A syntax for cubical type theory

Ambrus Kaposi

PhD student
University of Nottingham

(joint work with Thorsten Altenkirch)

University of Bath
November 4, 2014

Goal (i)

- Type Theory is a foundation of mathematics and a programming language.
  - It is based on the Curry-Howard correspondance:

$$\text{proof} : \text{proposition} = \text{program} : \text{type}$$

  - Examples:

$$-+- \; : \mathbb{N} \to \mathbb{N} \to \mathbb{N}$$

$$\text{comm} : \Pi(x, y : \mathbb{N}).x + y =_{\mathbb{N}} y + x$$

$$\mathbb{N} \quad : \mathsf{U}$$

- Homotopy Type Theory adds the univalence axiom to Type Theory:

$$\text{univ} : (A \simeq B) \to (A =_{\mathsf{U}} B)$$

The theory becomes extensional. Examples:

$$\lambda x.x + \text{zero} =_{\mathbb{N}\to\mathbb{N}} \lambda x.\text{zero} + x$$

$$\mathbb{N} \times \text{Bool} =_{\mathsf{U}} \text{Bool} \times \mathbb{N}$$

$$\pi_1(\mathbb{S}^1) =_{\Sigma(A:\mathsf{U}).\text{isGroup}(A)} \mathbb{Z}$$

$$\text{Bool} =_{\mathsf{U}} \text{Bool}$$

Goal (ii)

- 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.:

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
	$(\mathsf{suc}\, m =_{\mathbb{N}} \mathsf{suc}\, n)$	$\simeq$	$(m =_{\mathbb{N}} n)$
pairs:	$((a, b) =_{A \times B} (a', b'))$	$\simeq$	$(a =_A a' \times b =_B b')$
functions:	$(f =_{A \to B} g)$	$\simeq$	$(\Pi(x : A).f\, x =_B g\, x)$
types:	$(A =_{\mathsf{U}} B)$	$\simeq$	$(A \simeq B)$

- 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:

- 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:

Where do the cubes come from? (i)

- 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:

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

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

- But now we need to talk about equality of equalities:

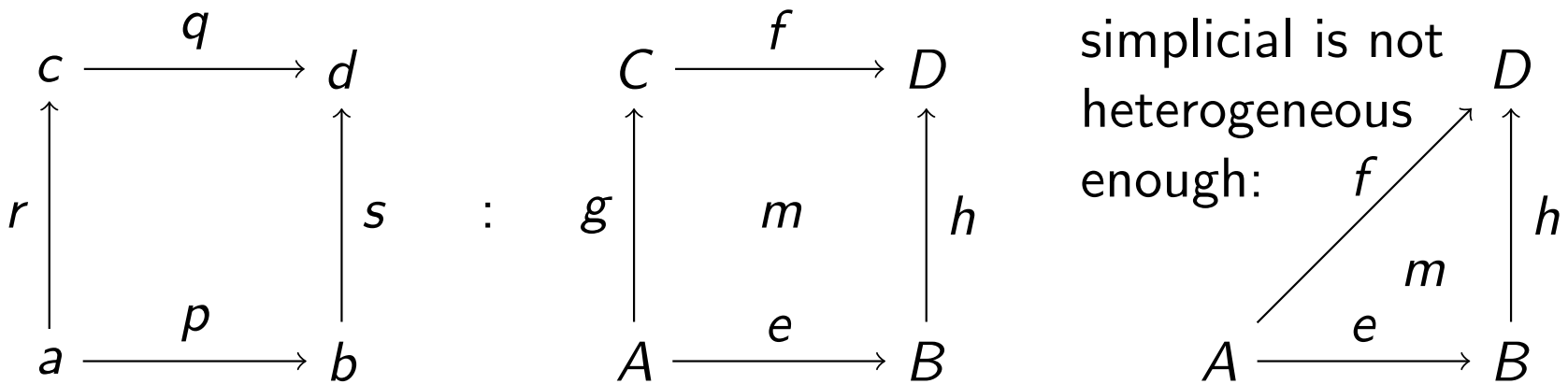
$$\frac{p : a \sim_e b \quad q : c \sim_f d}{p \sim_? q : \mathsf{U}}$$

Where do the cubes come from? (ii)

- ...now we need to talk about equality of equalities:

$$\frac{p : a \sim_e b \quad q : c \sim_f d}{p \sim_? q : \mathsf{U}}$$

- Picture:



- So we have:

$$\frac{e : A =_{\mathsf{U}} B \quad f : C =_{\mathsf{U}} D \quad g : A =_{\mathsf{U}} C \quad h : B =_{\mathsf{U}} D \quad m : e \sim_{g \sim_{\mathsf{U}} h} f \quad p : a \sim_e b \quad q : c \sim_f d \quad r : a \sim_g c \quad s : b \sim_h d}{p \sim_{g \sim_m h} q : \mathsf{U}}$$

The functor $-^=$

- We define the endofunctor $-^=$ on the category of contexts and substitutions.
  - Action on contexts:

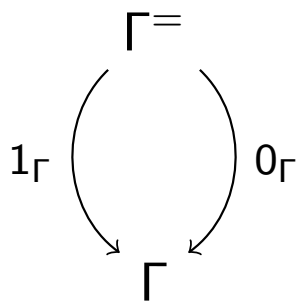
$$\begin{aligned} \emptyset^= &\equiv \emptyset \\ (\Gamma, x : A)^= &\equiv \Gamma^=, x_0 : A[0_\Gamma], x_1 : A[1_\Gamma], x_2 : x_0 \sim_A x_1 \end{aligned}$$

  - Action on substitutions:

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

  - 0, 1 substitutions project out the corresponding components:

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



  - $\sim_A$ is the heterogeneous equality for A
  - t^* says that t respects this equality

Heterogeneous equality

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

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

$f_0 \sim_{\Pi(x:A).B} f_1 \equiv \Pi(x_0 : A[0], x_1 : A[1], x_2 : x_0 \sim_A x_1).f_0\, x_0 \sim_B f_1\, x_1$

parametricity: $A \sim_{\mathsf{U}} B \equiv A \to B \to \mathsf{U}$

HoTT (later): $A \sim_{\mathsf{U}} B \equiv A \simeq B$

- 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:

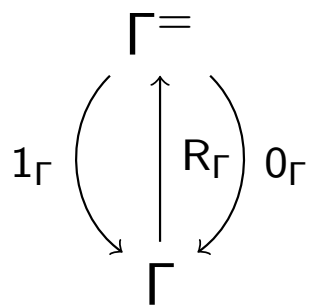
$$\frac{\Gamma \vdash A : \mathsf{U}}{\Gamma^{=} \vdash {\sim_A} : A[0] \to A[1] \to \mathsf{U}}$$

- 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 $R_\Gamma : \Gamma \Rightarrow \Gamma^{=}$:

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



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

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

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

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

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

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:

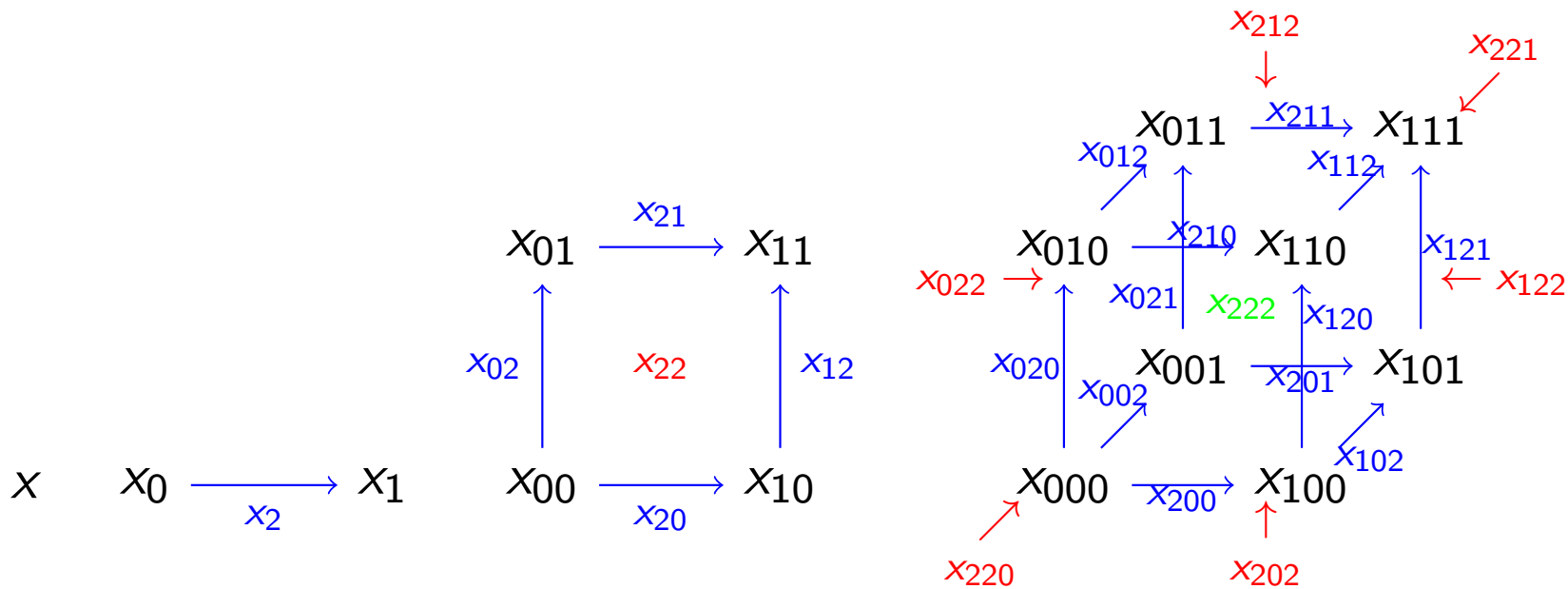
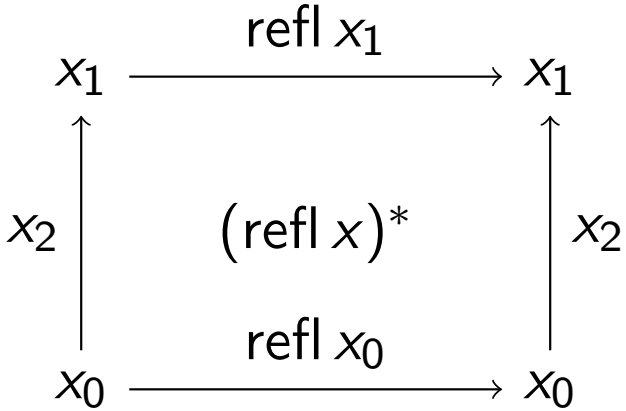
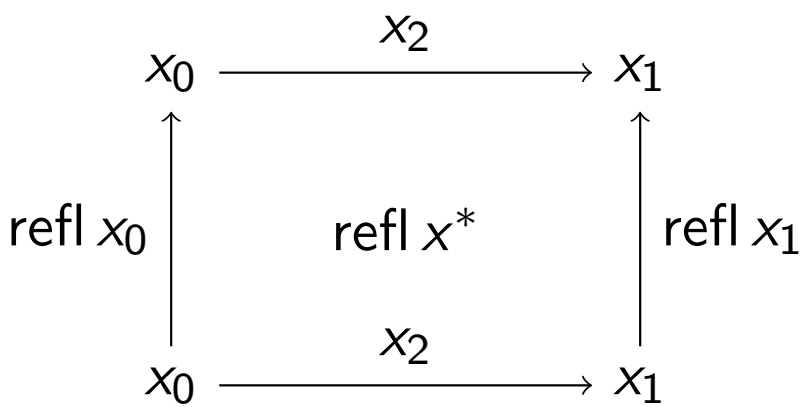


Figure : Cubes of dimension 0-3.

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

$$(x : A)^= \vdash (\mathsf{refl}\, x)^* : \mathsf{refl}\, x_0 \sim_{(x \sim_{\mathsf{refl}\, A\, x)^*}} \mathsf{refl}\, x_1$$
$$\equiv \sim_{((\sim_{A^*})^*\, x_0\, x_1\, x_2\, x_0\, x_1\, x_2)}\, (\mathsf{refl}\, x_0)\, (\mathsf{refl}\, x_1)$$
$$(x : A)^= \vdash \mathsf{refl}\, x^* : x_2 \sim_{\mathsf{refl}\, (x_0 \sim_{A^*} x_1)} x_2$$
$$\equiv \sim_{((\sim_{A^*})^*\, x_0\, x_0\, (\mathsf{refl}\, x_0)\, x_1\, x_1\, (\mathsf{refl}\, x_1))}\, x_2\, x_2$$

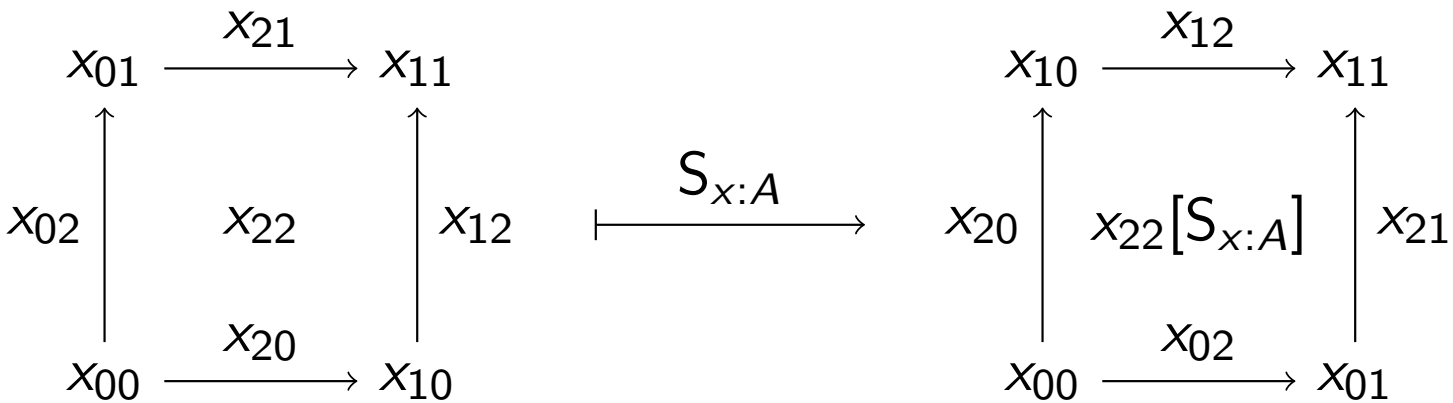




If we swap the vertical and horizontal dimensions we get one from the other.

Swap

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



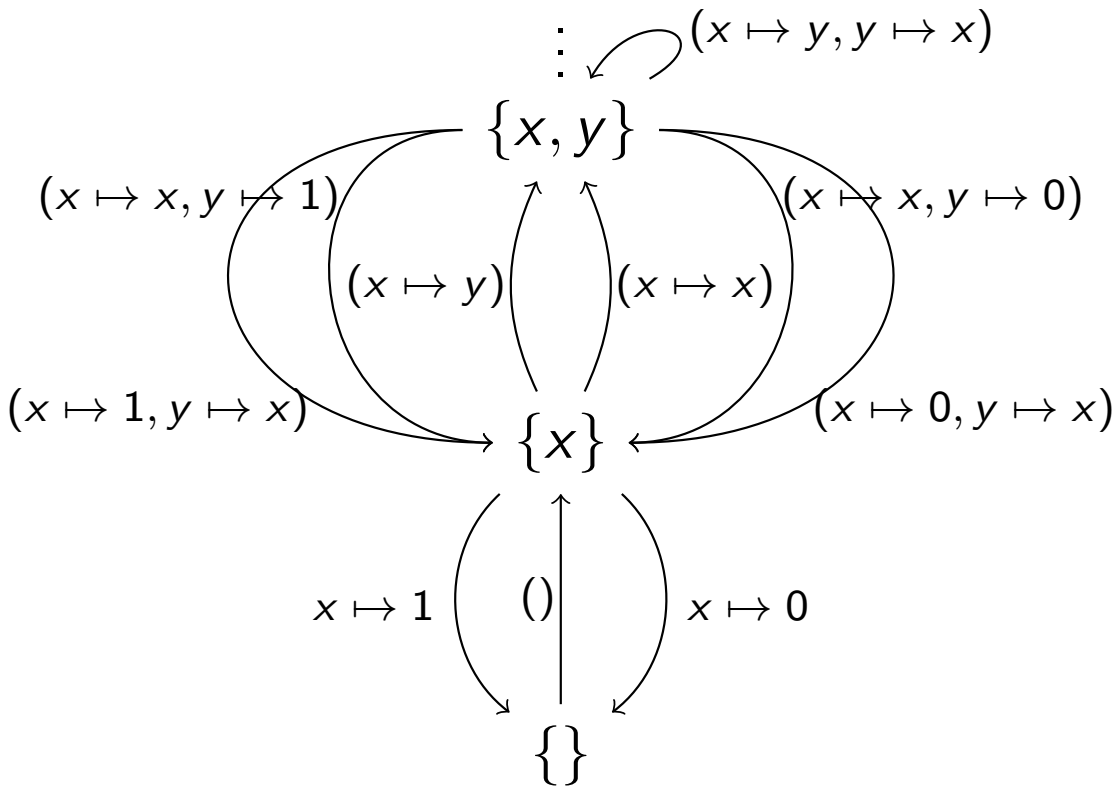
Now we can say that

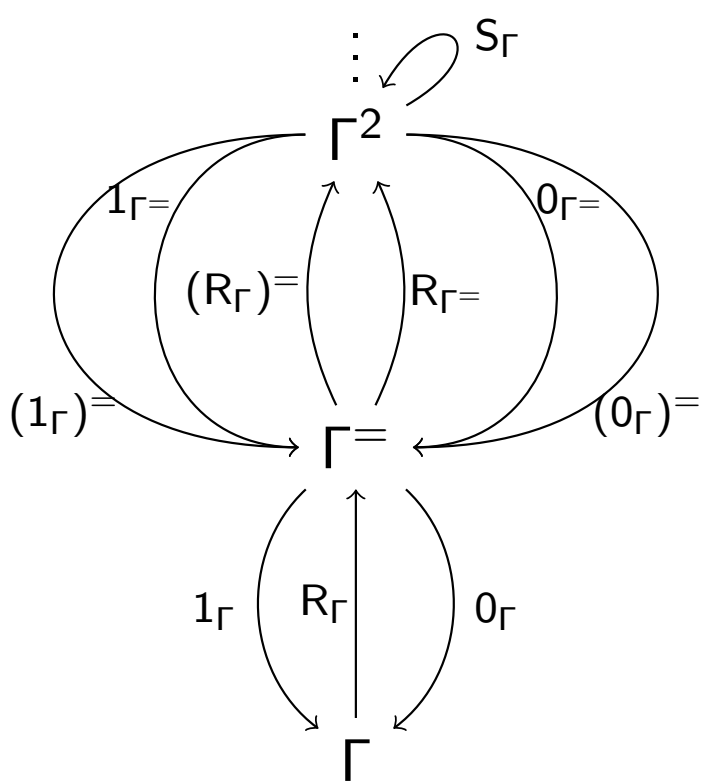
$$(\mathsf{refl}\, x)^* \equiv (x_2[\mathsf{R}])^* \equiv x_{22}[(\mathsf{R}_{x:A})^=] \equiv x_{22}[\mathsf{S}_{x:A}\mathsf{R}_{x:A}]$$

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 $\mathcal{C} \to \mathsf{Con}$.





This gives us a theory with internal parametricity, the new normal forms:

$$\Gamma^{n+2-k} \vdash x_{2\ldots2}[\mathsf{S}^{n-2-i_1}_{\Gamma^{i_1}} \mathsf{S}^{n-2-i_2}_{\Gamma^{i_2}} \ldots \mathsf{S}^{n-2-i_m}_{\Gamma^{i_m}} \mathsf{R}_{\Gamma^{n+1}} \mathsf{R}_{\Gamma^n} \ldots \mathsf{R}_{\Gamma^{n+2-k}}]$$

New definition of $\sim_{\mathsf{U}}$

Our previous definition:

$$A \sim_{\mathsf{U}} B \equiv A \to B \to \mathsf{U}$$

is replaced with

$$\begin{aligned} A \sim_{\mathsf{U}} B \equiv\ & \Sigma - \sim - : A \to B \to \mathsf{U} \\ & \overrightarrow{\mathrm{coe}} \quad : A \to B \\ & \overrightarrow{\mathrm{coh}} \quad : \Pi(x : A).x \sim \mathrm{coe}^0\, x \\ & \overrightarrow{\mathrm{uni}} \quad : \ldots \\ & \overleftarrow{\mathrm{coe}} \quad : B \to A \\ & \ldots \end{aligned}$$

which is equivalent to $A \simeq B$.

Now we need to provide not only $\sim_A$, but also coe_A, coh_A etc. for each type former. The latter correspond to first level Kan operations.

Coerce for Π

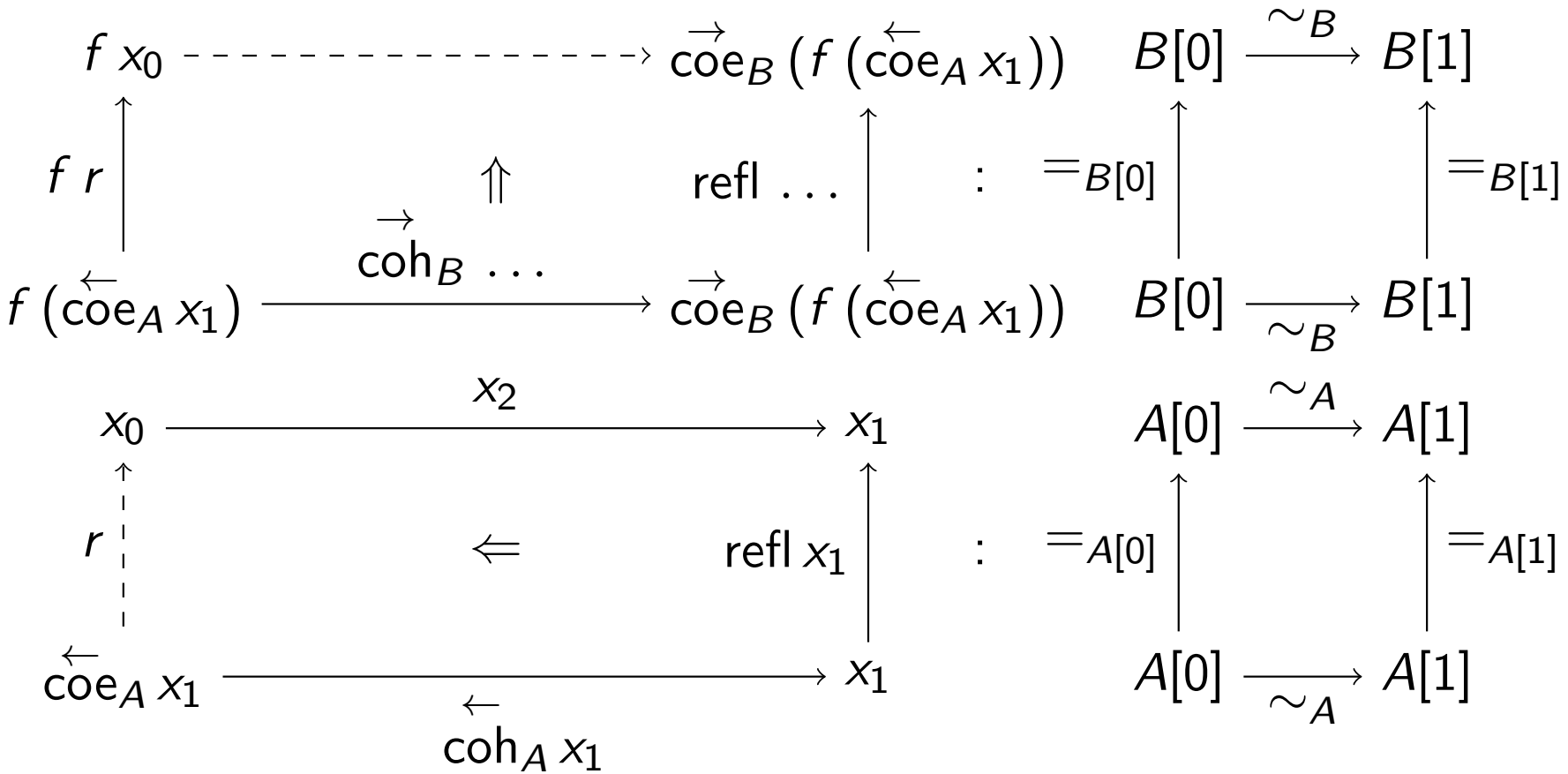
We have an $\Gamma^{=} \vdash f : (\Pi(x : A).B)[0]$, now $\overrightarrow{\mathrm{coe}}\, f : (\Pi(x : A).B)[1]$.

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

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

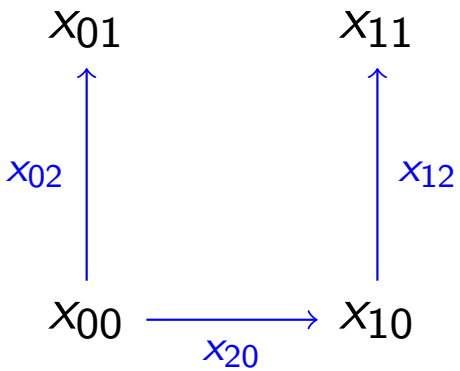
Coherence for Π

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



How do we get higher Kan operations?

These are just first-level Kan operations for higher types.



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:

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

We have that P respects equality:

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

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

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

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:
  - How to commute swaps and arbitrary substitutions (needed already for internal parametricity).
  - The universe is Kan (what is $\mathsf{coe}_{\mathsf{U}}$, $\mathsf{coh}_{\mathsf{U}}$).
  - Uniqueness conditions, how to lift them through type formers.
- Unfinished work:
  - An implementation in Haskell (parametricity part nearly done)
  - Swapping the universe
  - How to do higher inductive types
  - Definitional computation rule for the identity type

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.

All swaps

- In general, $S^j_{\Gamma i}$ swaps the i^{th} and $(i+1)^{\text{th}}$ dimensions.
- Normal forms ($x \in \Gamma$):

$$\Gamma^{n+2-k} \vdash x_{2^n}[S^{n-2-i_1}_{\Gamma i_1} S^{n-2-i_2}_{\Gamma i_2} \ldots S^{n-2-i_m}_{\Gamma i_m} R_{\Gamma^{n+1}} R_{\Gamma^n} \ldots R_{\Gamma^{n+2-k}}]$$

  where $n \geq 2$, $m \geq 0$, $k \geq 0$.
- This is the filler of an n-dimensional cube where the last k dimensions are degenerated, and then the given list of swaps are performed.
- What happens if we substitute with $\rho : \Delta \Rightarrow (x : A)^{n+2-k}$?

$$\Delta \vdash x_{2^n}[S^{n-2-i_1}_{\Gamma i_1} S^{n-2-i_2}_{\Gamma i_2} \ldots S^{n-2-i_m}_{\Gamma i_m} R_{\Gamma^{n+1}} R_{\Gamma^n} \ldots R_{\Gamma^{n+2-k}} \rho]$$

$$\Delta \vdash x_{2^n}[S^{n-2-i_1}_{\Gamma i_1} S^{n-2-i_2}_{\Gamma i_2} \ldots S^{n-2-i_m}_{\Gamma i_m} R_{\Gamma^{n+1}} R_{\Gamma^n} \ldots \rho^{=} R_\Delta]$$

$$\ldots$$

$$\Delta \vdash x_{2^n}[S^{n-2-i_1}_{\Gamma i_1} S^{n-2-i_2}_{\Gamma i_2} \ldots S^{n-2-i_m}_{\Gamma i_m} \rho^k R_{\Delta^{k-1}} R_{\Delta^{k-2}} \ldots R_\Delta]$$

- But how to commute swaps and general substitutions?

Commute swaps and subsitutions

$$\frac{(\Gamma.x : A)^n \vdash x_{2^n} : A^{*n}}{(\Gamma.x : A)^n \vdash x_{2^n}[\mathsf{S}^i_{(\Gamma.x:A)^j}] : A^{*n}[\mathsf{S}^i_{(\Gamma.x:A)^j}]}$$

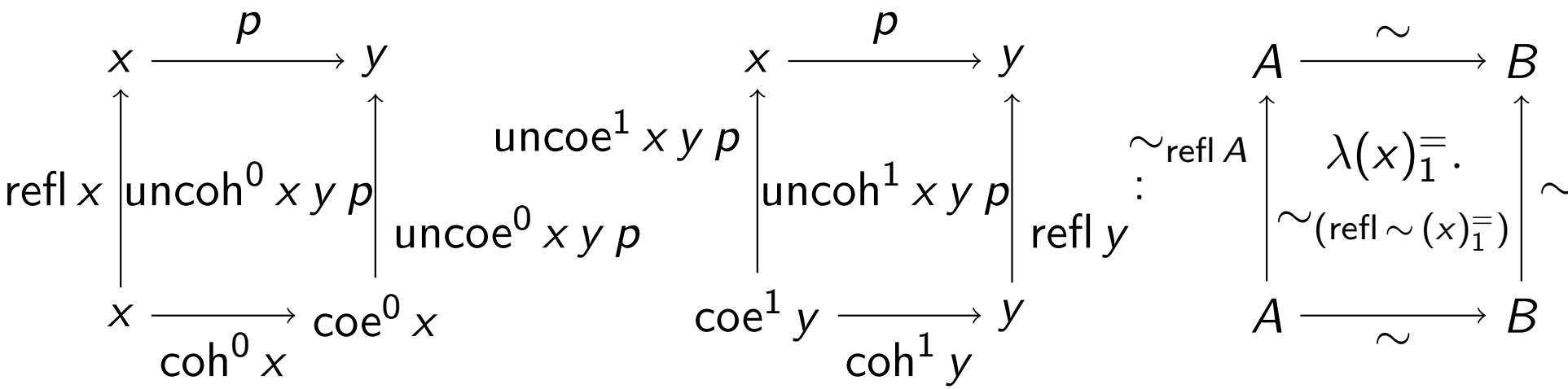
If A is a $\Pi/\Sigma/$variable type we can explain it in terms of smaller things:

$$(f : \Pi(x : A).B)^{*n} \equiv \Pi(x : A)^n.B^{*n}[(f\ x)_n]$$

We can rewrite a swapped variable of that type:

$$\begin{aligned}
(\Gamma.f : \Pi(x : A).B)^n \vdash \quad & f_{2^n}[\mathsf{S}^i_{(\Gamma.f)^j}] \\
(\eta\text{-expansion}) \qquad & \equiv \lambda(x : A)^n[\mathsf{S}^i_{\Gamma^j}].f_{2^n}[\mathsf{S}^i_{(\Gamma.f)^j}]\,(x)^n[\mathsf{S}^i_{\Gamma^j}] \\
(\text{new rule}) \qquad & \equiv \lambda(x : A)^n[\mathsf{S}^i_{\Gamma^j}].f_{2^n}\,(x)^n[\mathsf{S}^i_{(\Gamma.x:A)^j}\mathsf{S}^i_{\Gamma^j}]
\end{aligned}$$

Equivalence



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:

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

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