also can define composition for Glue $A\,\vec{f}$.

For the universe U, we can reduce composition in U to Glue.

Any path $P : \mathsf{Path}\,\mathsf{U}\,A\,B$ induces an equivalence $P^+ : \mathsf{Equiv}\,A\,B$ whose function part is given by:

$$a : A \vdash \mathsf{comp}^i\,(P\,i)\,a\,[\,] : B$$

Univalence from Glue

Using Glue we can also prove the Univalence Axiom! Main ingredients:

- Given an equivalence f : Equiv $A\,B$ we can construct a path E_f : Path U $A\,B$ by

$$E_f = \langle i \rangle\ \mathsf{Glue}\,B\,[(i=0) \mapsto f, (i=1) \mapsto (\langle k \rangle B)^+]$$

- Starting from P : Path U $A\,B$ we can also construct a square

$$\mathsf{Path}\,(\mathsf{Path}\,\mathsf{U}\,A\,B)\ E_{P^+}\ \ P$$

  using Glue:

$$\begin{aligned}\langle j\ i \rangle\ \ \mathsf{Glue}\,B\,[(i=0) &\mapsto P^+, \\ (i=1) &\mapsto (\langle k \rangle B)^+, \\ (j=1) &\mapsto (\langle k \rangle P\,(i \vee k))^+]\end{aligned}$$

Identity Types

For the path type Path $A\,u\,v$ we can define the J-eliminator.
But: the usual definitional equality only holds propositional.

Recently, Andrew Swan found a way to recover an identity type Id $A\,u\,v$ (based on a cofibration/trivial fibration factorization). This identity type interprets the definitional equality for J!

Identity Types

$$\frac{\Gamma \vdash w : \mathsf{Path}\, A\, u\, v \qquad \Gamma\alpha \vdash w\alpha = \langle i \rangle\, u\alpha : \mathsf{Path}\, A\alpha\, u\alpha\, v\alpha \quad (\alpha \in L)}{\Gamma \vdash (w, L) : \mathsf{Id}\, A\, u\, v}$$

The system L remembers where w is constant.

For $u : A$ define $\mathsf{refl}\, u$ as $(\langle i \rangle u, \mathbf{1})$, where $\mathbf{1}$ is the maximal system generated by the identity.

One can define J with the usual definitional equality!

We expect univalence also to hold for Id.

Implementation: Cubicaltt

Prototype proof-assistant implemented in Haskell.

Based on: "A simple type-theoretic language: Mini-TT",
T. Coquand, Y. Kinoshita, B. Nordström, M. Takeya (2008).

Mini-TT is a variant of Martin-Löf type theory with data types.
Cubicaltt extends Mini-TT with:

- name abstraction and application
- identity types
- composition
- equivalences can be transformed into equalities (glueing)
- some higher inductive types (experimental)

Try it: `https://github.com/mortberg/cubicaltt`

Further Work

- Formal correctness proof of model and implementation
- Proof of normalization and decidability of type-checking
- Related work: Brunerie/Licata, Polonsky, Altenkirch/Kaposi, Bernardy/Coquand/Moulin

Semantics

Consider the category $\mathcal{C}_{\mathsf{dM}}$ with objects finite sets of names $I, J, K, \ldots$ and a morphism $I \to J$ is a map $J \to \mathsf{dM}(I)$ where $\mathsf{dM}(I)$ is the free De Morgan algebra on the generators I.

A context $\Gamma \vdash$ is a presheaf on $\mathcal{C}$, i.e.,

- given by sets $\Gamma(I)$ for each I,
- and maps $\Gamma(J) \to \Gamma(I), \rho \mapsto \rho f$ for each $f \colon I \to J$ s.t.

$$(\rho f)g = \rho(fg) \quad \text{and} \quad \rho\,\mathrm{id} = \rho$$

The interval $\mathbb{I}$ is interpreted as the presheaf $\mathbb{I}(J) = \mathsf{dM}(J)$.

Semantics

A type $\Gamma \vdash A$ is given by a presheaf on the category of elements of Γ, i.e.,

- given by a family of sets $A\rho$ for each $\rho \in \Gamma(I)$,
- and maps $A\rho \to A(\rho f), a \mapsto af$ s.t.

$$(af)g = a(fg) \quad \text{and} \quad a\,\mathrm{id} = a$$

Moreover, we require a *composition structure*: for each $\rho \in \Gamma(I, i)$, family of compatible elements $u_\alpha \in A\rho\alpha$ ($\alpha \in L$, i not in α), and $u \in A\rho(i = 0)$ s.t. $u\alpha = u_\alpha(i = 0)$, there is

$$\mathsf{comp}^i(A\rho)\,u\,\vec{u} \;\in\; A\rho(i = 1)$$

such that

$$(\mathsf{comp}^i(A\rho)\,u\,\vec{u})f = \mathsf{comp}^j(A\rho(f, i = j))\,uf\,\vec{u}f \qquad \text{for } f : I \to J$$

$$(\mathsf{comp}^i(A\rho)\,u\,\vec{u})\alpha = u_\alpha(i = 1)$$

Examples of HITs: Propositional Truncation

$$\frac{\Gamma \vdash A}{\Gamma \vdash \mathsf{inh}\, A} \qquad \frac{\Gamma \vdash a : A}{\Gamma \vdash \mathsf{inc}\, a : \mathsf{inh}\, A}$$

$$\frac{\Gamma \vdash u : \mathsf{inh}\, A \qquad \Gamma \vdash v : \mathsf{inh}\, A \qquad \Gamma \vdash r : \mathbb{I}}{\Gamma \vdash \mathsf{squash}\; u\; v\; r : \mathsf{inh}\, A} \qquad \begin{array}{l} \mathsf{squash}\; u\; v\; 0 = u \\ \mathsf{squash}\; u\; v\; 1 = v \end{array}$$

$$\frac{\Gamma \vdash u : \mathsf{inh}\, A \qquad \Gamma\alpha, i : \mathbb{I} \vdash u_\alpha \qquad \Gamma\alpha \vdash u_\alpha(i = 0) = u\alpha : \mathsf{inh}\, A\alpha}{\begin{array}{c} \Gamma \vdash \mathsf{hcomp}^i\; u\; \vec{u} : \mathsf{inh}\, A \\ \Gamma\alpha \vdash (\mathsf{hcomp}^i\; u\; \vec{u})\alpha = u_\alpha(i = 1) : \mathsf{inh}\, A\alpha \end{array}}$$

One can define compositions for inh A (uses compositions in A).