A syntax for cubical type theory

Thorsten Altenkich, Ambrus Kaposi

University of Nottingham

Workshop on Homotopy Type Theory, Oxford
7th November 2014

Goal

- Homotopy Type Theory adds the univalence axiom to Type Theory.
  - The theory becomes extensional.

$$e : \mathsf{Bool} =_{\mathsf{U}} \mathsf{Bool}$$
$$e :\equiv \mathsf{univ}\,(\mathsf{not}, \ldots)$$

  - However, we don't know how to run certain programs:

$$\mathsf{coe} : (A =_{\mathsf{U}} B) \to A \to B$$
$$\mathsf{coe}\,(\mathsf{refl}\,A)\,a :\equiv a$$
$$b : \mathsf{Bool}$$
$$b :\equiv \mathsf{coe}\,e\,\mathsf{true}$$

  - We don't know how to compute b in general.
- Our goal is to fix this:
  - Define a type theory where univalence is admissible and every closed term of type Bool computes to true or false.

Plan of action

- Homotopy Type Theory teaches us that equality can be described individually for each type former, eg.:

$$
\begin{array}{lrcl}
\text{natural numbers:} & (\mathsf{zero} =_{\mathbb{N}} \mathsf{zero}) & \simeq & 1 \\
 & (\mathsf{zero} =_{\mathbb{N}} \mathsf{suc}\, m) & \simeq & 0 \\
 & (\mathsf{suc}\, m =_{\mathbb{N}} \mathsf{zero}) & \simeq & 0 \\
 & (\mathsf{suc}\, m =_{\mathbb{N}} \mathsf{suc}\, n) & \simeq & (m =_{\mathbb{N}} n) \\
\text{pairs:} & ((a, b) =_{A\times B} (a', b')) & \simeq & (a =_A a' \times b =_B b') \\
\text{functions:} & (f =_{A\to B} g) & \simeq & (\Pi(x : A).f\, x =_B g\, x) \\
\text{types:} & (A =_{\mathsf{U}} B) & \simeq & (A \simeq B)
\end{array}
$$

- Let's define equality separately for each type former, as above!
  - We start with Martin-Löf Type Theory without the identity type. We define identity by recursion on the type formers as above.

Inspiration and structure of talk

This work is based on the following papers:

- Altenkirch, McBride, Swierstra: Observational Equality, Now! 2007
- Bernardy, Jansson, Paterson: Parametricity for dependent types, 2012
- Bernardy, Moulin: A computational interpretation of parametricity, 2012
- Bezem, Coquand, Huber: A cubical set model of type theory, 2013

Structure of talk:

We need a heterogeneous equality

- The reason is type dependency
- Dependent pairs – the equality of the second components depends on the equality of the first components, eg.:

$$((m, xs) =_{\Sigma(i:\mathbb{N}).Vec\ i} (n, ys)) \simeq (\Sigma(r : m =_{\mathbb{N}} n).r \vdash xs =_{Vec\,-} ys)$$

- We add a heterogeneous equality:

$$\frac{a : A \quad b : B \quad e : A =_{\mathsf{U}} B}{a \sim_e b : \mathsf{U}}$$

$$\frac{xs : Vec\ m \quad ys : Vec\ n \quad \dfrac{r : m =_{\mathbb{N}} n}{\mathsf{ap}\ Vec\ r : Vec\ m =_{\mathsf{U}} Vec\ n}}{xs \sim_{\mathsf{ap}\ Vec\ r}\ ys : \mathsf{U}}$$

Heterogeneous equality (i)

- Specification:

$$\frac{\Gamma \vdash}{\Gamma^= \vdash} \qquad \frac{\Gamma \vdash A : \mathsf{U}}{\Gamma^= \vdash \sim_A : A[0] \to A[1] \to \mathsf{U}} \qquad 0_\Gamma, 1_\Gamma : \Gamma^= \Rightarrow \Gamma$$

- The operation $-^=$:

$$\begin{aligned} \emptyset^= &\equiv \emptyset \\ (\Gamma, x : A)^= &\equiv \Gamma^=, x_0 : A[0_\Gamma], x_1 : A[1_\Gamma], x_2 : x_0 \sim_A x_1 \end{aligned}$$

- Substitutions 0, 1 project out the corresponding components:

$$\begin{aligned} i_\emptyset &\equiv () &&: \emptyset \Rightarrow \emptyset \\ i_{\Gamma,A} &\equiv (i_\Gamma, x \mapsto x_i) &&: (\Gamma, x : A)^= \Rightarrow \Gamma, x : A \end{aligned}$$

$$\begin{tikzcd} \Gamma^= \arrow[d, bend right, "1_\Gamma"'] \arrow[d, bend left, "0_\Gamma"] \\ \Gamma \end{tikzcd}$$

Heterogeneous equality (ii)

Heterogeneous equality type defined as in "Plan of action":

$$\frac{\Gamma \vdash A : \mathsf{U}}{\Gamma^{=} \vdash \sim_A : A[0] \to A[1] \to \mathsf{U}}$$

$$\begin{aligned}
f_0 \sim_{\Pi(x:A).B} f_1 &\equiv \Pi(x_0 : A[0], x_1 : A[1], x_2 : x_0 \sim_A x_1).f_0\, x_0 \sim_B f_1\, x_1 \\
(a, b) \sim_{\Sigma(x:A).B} (a', b') &\equiv \Sigma(x_2 : a \sim_A a').b \sim_B [x_0 \mapsto a, x_1 \mapsto a']\, b' \\
A \sim_{\mathsf{U}} B &\equiv A \to B \to \mathsf{U} \text{ (parametricity)} \\
A \sim_{\mathsf{U}} B &\equiv A \simeq B \text{ (later)}
\end{aligned}$$

$-^{=}$ is an endofunctor on the category of contexts

- Action on substitutions:

$$\begin{aligned} ()^{=} &\equiv () \\ (\rho, x \mapsto t)^{=} &\equiv (\rho^{=}, x_0 \mapsto t[0], x_1 \mapsto t[1], x_2 \mapsto t^*) \end{aligned}$$

- Terms respect this equality (Reynold's abstraction theorem):

$$\frac{\Gamma \vdash t : A}{\Gamma^{=} \vdash t^* : t[0] \sim_A t[1]}$$

$$\begin{aligned} (f\ u)^* &\equiv f^*\ u[0]\ u[1]\ u^* \\ (\lambda x.t)^* &\equiv \lambda x_0, x_1, x_2\ .\ t^* \\ x^* &\equiv x_2 \\ \mathsf{U}^* &\equiv \sim_{\mathsf{U}} \end{aligned}$$

Homogeneous equality

- Heterogeneous equality:

$$\frac{\Gamma \vdash A : \mathsf{U}}{\Gamma^{=} \vdash \sim_A : A[0] \to A[1] \to \mathsf{U}}$$

- We need equality in the same context:

$$\frac{\Gamma \vdash A : \mathsf{U}}{\Gamma \vdash =_A : A \to A \to \mathsf{U}}$$

- Therefore we define a substitution $R_\Gamma : \Gamma \Rightarrow \Gamma^{=}$:

$$\begin{aligned} \mathsf{R}_\emptyset &\equiv () && : \emptyset \Rightarrow \emptyset \\ \mathsf{R}_{\Gamma.x:A} &\equiv (\mathsf{R}_\Gamma, x, x, \mathsf{refl}\, x) && : (\Gamma.x : A) \Rightarrow (\Gamma.x : A)^{=} \end{aligned}$$

$$\begin{tikzcd} \Gamma^{=} \arrow[d, bend right=60, "1_\Gamma"'] \arrow[d, bend left=60, "0_\Gamma"] \\ \Gamma \arrow[u, "\mathsf{R}_\Gamma"'] \end{tikzcd}$$

$$\frac{\Gamma \vdash a : A}{\Gamma \vdash \mathsf{refl}\, a \equiv (a^*)[\mathsf{R}_\Gamma] : \underbrace{a \sim_A [\mathsf{R}_\Gamma]\, a}_{a =_A a}}$$

- $\mathsf{refl}\, x$ is a new normal form if x is a variable.

What is $(\mathsf{refl}\,x)^*$? (i)

Maybe we could define it just as $\mathsf{refl}\,x^*$.

$$
\begin{aligned}
(x : A)^= \vdash (\mathsf{refl}\,x)^* &: \mathsf{refl}\,x_0 \sim_{(x \sim_{\mathsf{refl}\,A} x)^*} \mathsf{refl}\,x_1 \\
&\equiv \sim_{((\sim_{A^*[\mathsf{R}]})^*\,x_0\,x_1\,x_2\,x_0\,x_1\,x_2)} (\mathsf{refl}\,x_0)\,(\mathsf{refl}\,x_1) \\
(x : A)^= \vdash \mathsf{refl}\,x^* &: x_2 \sim_{\mathsf{refl}\,(x_0 \sim_{A^*} x_1)} x_2 \\
&\equiv \sim_{((\sim_{A^*[\mathsf{R}]})^*\,x_0\,x_0\,(\mathsf{refl}\,x_0)\,x_1\,x_1\,(\mathsf{refl}\,x_1))} x_2\,x_2
\end{aligned}
$$

Higher dimensions

By iterating $-^{=}$, we get higher dimensional cubes. Eg. if $A : \mathrm{U}$, elements of $x : A$, $(x : A)^{=}$, $((x : A)^{=})^{=}$, $(x : A)^{3}$ look like this:

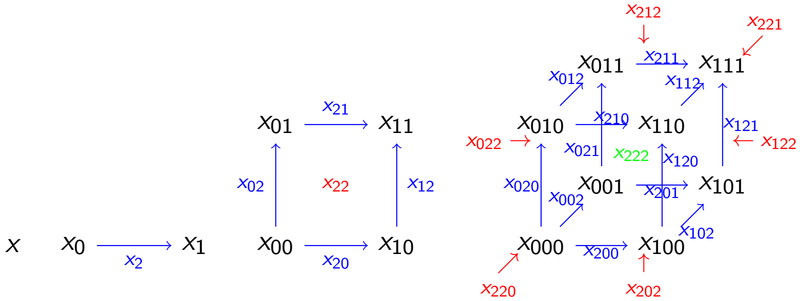

Figure : Cubes of dimension 0-3.

What is $(\mathsf{refl}\, x)^*$? (ii)

$$
\begin{aligned}
(x : A)^= \vdash (\mathsf{refl}\, x)^* &: \mathsf{refl}\, x_0 \sim_{(x \sim_{\mathsf{refl}\, A} x)^*} \mathsf{refl}\, x_1 \\
&\equiv \sim_{((\sim_{A^*[\mathsf{R}]})^*\, x_0\, x_1\, x_2\, x_0\, x_1\, x_2)} (\mathsf{refl}\, x_0)\, (\mathsf{refl}\, x_1) \\
(x : A)^= \vdash \mathsf{refl}\, x^* &: x_2 \sim_{\mathsf{refl}\, (x_0 \sim_{A^*} x_1)} x_2 \\
&\equiv \sim_{((\sim_{A^*[\mathsf{R}]})^*\, x_0\, x_0\, (\mathsf{refl}\, x_0)\, x_1\, x_1\, (\mathsf{refl}\, x_1))} x_2\, x_2
\end{aligned}
$$

$$
\begin{tikzcd}
x_1 \arrow[r, "\mathsf{refl}\, x_1"] & x_1 \\
x_0 \arrow[u, "x_2"] \arrow[r, "\mathsf{refl}\, x_0"] \arrow[ur, phantom, "(\mathsf{refl}\, x)^*"] & x_0 \arrow[u, "x_2"']
\end{tikzcd}
\qquad
\begin{tikzcd}
x_0 \arrow[r, "x_2"] & x_1 \\
x_0 \arrow[u, "\mathsf{refl}\, x_0"] \arrow[r, "x_2"] \arrow[ur, phantom, "\mathsf{refl}\, x^*"] & x_1 \arrow[u, "\mathsf{refl}\, x_1"']
\end{tikzcd}
$$

If we swap the vertical and horizontal dimensions we get one from the other.

Swap

We define a substitution $\mathsf{S}_\Gamma : \Gamma^2 \Rightarrow \Gamma^2$.

Visually:

$$
\begin{tikzcd}
x_{01} \arrow[r, "x_{21}"] & x_{11} & & x_{10} \arrow[r, "x_{12}"] & x_{11} \\
x_{00} \arrow[u, "x_{02}"] \arrow[r, "x_{20}"'] \arrow[ur, phantom, "x_{22}"] & x_{10} \arrow[u, "x_{12}"'] \arrow[r, phantom, "\overset{\mathsf{S}_{x:A}}{\longmapsto}"] & & x_{00} \arrow[u, "x_{20}"] \arrow[r, "x_{02}"'] \arrow[ur, phantom, "x_{22}[\mathsf{S}_{x:A}]"] & x_{01} \arrow[u, "x_{21}"']
\end{tikzcd}
$$

Now we can say that

$$(\mathsf{refl}\, x)^* \equiv (x_2[\mathsf{R}])^* \equiv x_{22}[(\mathsf{R}_{x:A})^=] \equiv x_{22}[\mathsf{S}_{x:A}\mathsf{R}_{x:A}]$$

The last element $x_{22}[\mathsf{S}_{x:A}\mathsf{R}_{x:A}]$ is a new normal form like $x_2[\mathsf{R}_{x:A}] \equiv \mathsf{refl}\, x$.
But now we can do $(\mathsf{S}_{x:A})^=$ and $((\mathsf{S}_{x:A})^=)^=$ etc.

The full picture

The iterated version of $-^{=}$ makes any context into a presheaf over the base category of cubical sets. A context Γ is a presheaf $\mathcal{C} \to \mathsf{Con}$.

$$
\begin{tikzcd}
\vdots \arrow[loop right, "(x \mapsto y, y \mapsto x)"] \\
\{x, y\} \arrow[d, bend right=80, swap, "(x \mapsto 1, y \mapsto x)"] \arrow[d, bend right=40, "(x \mapsto x, y \mapsto 1)"] \arrow[d, bend left=40, swap, "(x \mapsto x, y \mapsto 0)"] \arrow[d, bend left=80, "(x \mapsto 0, y \mapsto x)"] \\
\{x\} \arrow[u, shift left, "(x \mapsto y)"] \arrow[u, shift right, swap, "(x \mapsto x)"] \arrow[d, bend right=50, swap, "x \mapsto 1"] \arrow[d, bend left=50, "x \mapsto 0"] \\
\{\} \arrow[u, "()"]
\end{tikzcd}
\qquad
\begin{tikzcd}
\vdots \arrow[loop right, "\mathsf{S}_\Gamma"] \\
\Gamma^2 \arrow[d, bend right=80, swap, "(1_\Gamma)^{=}"] \arrow[d, bend right=40, "1_{\Gamma^=}"] \arrow[d, bend left=40, swap, "0_{\Gamma^=}"] \arrow[d, bend left=80, "(0_\Gamma)^{=}"] \\
\Gamma^{=} \arrow[u, shift left, "(\mathsf{R}_\Gamma)^{=}"] \arrow[u, shift right, swap, "\mathsf{R}_{\Gamma^=}"] \arrow[d, bend right=50, swap, "1_\Gamma"] \arrow[d, bend left=50, "0_\Gamma"] \\
\Gamma \arrow[u, "\mathsf{R}_\Gamma"]
\end{tikzcd}
$$

The new normal forms:

$$\Gamma^{n+2-k} \vdash x_{2\ldots 2}[\mathsf{S}^{n-2-i_1}_{\Gamma^{i_1}} \ldots \mathsf{S}^{n-2-i_m}_{\Gamma^{i_m}} \mathsf{R}_{\Gamma^{n+1}} \ldots \mathsf{R}_{\Gamma^{n+2-k}}]$$

Substitution rule for variables with extra structure

- Normal form:

$$\Gamma^{n+2-k} \vdash x_{2^n}[\mathsf{S}^{n-2-i_1}_{\Gamma^{i_1}} \dots \mathsf{S}^{n-2-i_m}_{\Gamma^{i_m}} \mathsf{R}_{\Gamma^{n+1}} \dots \mathsf{R}_{\Gamma^{n+2-k}}]$$

- Given $\rho : \Delta \Rightarrow (x : A)^{n+2-k}$, we can commute it with degeneracies:

$$\begin{aligned}
&\Delta \vdash x_{2^n}[\mathsf{S}^{n-2-i_1}_{\Gamma^{i_1}} \dots \mathsf{S}^{n-2-i_m}_{\Gamma^{i_m}} \mathsf{R}_{\Gamma^{n+1}} \dots \mathsf{R}_{\Gamma^{n+2-k}} \rho] \\
&\Delta \vdash x_{2^n}[\mathsf{S}^{n-2-i_1}_{\Gamma^{i_1}} \dots \mathsf{S}^{n-2-i_m}_{\Gamma^{i_m}} \mathsf{R}_{\Gamma^{n+1}} \dots \rho^{=} \mathsf{R}_{\Delta}] \\
&\dots \\
&\Delta \vdash x_{2^n}[\mathsf{S}^{n-2-i_1}_{\Gamma^{i_1}} \dots \mathsf{S}^{n-2-i_m}_{\Gamma^{i_m}} \rho^{k} \mathsf{R}_{\Delta^{k-1}} \dots \mathsf{R}_{\Delta}]
\end{aligned}$$

  using the rule $\mathsf{R}\rho \equiv \rho^{=}\mathsf{R}$.
- We need to commute swaps and substitutions.

Commute swaps and substitutions

The 2-dimensional case:

$$(\Gamma.x : A)^2 \vdash x_{22}[\mathsf{S}_{\Gamma.x:A}]$$

If A is a nondependent Π type we can explain it in terms of smaller things:

$$(A \to B)^{**} \equiv (x : A)^2 \to B^{**}$$

We can rewrite a swapped variable of that type:

$$\begin{aligned}(\Gamma.f : A \to B)^2 \vdash\ & f_{22}[\mathsf{S}_{(\Gamma.f:A\to B)}] \\ \equiv\ & \lambda(x : A)^2[\mathsf{S}_\Gamma].f_{22}[\mathsf{S}_{(\Gamma.f:A\to B)}]\,(x)^2 \\ \equiv\ & \lambda(x : A)^2[\mathsf{S}_\Gamma].(f_{22}\,(x^2)[\mathsf{S}_{\Gamma.x:A}])[\mathsf{S}_{\Gamma.y:B}]\end{aligned}$$

For base types like $\mathbb{N}$ we have:

$$x_{22}[\mathsf{S}_{\mathbb{N}}\rho] \equiv x_{22}[\rho]$$

New definition of $\sim_U$

$$A \sim_U B \equiv A \to B \to U$$

is replaced by

$$\begin{aligned}
A \sim_U B \equiv \; \Sigma - \sim - \;&: A \to B \to U \\
\overrightarrow{\text{coe}} \;&: A \to B \\
\overrightarrow{\text{coh}} \;&: \Pi(x : A).x \sim \overrightarrow{\text{coe}}\, x \\
\text{un}\overrightarrow{\text{coe}} \;&: \Pi(x : A, y : B, p : x \sim y).\overrightarrow{\text{coe}}\, x =_B y \\
\text{un}\overrightarrow{\text{coh}} \;&: \Pi(x : A, y : B, p : x \sim y).\overrightarrow{\text{coh}}\, x \sim_{(x \sim z)^*[\mathsf{R}_{x:A}\rho]} p \\
\overleftarrow{\text{coe}} \;&: B \to A \;\ldots
\end{aligned}$$

which is equivalent to

$$\begin{aligned}
\Sigma\,(- \sim - : A \to B \to U).\Pi(x : A).\text{isContr}(\Sigma(y : B).x \sim y) \\
\times\; \Pi(y : B).\text{isContr}(\Sigma(x : A).x \sim y)
\end{aligned}$$

Coerce and coherence for the universe

Now given $A : \mathsf{U}$, A^* will have the following components:

- $\sim_{A^*}$: the relation we defined previously
- $\mathsf{coe}_{A^*} : A[0] \to A[1]$
- $\mathsf{coh}_{A^*} : Pi(x : A).x \sim \overrightarrow{\mathsf{coe}}\, x$
- ...

If $A \equiv \mathsf{U}$, we need $\mathsf{coe}_{\mathsf{U}^*}$, $\mathsf{coh}_{\mathsf{U}^*}$, ...:

$$
\begin{aligned}
\mathsf{coe}_{\mathsf{U}^*}\, A_0 \equiv & A_0 \\
\mathsf{coh}_{\mathsf{U}^*}\, A_0 \equiv & (\sim_{\mathsf{refl}\, A_0} \\
& , \lambda x_0 . x_0 \\
& , \lambda x_0 . \mathsf{refl}\, x_0 \\
& , \lambda x_0\, x_1\, x_2 . \mathsf{uncoe}_{A^*[\mathsf{R}]}\, x_0\, x_1\, x_2 \\
& , \lambda x_0\, x_1\, x_2 . \mathsf{uncoh}_{A^*[\mathsf{R}]}\, x_0\, x_1\, x_2)
\end{aligned}
$$

Coerce for Π

We have an $\Gamma^{=} \vdash f : (\Pi(x : A).B)[0]$, now $\overrightarrow{\mathrm{coe}}\, f : (\Pi(x : A).B)[1]$.

$$f\,(\overleftarrow{\mathrm{coe}}_A\, x_1) \xrightarrow{\overrightarrow{\mathrm{coh}}_B\,(f\,(\overleftarrow{\mathrm{coe}}_A\, x_1))} \overrightarrow{\mathrm{coe}}_B\,(f\,(\overleftarrow{\mathrm{coe}}_A\, x_1)) \quad : \; B[0] \xrightarrow{\sim_B} B[1]$$

$$\overleftarrow{\mathrm{coe}}_A\, x_1 \xrightarrow[\overleftarrow{\mathrm{coh}}_A\, x_1]{} x_1 \quad : \; A[0] \xrightarrow[\sim_A]{} A[1]$$

Coherence for Π

We need $\mathsf{coh}\, f : \Pi(x : A)^{=}. f\, x_0 \sim_B \overrightarrow{\mathsf{coe}}_B\, (f\, (\overleftarrow{\mathsf{coe}}_A\, x_1))$.

$$
\begin{tikzcd}
f\, x_0 \arrow[rr, dashed] & & \overrightarrow{\mathsf{coe}}_B\, (f\, (\overleftarrow{\mathsf{coe}}_A\, x_1)) \\
& \Uparrow & \\
f\, (\overleftarrow{\mathsf{coe}}_A\, x_1) \arrow[uu, "f\, r"] \arrow[rr, "\overrightarrow{\mathsf{coh}}_B"] & & \overrightarrow{\mathsf{coe}}_B\, (f\, (\overleftarrow{\mathsf{coe}}_A\, x_1)) \arrow[uu, "\mathsf{refl}"']
\end{tikzcd}
\quad : \quad
\begin{tikzcd}
B[0] \arrow[r, "\sim_B"] & B[1] \\
B[0] \arrow[u, "=_{B[0]}"] \arrow[r, "\sim_B"'] & B[1] \arrow[u, "=_{B[1]}"']
\end{tikzcd}
$$

$$
\begin{tikzcd}
x_0 \arrow[rr, "x_2"] & & x_1 \\
& \Leftarrow & \\
\overleftarrow{\mathsf{coe}}_A\, x_1 \arrow[uu, dashed, "r"] \arrow[rr, "\overleftarrow{\mathsf{coh}}_A\, x_1"'] & & x_1 \arrow[uu, "\mathsf{refl}\, x_1"']
\end{tikzcd}
\quad : \quad
\begin{tikzcd}
A[0] \arrow[r, "\sim_A"] & A[1] \\
A[0] \arrow[u, "=_{A[0]}"] \arrow[r, "\sim_A"'] & A[1] \arrow[u, "=_{A[1]}"']
\end{tikzcd}
$$

How do we get higher Kan operations?

Some of them are just first-level Kan operations for higher types.

$$\begin{tikzcd}
x_{01} & x_{11} \\
x_{00} \arrow[u, "x_{02}"] \arrow[r, "x_{20}"'] & x_{10} \arrow[u, "x_{12}"']
\end{tikzcd}$$

We have

$$\Gamma^{=}.x_0 : A[0].x_1 : A[1] \vdash x_0 \sim_A x_1 : \mathsf{U},$$

so

$$(\Gamma^{=}.x_0 : A[0].x_1 : A[1])^{=} \vdash (x_0 \sim_A x_1)^* : (x_{00} \sim_A [0]\, x_{10}) \sim_{\mathsf{U}} (x_{01} \sim_A [1]\, x_{11}).$$

Identity type

Non-dependent eliminator:

$$\frac{P : A \to \mathsf{U} \quad r : x =_A y \quad u : P\,x}{\mathsf{transport}_P\, r\, u : P\, y}$$

We have that P respects equality:

$$\frac{P : A \to \mathsf{U}}{P^*[\mathsf{R}_\Gamma] : \Pi(x_0, x_1 : A, x_2 : x_0 = x_1).P\,x_0 \sim_{\mathsf{U}} P\,x_1}$$

And we define transport by using $P^*[\mathsf{R}]$:

$$\frac{P : A \to \mathsf{U} \quad r : x =_A y \quad u : P\,x}{\mathsf{transport}_P\, r\, u \equiv \overrightarrow{\mathsf{coe}}_{(P^*[\mathsf{R}_\Gamma]\,x\,y\,r)}\, u : P\, y}$$

We can validate the dependent eliminator by also proving that singletons are contractible.

Conclusion

- A different presentation of internal parametricity showing the connections with the cubical set model.
- Changing parametricity for the universe from relation space to equivalence.
- This forces us to define the first level Kan operations for each type.
- Higher Kan operations are first level Kan operations for higher types.
- Not shown here:
  - Uniqueness conditions, how to lift them through type formers.
- Unfinished work:
  - An implementation in Haskell
  - Swapping the universe
  - How to do higher inductive types
  - Definitional computation rule for the identity type: making $-^{=}$ a monad

Need for internal parametricity

The basic example for parametricity is the polymorphic identity function: we would like to prove that any given function f of type $\Pi(A : \mathsf{U}).A \to A$ is the identity function. Parametricity for f (denoted by t) says that f maps related arguments to related results:

$$\begin{aligned} f : \Pi(A : \mathsf{U}).A \to A \vdash t : \, & \Pi(A_0, A_1 : \mathsf{U}, A_2 : A_0 \to A_1 \to \mathsf{U})\,. \\ & \Pi(x_0 : A_0, x_1 : A_1, x_2 : A_2\, x_0\, x_1) \\ & \quad .\, A_2\,(f\, A_0\, x_0)\,(f\, A_1\, x_1), \end{aligned}$$

and then using t and a relation A_2 which relates anything to c we can do:

$$\begin{aligned} f : \Pi(A : \mathsf{U}).A \to A \vdash & \lambda A\, c\,.\, t\, A\, A\,(\lambda x\, \_\,.\, x = c)\, c\, c\,(\mathsf{refl}\, c) \\ & : \Pi(A : \mathsf{U}, c : A)\,.\, f\, A\, c = c. \end{aligned}$$

f^* would be a good candidate for t, however it doesn't live in the desired context but in the $\_^{=}$-d context.

Identity type: singletons are contractible

We also show that singletons are contractible i.e. we show how to construct the terms s and t of the following type:

$$
\frac{\Gamma \vdash a, b : A \quad \Gamma \vdash r : a =_A b}{\begin{aligned} \Gamma \vdash (s, t) : \quad & (a, \mathsf{refl}\, a) =_{\Sigma(x:A).a =_A x} (b, r) \\ \equiv \quad & \Sigma(s : a \sim_A [\mathsf{R}_\Gamma]\, b).\mathsf{refl}\, a \sim_{a \sim_A [\mathsf{R}_\Gamma] x} [\mathsf{R}_\Gamma, a, b, s]\, r \end{aligned}}
$$

s is constructed by filling the following incomplete square from bottom to top:

$$
\begin{tikzcd}
a \arrow[r, dashed, "s"] & b \\
a \arrow[u, "\mathsf{refl}\, a"] \arrow[r, "\mathsf{refl}\, a"'] & a \arrow[u, "r"']
\end{tikzcd}
$$