Polynomial Functors and Natural Models of Type Theory

Steve Awodey

Workshop on Polynomial Functors
Topos Institute
March 2021

Outline

1. Dependent type theory
2. Natural models
3. Type formers
4. Polynomial monad
5. Propositions as types

1. Dependent type theory

Types:

$$A, B, \ \ldots, \ A \times B, \ A \to B, \ \ldots$$

Terms:

$$x : A, \ b : B, \ \langle a, b \rangle, \ \lambda x. b(x), \ \ldots$$

Dependent Types:

$x : A \vdash B(x)$ "indexed families of types"

Type Forming Operations:

$$\sum_{x:A} B(x), \quad \prod_{x:A} B(x), \ \ldots$$

Equations:

$$s = t : A$$

1. Dependent type theory: Rules

Contexts:

$$\frac{x:A \vdash B(x)}{x:A,\ y:B(x) \vdash} \qquad \frac{\Gamma \vdash C}{\Gamma, z:C \vdash}$$

1. Dependent type theory: Rules

Contexts:

$$\frac{x : A \vdash B(x)}{x : A,\ \ y : B(x) \vdash} \qquad \frac{\Gamma \vdash C}{\Gamma, z : C \vdash}$$

Sums:

$$\frac{\Gamma, x : A \vdash B(x)}{\Gamma \vdash \sum_{x:A} B(x)} \qquad \frac{\Gamma \vdash a : A, \quad \Gamma \vdash b : B(a)}{\Gamma \vdash \langle a, b\rangle : \sum_{x:A} B(x)}$$

$$\frac{\Gamma \vdash c : \sum_{x:A} B(x)}{\Gamma \vdash \mathsf{fst}\, c : A} \qquad \frac{\Gamma \vdash c : \sum_{x:A} B(x)}{\Gamma \vdash \mathsf{snd}\, c : B(\mathsf{fst}\, c)}$$

$$\Gamma \vdash \mathsf{fst}\langle a, b\rangle = a : A \qquad \Gamma \vdash \mathsf{snd}\langle a, b\rangle = b : B$$

$$\Gamma \vdash \langle \mathsf{fst}\, c, \mathsf{snd}\, c\rangle = c : \sum_{x:A} B(x)$$

1. Dependent type theory: Rules

Contexts:

$$\frac{x : A \vdash B(x)}{x : A,\ \ y : B(x) \vdash} \qquad \frac{\Gamma \vdash C}{\Gamma, z : C \vdash}$$

Sums:

$$\frac{\Gamma, x : A \vdash B(x)}{\Gamma \vdash \sum_{x:A} B(x)} \qquad \frac{\Gamma \vdash a : A, \quad \Gamma \vdash b : B(a)}{\Gamma \vdash \langle a, b \rangle : \sum_{x:A} B(x)}$$

$$\frac{\Gamma \vdash c : \sum_{x:A} B(x)}{\Gamma \vdash \mathsf{fst}\, c : A} \qquad \frac{\Gamma \vdash c : \sum_{x:A} B(x)}{\Gamma \vdash \mathsf{snd}\, c : B(\mathsf{fst}\, c)}$$

$$\Gamma \vdash \mathsf{fst}\langle a, b \rangle = a : A \qquad \Gamma \vdash \mathsf{snd}\langle a, b \rangle = b : B$$

$$\Gamma \vdash \langle \mathsf{fst}\, c, \mathsf{snd}\, c \rangle = c : \sum_{x:A} B(x)$$

1. Dependent type theory: Rules

Sums:

$$\frac{x : A \vdash B(x)}{\sum_{x:A} B(x)} \qquad \frac{a : A \qquad b : B(a)}{\langle a, b\rangle : \sum_{x:A} B(x)}$$

$$\frac{c : \sum_{x:A} B(x)}{\mathsf{fst}\, c : A} \qquad \frac{c : \sum_{x:A} B(x)}{\mathsf{snd}\, c : B(\mathsf{fst}\, c)}$$

$$\mathsf{fst}\langle a, b\rangle = a : A \qquad \mathsf{snd}\langle a, b\rangle = b : B$$

$$\langle \mathsf{fst}\, c, \mathsf{snd}\, c\rangle = c : \sum_{x:A} B(x)$$

1. Dependent type theory: Rules

Products:

$$\frac{x : A \vdash B(x)}{\prod_{x:A} B(x)} \qquad \frac{x : A \vdash b : B(x)}{\lambda x.b : \prod_{x:A} B(x)}$$

$$\frac{a : A \qquad f : \prod_{x:A} B(x)}{fa : B(a)}$$

$$x : A \vdash (\lambda x.b)x = b : B(x)$$

$$\lambda x.fx = f : \prod_{x:A} B(x)$$

1. Dependent type theory: Rules

Products:

$$\frac{x : A \vdash B(x)}{\prod_{x:A} B(x)} \qquad \frac{x : A \vdash b : B(x)}{\lambda x.b : \prod_{x:A} B(x)}$$

$$\frac{a : A \qquad f : \prod_{x:A} B(x)}{fa : B(a)}$$

$$x : A \vdash (\lambda x.b)x = b : B(x)$$

$$\lambda x.fx = f : \prod_{x:A} B(x)$$

Substitution:

$$\frac{\sigma : \Delta \to \Gamma \qquad \Gamma \vdash a : A}{\Delta \vdash a[\sigma] : A[\sigma]}$$

2. Natural models

Definition

A natural transformation $f : Y \to X$ of presheaves on a category $\mathbb{C}$ is called *representable* if its pullback along any $\mathrm{y}C \to X$ is representable:

$$\begin{array}{ccc} \mathrm{y}D & \longrightarrow & Y \\ \big\downarrow & \lrcorner & \big\downarrow f \\ \mathrm{y}C & \longrightarrow & X \end{array}$$

2. Natural models

Definition

A natural transformation $f : Y \to X$ of presheaves on a category $\mathbb{C}$ is called *representable* if its pullback along any $\mathrm{y}C \to X$ is representable:

$$\begin{array}{ccc} \mathrm{y}D & \longrightarrow & Y \\ \big\downarrow & \lrcorner & \big\downarrow f \\ \mathrm{y}C & \longrightarrow & X \end{array}$$

Proposition (A, Fiore)

A representable natural transformation is the same thing as a **Category with Families** *in the sense of Dybjer.*

2. Natural models

Definition

A natural transformation $f : Y \to X$ of presheaves on a category $\mathbb{C}$ is called *representable* if its pullback along any $\mathrm{y}C \to X$ is representable: for all $C \in \mathbb{C}$ and $x \in X(C)$ there is given $p : D \to C$ and $y \in Y(D)$ such that the following is a pullback:

$$\begin{array}{ccc} \mathrm{y}D & \xrightarrow{\;y\;} & Y \\ {\scriptstyle \mathrm{y}p}\downarrow & \lrcorner & \downarrow{\scriptstyle f} \\ \mathrm{y}C & \xrightarrow[\;x\;]{} & X \end{array}$$

Proposition (A, Fiore)

A representable natural transformation equipped with a choice of such pullbacks is the same thing as a **Category with Families** *in the sense of Dybjer.*

2. Natural models

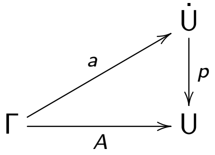
Write the objects and arrows of $\mathbb{C}$ as $\sigma : \Delta \to \Gamma$, thinking of a *category of contexts and substitutions*.

Let $p : \dot{\mathsf{U}} \to \mathsf{U}$ be a representable map of presheaves on $\mathbb{C}$.

Think of U as the *presheaf of types*, $\dot{\mathsf{U}}$ as the *presheaf of terms*, and then p gives the type of a term.

$$\begin{aligned} \Gamma \vdash A \quad &\approx \quad A \in \mathsf{U}(\Gamma) \\ \Gamma \vdash a : A \quad &\approx \quad a \in \dot{\mathsf{U}}(\Gamma) \end{aligned}$$

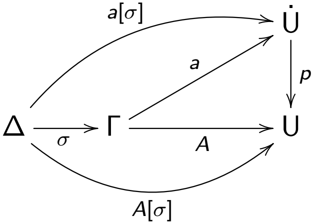
where $A = p \circ a$.



2. Natural models

Naturality of $p : \dot{\mathsf{U}} \to \mathsf{U}$ means that for any *substitution* $\sigma : \Delta \to \Gamma$, we have the required action on types and terms:

$$\begin{array}{rcl} \Gamma \vdash A & \Rightarrow & \Delta \vdash A[\sigma] \\ \Gamma \vdash a : A & \Rightarrow & \Delta \vdash a[\sigma] : A[\sigma] \end{array}$$



2. Natural models

Given any further $\tau : \Delta' \to \Delta$ we clearly have

$$A[\sigma][\tau] = A[\sigma \circ \tau] \qquad\qquad a[\sigma][\tau] = a[\sigma \circ \tau]$$

and for the identity substitution $1 : \Gamma \to \Gamma$

$$A[1] = A \qquad\qquad a[1] = a.$$

This is the basic structure of a CwF.

2. Natural models

Given any further $\tau : \Delta' \to \Delta$ we clearly have

$$A[\sigma][\tau] = A[\sigma \circ \tau] \qquad a[\sigma][\tau] = a[\sigma \circ \tau]$$

and for the identity substitution $1 : \Gamma \to \Gamma$

$$A[1] = A \qquad a[1] = a.$$

This is the basic structure of a CwF.

The remaining operation of **context extension**

$$\frac{\Gamma \vdash A}{\Gamma, x : A \vdash}$$

is given by the representability of $p : \dot{\mathrm{U}} \to \mathrm{U}$ as follows.

2. Natural models, context extension

Given $\Gamma \vdash A$ we need a new context $\Gamma.A$ together with a substitution $p_A : \Gamma.A \to A$ and a term

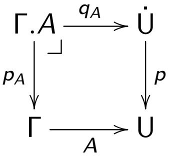
$$\Gamma.A \vdash q_A : A[p_A]\,.$$

2. Natural models, context extension

Given $\Gamma \vdash A$ we need a new context $\Gamma.A$ together with a substitution $p_A : \Gamma.A \to A$ and a term

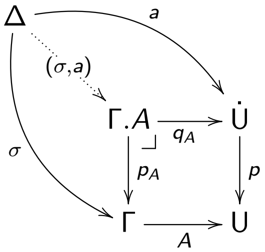
$$\Gamma.A \vdash q_A : A[p_A]\,.$$

Let $p_A : \Gamma.A \to \Gamma$ be the pullback of p along A.



The map $q_A : \Gamma.A \to \dot{\mathrm{U}}$ gives the required term $\Gamma.A \vdash q_A : A[p_A]$.

2. Natural models, context extension



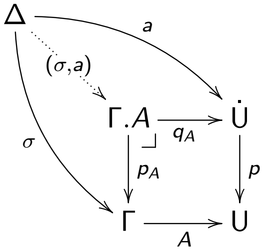
The pullback means that given any substitution $\sigma : \Delta \to \Gamma$ and term $\Delta \vdash a : A[\sigma]$ there is a map

$$(\sigma, a) : \Delta \to \Gamma.A$$

satisfying

$$p_A(\sigma, a) = \sigma$$
$$q_A[\sigma, a] = a.$$

2. Natural models, context extension



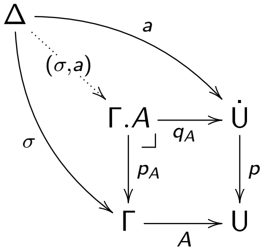
By the uniqueness of (σ, a), we also have

$$(\sigma, a) \circ \tau \ = \ (\sigma \circ \tau, a[\tau]) \qquad \text{for any } \tau : \Delta' \to \Delta$$

and

$$(p_A, q_A) = 1.$$

2. Natural models, context extension



By the uniqueness of (σ, a), we also have

$$(\sigma, a) \circ \tau \;=\; (\sigma \circ \tau, a[\tau]) \qquad \text{for any } \tau : \Delta' \to \Delta$$

and

$$(p_A, q_A) = 1.$$

These are all the laws for a CwF.

2. Natural models and initiality

- The notion of a natural model is *essentially algebraic*.

2. Natural models and initiality

- The notion of a natural model is *essentially algebraic*.
- The algebraic homomorphisms correspond to syntactic translations.

2. Natural models and initiality

- The notion of a natural model is *essentially algebraic*.
- The algebraic homomorphisms correspond to syntactic translations.
- There are *initial algebras* as well as *free algebras* over basic types and terms.

2. Natural models and initiality

- The notion of a natural model is *essentially algebraic*.
- The algebraic homomorphisms correspond to syntactic translations.
- There are *initial algebras* as well as *free algebras* over basic types and terms.
- The rules of type theory are a procedure for generating the free algebras.

2. Natural models and tribes

Let $p : \dot{\mathrm{U}} \to \mathrm{U}$ be a natural model.

The fibration $\mathcal{F} \to \mathbb{C}$ of all *display maps*

$$p_A : \Gamma.A \to \Gamma \qquad \text{for all } A : \Gamma \to \mathrm{U}$$

form a *clan* in the sense of Joyal.

2. Natural models and tribes

Let $p : \dot{\mathsf{U}} \to \mathsf{U}$ be a natural model.

The fibration $\mathcal{F} \to \mathbb{C}$ of all *display maps*

$$p_A : \Gamma.A \to \Gamma \qquad \text{for all } A : \Gamma \to \mathsf{U}$$

form a *clan* in the sense of Joyal.

Conversely, given a clan $(\mathbb{C}, \mathcal{F})$, there is a natural model in $\hat{\mathbb{C}}$,

$$\coprod_{f \in \mathcal{F}} \mathsf{y}(f) : \coprod_{f \in \mathcal{F}} \mathsf{y}(\mathrm{dom}(f)) \to \coprod_{f \in \mathcal{F}} \mathsf{y}(\mathrm{cod}(f)).$$

The natural model determines a *splitting* of the fibration $\mathcal{F} \to \mathbb{C}$.

3. Modeling the type formers

Consider the *polynomial endofunctor* $P = \mathsf{U}_! p_* \dot{\mathsf{U}}^* : \widehat{\mathbb{C}} \to \widehat{\mathbb{C}}$ determined by $p : \dot{\mathsf{U}} \to \mathsf{U}$,

$$P(X) \; = \; \sum_{A : \mathsf{U}} X^{[A]}$$

where $[A] = p^{-1}(A)$ is the fiber of $p : \dot{\mathsf{U}} \to \mathsf{U}$ at $A : \mathsf{U}$.

3. Modeling the type formers

Consider the *polynomial endofunctor* $P = \mathsf{U}_! p_* \dot{\mathsf{U}}^* : \widehat{\mathbb{C}} \to \widehat{\mathbb{C}}$ determined by $p : \dot{\mathsf{U}} \to \mathsf{U}$,

$$P(X) \;=\; \sum_{A\,:\,\mathsf{U}} X^{[A]}$$

where $[A] = p^{-1}(A)$ is the fiber of $p : \dot{\mathsf{U}} \to \mathsf{U}$ at $A : \mathsf{U}$.

Lemma

Maps $\Gamma \to P(X)$ *correspond naturally to pairs* (A, B) *where*

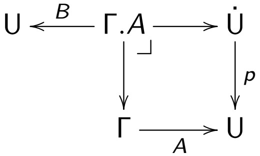
$$\begin{array}{ccccc}
X & \xleftarrow{\;B\;} & \Gamma.A & \longrightarrow & \dot{\mathsf{U}} \\
 & & \big\downarrow & \lrcorner & \big\downarrow{\scriptstyle p} \\
 & & \Gamma & \xrightarrow[\;A\;]{} & \mathsf{U}
\end{array} \; .$$

3. Modeling the type formers

Applying P to U itself therefore gives an object

$$P(\mathsf{U}) = \sum_{A:\mathsf{U}} \mathsf{U}^{[A]}$$

maps $\Gamma \to P(\mathsf{U})$ into which correspond naturally to types in an extended context $\Gamma.A \vdash B$



3. Modeling the type formers: Π

Proposition

The map $p : \dot{\mathsf{U}} \to \mathsf{U}$ *models the rules for products just if there are maps* λ, Π *making the following a pullback.*

$$\begin{array}{ccc} P(\dot{\mathsf{U}}) & \xrightarrow{\lambda} & \dot{\mathsf{U}} \\ {\scriptstyle P(p)}\downarrow & & \downarrow{\scriptstyle p} \\ P(\mathsf{U}) & \xrightarrow[\Pi]{} & \mathsf{U} \end{array}$$

3. Modeling the type formers: Π

Proposition

The map $p : \dot{\mathsf{U}} \to \mathsf{U}$ models the rules for products just if there are maps λ, Π making the following a pullback.

Proof:

$$\begin{array}{ccc} P(\dot{\mathsf{U}}) & \xrightarrow{\lambda} & \dot{\mathsf{U}} \\ \downarrow & & \downarrow p \\ P(\mathsf{U}) & \xrightarrow[\Pi]{} & \mathsf{U} \end{array}$$

$$\sum_{A:\mathsf{U}} \mathsf{U}^{[A]}$$

3. Modeling the type formers: Π

Proposition

The map $p : \dot{\mathsf{U}} \to \mathsf{U}$ models the rules for products just if there are maps λ, Π making the following a pullback.

Proof:

$$\begin{array}{ccc} P(\dot{\mathsf{U}}) & \xrightarrow{\ \lambda\ } & \dot{\mathsf{U}} \\ \downarrow & & \downarrow{\scriptstyle p} \\ P(\mathsf{U}) & \xrightarrow[\ \Pi\ ]{} & \mathsf{U} \end{array}$$

$$\sum_{A:\mathsf{U}} \mathsf{U}^{[A]}$$

$A \vdash B$ $\qquad\qquad$ $\Pi_A B$

3. Modeling the type formers: Π

Proposition

The map $p : \dot{\mathsf{U}} \to \mathsf{U}$ *models the rules for products just if there are maps* λ, Π *making the following a pullback.*

Proof:

$A \vdash b : B$ $\qquad\qquad$ $\lambda_A b$

$$\sum_{A:\mathsf{U}} \dot{\mathsf{U}}^{[A]} \qquad \begin{array}{ccc} P(\dot{\mathsf{U}}) & \xrightarrow{\ \lambda\ } & \dot{\mathsf{U}} \\ \downarrow & & \downarrow{\scriptstyle p} \\ P(\mathsf{U}) & \xrightarrow[\ \Pi\ ]{} & \mathsf{U} \end{array}$$

$$\sum_{A:\mathsf{U}} \mathsf{U}^{[A]}$$

$A \vdash B$ $\qquad\qquad$ $\Pi_A B$

3. Modeling the type formers: Π

Proposition

The map $p : \dot{\mathsf{U}} \to \mathsf{U}$ models the rules for products just if there are maps λ, Π making the following a pullback.

Proof:

f

$$\begin{array}{ccccc}
\sum_{A:\mathsf{U}} \dot{\mathsf{U}}^{[A]} & & P(\dot{\mathsf{U}}) & \xrightarrow{\lambda} & \dot{\mathsf{U}} \\
& & \downarrow & & \downarrow p \\
\sum_{A:\mathsf{U}} \mathsf{U}^{[A]} & & P(\mathsf{U}) & \xrightarrow[\Pi]{} & \mathsf{U}
\end{array}$$

$A \vdash B$ $\qquad\qquad$ $\Pi_A B$

3. Modeling the type formers: Π

Proposition

The map $p : \dot{\mathsf{U}} \to \mathsf{U}$ models the rules for products just if there are maps λ, Π making the following a pullback.

Proof:

$A \vdash f(x) : B$ $\qquad\qquad$ $\lambda_A f(x) = f$

$$\sum_{A:\mathsf{U}} \dot{\mathsf{U}}^{[A]} \qquad \begin{array}{ccc} P(\dot{\mathsf{U}}) & \xrightarrow{\lambda} & \dot{\mathsf{U}} \\ \downarrow & & \downarrow{\scriptstyle p} \\ P(\mathsf{U}) & \xrightarrow[\Pi]{} & \mathsf{U} \end{array}$$

$$\sum_{A:\mathsf{U}} \mathsf{U}^{[A]}$$

$A \vdash B$ $\qquad\qquad$ $\Pi_A B$

3. Modeling the type formers: Σ

Proposition

The map $p : \dot{\mathsf{U}} \to \mathsf{U}$ *models the rules for sums just if there are maps* (pair, Σ) *making the following a pullback*

$$\begin{array}{ccc} Q & \xrightarrow{\ \mathsf{pair}\ } & \dot{\mathsf{U}} \\ {\scriptstyle q}\downarrow & & \downarrow{\scriptstyle p} \\ P(\mathsf{U}) & \xrightarrow[\ \Sigma\ ]{} & \mathsf{U} \end{array}$$

where $q : Q \to P(\mathsf{U})$ *is the polynomial composition* $P_q = P \circ P$.

Explicitly:

$$Q \;=\; \sum_{A:\mathsf{U}} \sum_{B:\mathsf{U}^A} \sum_{x:A} B(x)$$

3. Modeling the type formers: T

Rules for a terminal type T

$$\frac{}{\vdash \mathrm{T}} \qquad\qquad \frac{}{\vdash * : \mathrm{T}} \qquad\qquad \frac{}{x : \mathrm{T} \vdash x = * : \mathrm{T}}$$

Proposition

The map $p : \dot{\mathrm{U}} \to \mathrm{U}$ *models the rules for a terminal type just if there are maps* $(*, \mathrm{T})$ *making the following a pullback.*

$$\begin{array}{ccc} 1 & \xrightarrow{\;\;*\;\;} & \dot{\mathrm{U}} \\ \big\downarrow & & \big\downarrow p \\ 1 & \xrightarrow[\;\;\mathrm{T}\;\;]{} & \mathrm{U} \end{array}$$

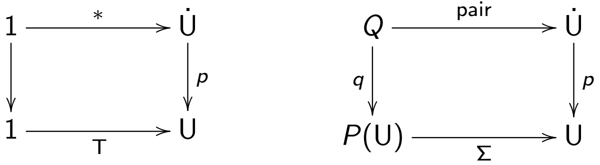
4. Polynomial monad

Consider the pullback squares for T and Σ.

$$\begin{array}{ccc} 1 & \xrightarrow{\;*\;} & \dot{\mathsf{U}} \\ \downarrow & & \downarrow{\scriptstyle p} \\ 1 & \xrightarrow[\;\mathsf{T}\;]{} & \mathsf{U} \end{array} \qquad\qquad \begin{array}{ccc} Q & \xrightarrow{\;\mathsf{pair}\;} & \dot{\mathsf{U}} \\ {\scriptstyle q}\downarrow & & \downarrow{\scriptstyle p} \\ P(\mathsf{U}) & \xrightarrow[\;\Sigma\;]{} & \mathsf{U} \end{array}$$

4. Polynomial monad

Consider the pullback squares for T and Σ.



These determine cartesian natural transformations between the corresponding polynomial endofunctors.

$$\tau : 1 \Rightarrow P \qquad\qquad \sigma : P \circ P \Rightarrow P$$

4. Polynomial monad

Theorem (A-Newstead)

A natural model $p : \dot{\mathsf{U}} \to \mathsf{U}$ *models the* T *and* Σ *type formers iff the associated polynomial endofunctor* P *has the structure maps of a cartesian monad.*

$$\tau : 1 \Rightarrow P \qquad\qquad \sigma : P \circ P \Rightarrow P$$

4. Polynomial monad

The monad laws correspond to the following type isomorphisms.

$\sigma \circ P\sigma = \sigma \circ \sigma_P$	$\sum_{a:A} \sum_{b:B(a)} C(a,b) \;\cong\; \sum_{(a,b):\sum_{a:A} B(a)} C(a,b)$
$\sigma \circ P\tau = 1$	$\sum_{a:A} 1 \;\cong\; A$
$\sigma \circ \tau_P = 1$	$\sum_{x:1} A \;\cong\; A$

4. Polynomial monad

The pullback square for Π

$$\begin{array}{ccc} P(\dot{\mathrm{U}}) & \xrightarrow{\lambda} & \dot{\mathrm{U}} \\ {\scriptstyle P(p)}\big\downarrow & & \big\downarrow{\scriptstyle p} \\ P(\mathrm{U}) & \xrightarrow[\Pi]{} & \mathrm{U} \end{array}$$

determines a cartesian natural transformation

$$\pi : P^{\mathbf{2}} p \Rightarrow p$$

where $P^{\mathbf{2}} : \hat{\mathbb{C}}^{\mathbf{2}} \to \hat{\mathbb{C}}^{\mathbf{2}}$ is the extension of P to the arrow category.

4. Polynomial monad

Theorem (A-Newstead)

A natural model $p : \dot{\mathsf{U}} \to \mathsf{U}$ *models the* Π *type former iff it has an algebra structure for the lifted endofunctor* $P^{\mathbf{2}}$.

$$\pi : P^{\mathbf{2}} p \Rightarrow p$$

4. Polynomial monad

The algebra laws correspond to the following type isomorphisms.

$\pi \circ P\pi \;=\; \pi \circ \sigma$	$\prod_{a:A} \prod_{b:B(a)} C(a,b) \;\cong\; \prod_{(a,b):\sum_{a:A} B(a)} C(a,b)$
$\pi \circ \tau \;=\; 1$	$\prod_{x:1} A \;\cong\; A$

5. Propositions as types

Let $p : \dot{\mathsf{U}} \to \mathsf{U}$ be a universe of *small* objects in $\mathcal{E} = \hat{\mathbb{C}}$.

5. Propositions as types

Let $p : \dot{\mathsf{U}} \to \mathsf{U}$ be a universe of *small* objects in $\mathcal{E} = \hat{\mathbb{C}}$.

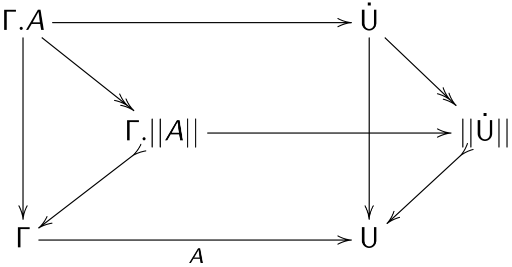
Though p is not representable in $\mathcal{E}$ it is still a natural model in $\hat{\mathcal{E}}$.

5. Propositions as types

Let $p : \dot{\mathsf{U}} \to \mathsf{U}$ be a universe of *small* objects in $\mathcal{E} = \hat{\mathbb{C}}$.

Though p is not representable in $\mathcal{E}$ it is still a natural model in $\hat{\mathcal{E}}$.

Factor p as on the right below.

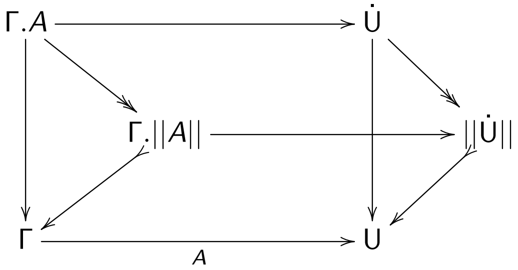


5. Propositions as types

Let $p : \dot{\mathsf{U}} \to \mathsf{U}$ be a universe of *small* objects in $\mathcal{E} = \hat{\mathbb{C}}$.

Though p is not representable in $\mathcal{E}$ it is still a natural model in $\hat{\mathcal{E}}$.

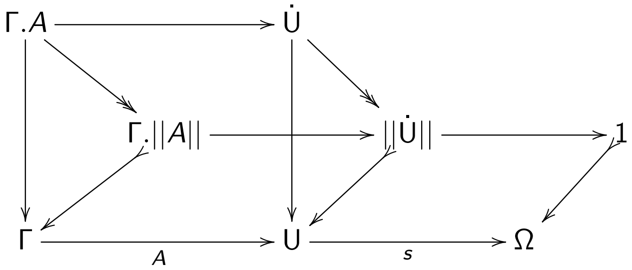
Factor p as on the right below.



So $||\dot{\mathsf{U}}|| \rightarrowtail \mathsf{U}$ is a universal family of small propositions.

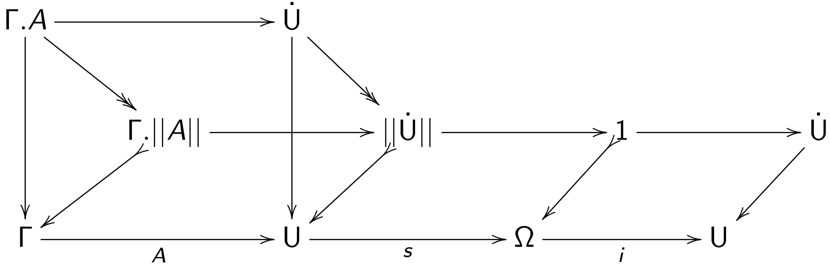
5. Propositions as types

Let $s : \mathsf{U} \to \Omega$ classify the mono $||\dot{\mathsf{U}}|| \rightarrowtail \mathsf{U}$.



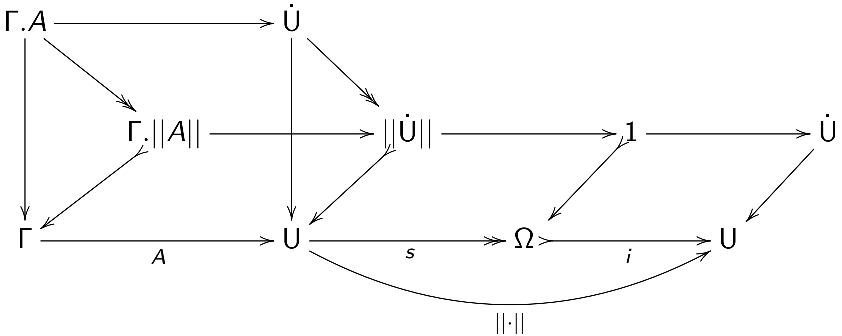
5. Propositions as types

Let $s : \mathsf{U} \to \Omega$ classify the mono $||\dot{\mathsf{U}}|| \rightarrowtail \mathsf{U}$.



Let $i : \Omega \to \mathsf{U}$ classify the family of small propositions $1 \rightarrowtail \Omega$.

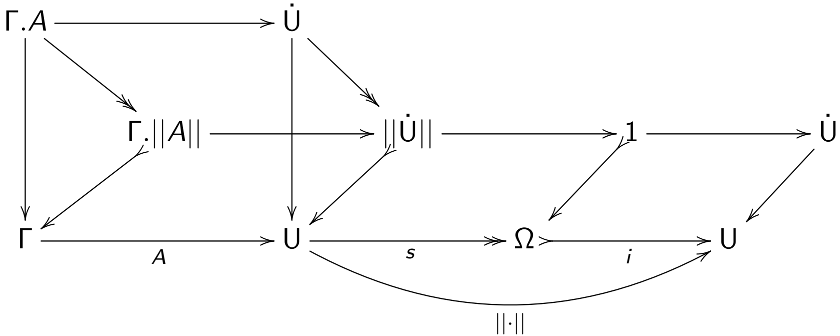
5. Propositions as types



Let

$$||\cdot|| := i \circ s : \mathsf{U} \to \mathsf{U}.$$

5. Propositions as types



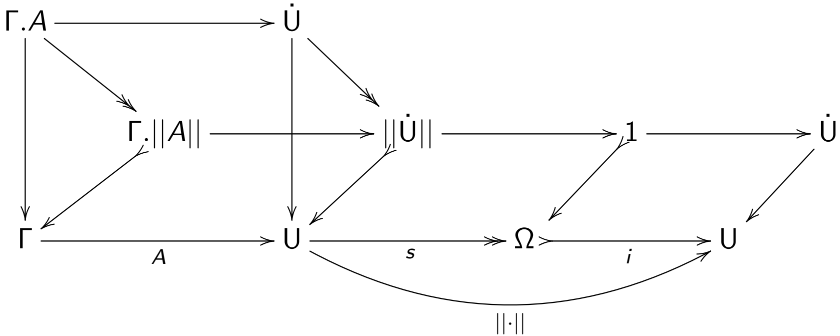
Let

$$||\cdot|| := i \circ s : \mathsf{U} \to \mathsf{U}.$$

We have

$$s \circ i = 1 : \Omega \to \Omega.$$

5. Propositions as types



Let

$$||\cdot|| := i \circ s : \mathrm{U} \to \mathrm{U}.$$

We have

$$s \circ i = 1 : \Omega \to \Omega.$$

So

$$\Omega = \mathrm{im}(||\cdot||).$$

5. Propositions as types

The following commute.

$$
\begin{array}{ccc}
\sum_{A:\mathrm{U}} \Omega^A & \xrightarrow{\exists} & \Omega \\
{\scriptstyle Pi}\downarrow & & \uparrow{\scriptstyle s} \\
\sum_{A:\mathrm{U}} \mathrm{U}^A & \xrightarrow[\Sigma]{} & \mathrm{U}
\end{array}
\qquad\qquad
\begin{array}{ccc}
\sum_{A:\mathrm{U}} \Omega^A & \xrightarrow{\forall} & \Omega \\
{\scriptstyle Pi}\downarrow & & \uparrow{\scriptstyle s} \\
\sum_{A:\mathrm{U}} \mathrm{U}^A & \xrightarrow[\Pi]{} & \mathrm{U}
\end{array}
$$

Where, recall,

$$PX \;=\; \sum_{A:\mathrm{U}} X^A$$

is the polynomial functor of the natural model $p : \dot{\mathrm{U}} \to \mathrm{U}$.

References

1. Awodey, S. (2017) Natural models of homotopy type theory, MSCS 28(2). arXiv:1406.3219

2. Awodey, S. and N. Gambino and S. Hazratpour (in progress) Kripke-Joyal semantics for homotopy type theory.

3. Awodey, S. and C. Newstead (2018) Polynomial pseudomonads and dependent type theory. arXiv:1802.00997

4. Newstead, C. (2018) Algebraic Models of Dependent Type Theory, CMU PhD thesis. arXiv:2103.06155