Types are Internal ∞-groupoids

Joint w/ Matthieu Sozeau + Antoine Allioux

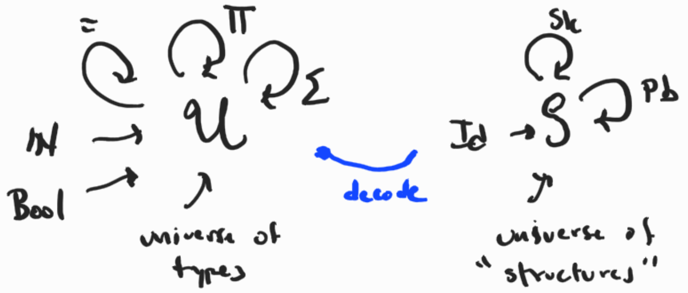
- A technique for describing infinitely coherent algebraic structures in type.
- An internal definition of ∞-groupoid.

$$\Pi_\infty : \mathrm{Type} \xrightarrow{\simeq} \infty\text{-groupoid}$$

- Main problem: higher algebraic structures are most often described using algebraic structures
  - Presheaves
  - Operads . .

  Need algebraic structures to describe algebraic structures!

- Main idea: Type theory will come equipped with some "basic structures" from which we can describe others.



$$\sum_{x:X} \prod_{y:X} x = y \qquad\qquad \mathrm{Sk}\ (\mathrm{Pb}\ \mathrm{Id}\ X)$$

Type expression $\qquad\qquad$ Structure expressions

- Concretely: take for our definition of "structure" the notion of cartesian polynomial monad.

Type $\qquad\qquad$ **$\mathbb{M}$**

- Describe by a "finite" collection of data
- Already a strong link w/ type theory (co)inductive data types (definitions)

Well understood "weak" versions

- Well-[illegible]

- The universe "Type" is the terminal polynomial monad $(\mathrm{Type}, \Sigma, \mathbb{1})$

The universe of Polynomial Monads

$$\frac{}{\mathbb{M} : \mathrm{Type}} \qquad \frac{M : \mathbb{M}}{\mathrm{Idx}\, M : \mathrm{Type}} \qquad \frac{M : \mathbb{M} \quad i : \mathrm{Idx}\, M}{\mathrm{Cns}\, i : \mathrm{Type}}$$

$$\frac{M : \mathbb{M} \quad i : \mathrm{Idx}\, M \quad c : \mathrm{Cns}_M\, i}{\mathrm{Pos}\, c : \mathrm{Type}} \qquad \frac{M : \mathbb{M} \quad i : \mathrm{Idx}\, M \quad c : \mathrm{Cns}\, i \quad p : \mathrm{Pos}\, c}{\mathrm{Typ}\, p : \mathrm{Idx}\, M}$$

$$\left(\mathrm{Idx}\, M \longleftarrow \mathrm{Pos}\, M \longrightarrow \mathrm{Cns}\, M \longrightarrow \mathrm{Idx}\, M \right)$$

→ More concretely: implementation in Agda + rewrite rules

"sorts" $\mathrm{Idx} : \mathbb{M} \to \mathrm{Type}$

"operations" $\mathrm{Cns} : (M : \mathbb{M})(i : \mathrm{Idx}\, M) \to \mathrm{Type}$

"arities" $\underset{\sim}{\mathrm{Pos}} : (M : \mathbb{M})(i : \mathrm{Idx}\, M)(c : \mathrm{Cns}\, M\, i) \longrightarrow \mathrm{Type}$

"types of arities" $\underline{\mathrm{Typ}} : (M : \mathbb{M})(i : \mathrm{Idx}\, M)(c : \mathrm{Cns}\, M\, i)(p : \mathrm{Pos}\, M\, c) \to \mathrm{Idx}\, M$

$\left(\mathrm{Type}/_{\mathrm{Idx}\, M} \to \mathrm{Type}/_{\mathrm{Idx}\, M} \right)$

$p_0 : \mathrm{Pos}\, c$

$$[M] : (\mathrm{Idx}\, M \to \mathrm{Type}) \to (\mathrm{Idx}\, M \to \mathrm{Type})$$

$$[M]\, X\, i = \sum_{c : \mathrm{Cns}\, i} \underbrace{(p : \mathrm{Pos}\, c) \to X(\mathrm{Typ}\, p)}_{\text{decorations}}$$

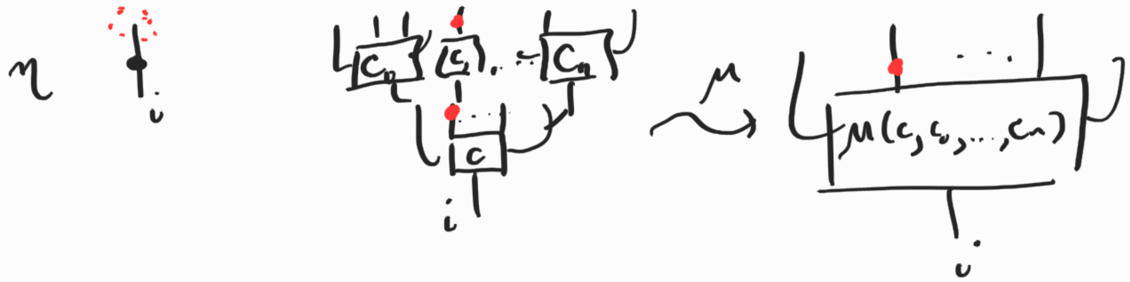
$X : \mathrm{Idx}\, M \to \mathrm{Type}$

$$\begin{tikzcd} \Sigma E \arrow[r] \arrow[d] & B \arrow[r, "E"] \arrow[d] & \mathrm{Type} \\ 1 & 1 & \end{tikzcd}$$

$\mathrm{Idx} = \mathbb{1}$
$\mathrm{Cns} = B$
$\mathrm{Pos} = E$

Algebraic Structure

$$\begin{cases} \eta : (M : \mathbb{M})(i : \mathrm{Idx}\, M) \to \mathrm{Cns}\, M\, i \\ \mu : (M : \mathbb{M})(i : \mathrm{Idx}\, M)(c : \mathrm{Cns}\, M\, i) \\ \qquad (\delta : (p : \mathrm{Pos}\, c) \to \underline{\underline{\mathrm{Cns}}}\, (\mathrm{Typ}\, p)) \\ \qquad \to \mathrm{Cns}\;\; i \end{cases}$$



→ Need our polynomial monads to be <u>cartesian</u>.

$$\mathrm{Pos}\,(\eta\, i) \simeq \mathbb{1} \qquad \mathrm{Pos}\,(\mu\, c\, \delta) \simeq \sum_{p : \mathrm{Pos}\, c} \mathrm{Pos}\,(\delta\, p)$$

- Equip the position types with intro/elim rules forcing such an equivalence to exist.

$$\left.\begin{array}{l} \mathrm{pos}\;(\_,\_) \\ \mathrm{pos\text{-}fst}\;\ldots \\ \mathrm{pos\text{-}snd}\;\ldots \end{array}\right\}$$

Definitional assoc + unit laws

+ $\mu\, c\, (\lambda p.\, \eta\, (\mathrm{Typ}\, p)) \rightsquigarrow c$
+ unit
+ assoc

Also need <u>dependent monads</u>.

$$\mathbb{M}{\downarrow} : \mathbb{M} \to \mathrm{Type} \qquad \begin{array}{l} M : \mathbb{M} \\ \rightsquigarrow \text{“}\mathbb{M}{\downarrow}\, M\text{”} \end{array}$$

universe of monads equipped with a cartesian morphism to M.

$$
\begin{tikzcd}
\overline{\mathrm{Idx}'}\,M \arrow[d] & \overline{\mathrm{Pos}{\downarrow}}\,M \arrow[l] \arrow[r] \arrow[d] & \overline{\mathrm{Cns}{\downarrow}}\,M \arrow[r] \arrow[d] & \overline{\mathrm{Idx}{\downarrow}}\,M \arrow[d] \\
\mathrm{Idx}\ M & \mathrm{Pos}\ M \arrow[l] \arrow[r] & \mathrm{Cns}\ M \arrow[r] & \mathrm{Idx} \arrow[d] \\
 & & & \mathrm{Type}
\end{tikzcd}
$$

$$\mathrm{Idx}{\downarrow} : (M : \mathbb{M})(M' : \mathbb{M}{\downarrow}\, M)(i : \mathrm{Idx}\ M) \longrightarrow \mathrm{Type}$$

$$\mathrm{Cns}{\downarrow} : (M : \mathbb{M})(M' : \mathbb{M}{\downarrow}\, M)(i : \mathrm{Idx}\ M)$$
$$(i' : \mathrm{Idx}{\downarrow}\ i)(c : \mathrm{Cns}\ M\ i) \longrightarrow \mathrm{Type}$$

$$\eta{\downarrow} : (M : \mathbb{M})(M' : \mathbb{M}{\downarrow}\, M)(i : \mathrm{Idx}\ M)$$
$$(i' : \mathrm{Idx}{\downarrow}\ i) \to \mathrm{Cns}{\downarrow}\ i'\ (\eta\ i)$$

$$\mu{\downarrow} : (M : \mathbb{M})(M' : \mathbb{M}{\downarrow}\, M)(i : \mathrm{Idx}\ M)$$
$$(c : \mathrm{Cns}\ i)(\delta : (p : \mathrm{Pos}\ c) \to \mathrm{Cns}\ (\mathrm{Typ}\ p))$$
$$(i' : \mathrm{Idx}{\downarrow}\ i)(c : \mathrm{Cns}{\downarrow}\ i'\ c)$$
$$(\delta' : \underbrace{(p : \mathrm{Pos}\ c)} \longrightarrow \mathrm{Cns}{\downarrow}\ (\mathrm{Typ}'\ p)\ (\delta\ p))$$
$$\longrightarrow \mathrm{Cns}{\downarrow}\ i\ (\mu\ c\ \delta)$$

$(M, M') \rightsquigarrow$ "monad extension"

$$\mathbb{M} \xleftarrow{\Sigma_M} \mathbb{M}{\downarrow}$$

Adding constants and constructs:

$$\mathrm{Id} : \mathbb{M}$$

$$
\begin{tikzcd}
A \arrow[r, equal] \arrow[d] & A \arrow[r, equal] \arrow[d] & A \arrow[r, equal] \arrow[d] & A \arrow[d] \\
1 & \mathbb{1} \arrow[l] \arrow[r] & \mathbb{1} \arrow[r] & \mathbb{1}
\end{tikzcd}
$$

$$\mathrm{Idx}\ \mathrm{Id} = \mathbb{1}$$
$$\mathrm{Cns}\ \mathrm{Id}\ \_ = \mathbb{1}$$

$$\mathrm{Id}{\downarrow} : \mathrm{Type} \longrightarrow \mathbb{M}{\downarrow}\ \mathrm{Id}$$

$\mathrm{Pos}\ \mathrm{Id}\ \_\ \_\ \_ = \top$

$\mathrm{Typ}\ \mathrm{Id}\ \_\ \_\ \_\ \_ = tt$

$\eta\ \mathrm{Id} = tt$

$\mu\ \mathrm{Id}\ \_\ \_\ \_ = tt$

$\mathrm{Id}{\downarrow}\ A\ \_ = A$

$\mathrm{Cns}{\downarrow}\ A\ \_\ \_\ \_\ \_ = \top$

$\mathrm{Typ}{\downarrow}\ a\ \_\ \_\ \_\ \_ = a$

$\eta{\downarrow}\ \_ = tt$

$\mu{\downarrow} = tt$

The Baez-Dolan slice construction

$$\mathrm{Slice} : \mathbb{M} \longrightarrow \mathbb{M}$$

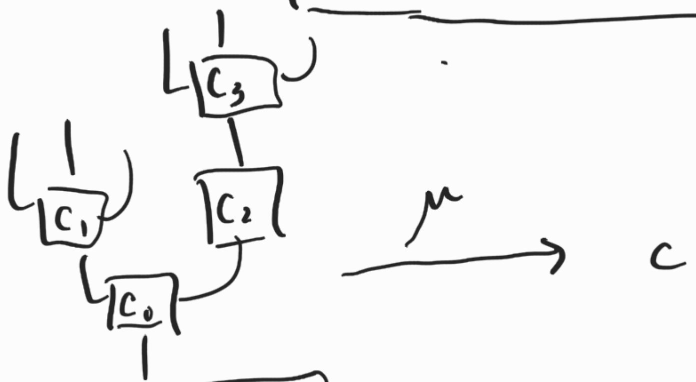
Slice M is the "monad of relations in M"

$$M \quad \mathrm{Slice}\ M \quad \mathrm{Slice}\ (\mathrm{Slice}\ M) \ \ldots$$

$$\mathrm{Idx}\ (\mathrm{Slice}\ M) = \sum_{i : \mathrm{Idx}\ M} \mathrm{Cns}\ M\ i$$

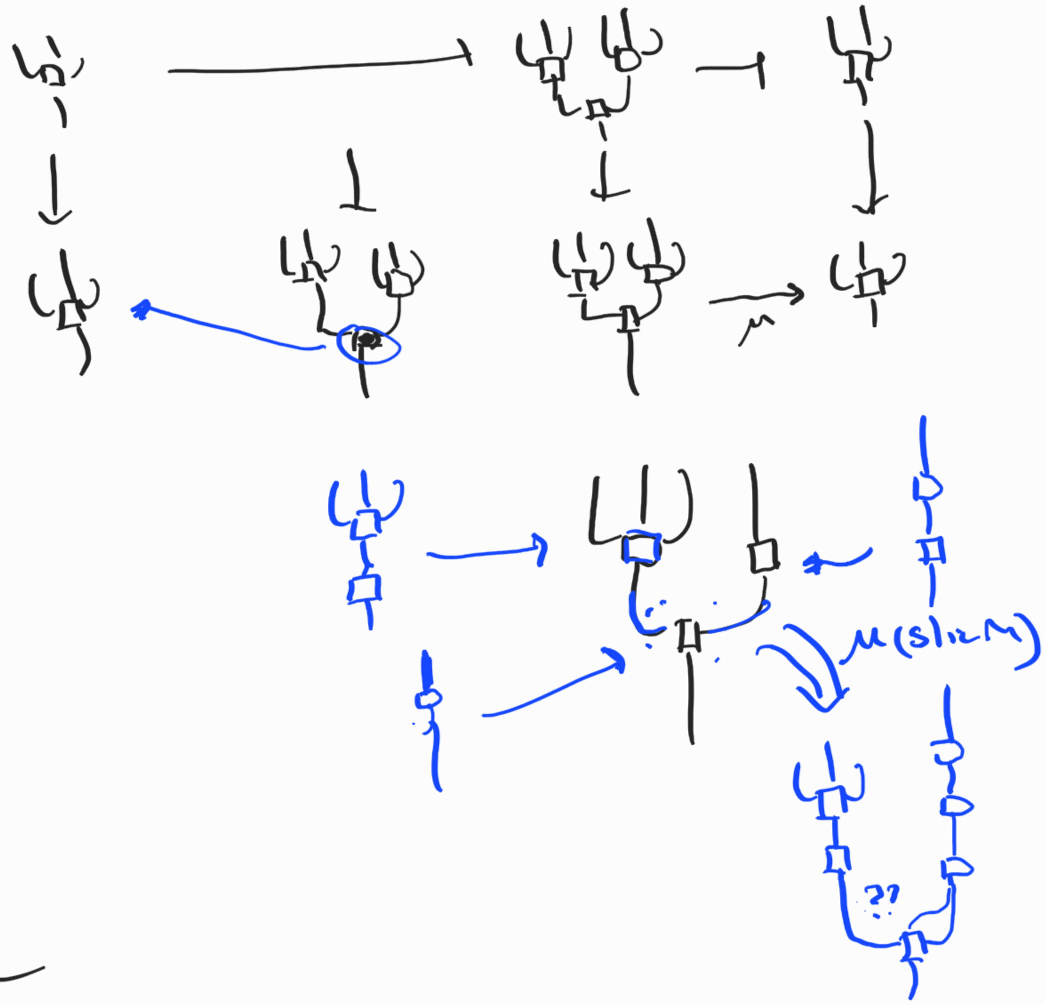
```
data Cns (Slice M) : Idx (Slice M) → Type where
   lf : (i : Idx M) → Cns (Slc M) (i , η i)
   nd : (i : Idx M)(c : Cns M i)
          → (δ : (p : Pos c) → Cns (Typ p))
          → (ε : (p : Pos c) → Cns (Slice M) (Typ p , δ p))
          → Cns (Slice M) (i , μ c δ)
```

An element $\boxed{\sigma : \mathrm{Cns}\ (\mathrm{Slice}\ M)\ (i , \underline{c})}$

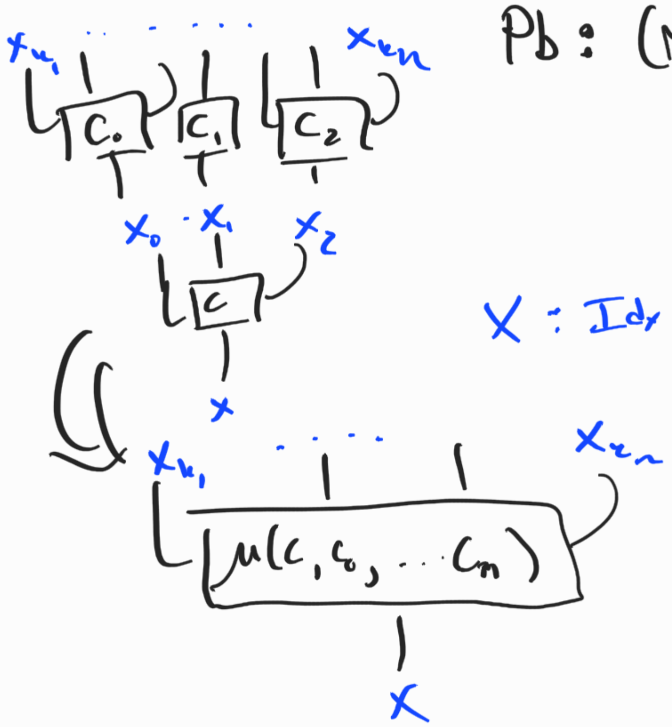


$\boxed{\mu\ \sigma = c}$

There's a dependent Slice construction as well.



Pullback Monad



$$Pb : (M : \mathbb{M})(X : Idx\ M \to Type) \to \mathbb{M}$$

$X : Idx\ M \to Type$

$$Pb{\downarrow} : (M : \mathbb{M})(X : Idx\ M \to Type) \downarrow$$

$$(M' : \mathbb{M}{\downarrow}\ M)(Y : (i : Idx\ M)(i' : Idx{\downarrow}\ i) \quad)$$

$(x : X\, i) \to \mathrm{Type}\,)$

$\to \mathbb{M}\!\downarrow (\mathrm{Pb}\, M\, X))$

Opetopic Types

Def (Baez-Dolan) An M-Opetopic type is

$$C : \mathrm{Idx}\, M \to \mathrm{Type}$$

$$R : \underbrace{(\mathrm{Slice}\,(\mathrm{Pb}\, M\, C))}_{\text{monad expression}} - \text{Opetopic Type}$$

$X : \mathrm{OpetopicType}\, M$

$$\left.\begin{array}{ll} M = M_0 & X_0 : \mathrm{Idx}\, M_0 \to \mathrm{Type} \\ \mathrm{Slice}\,(\mathrm{Pb}\, M_0\, CX) = M_1 & X_1 : \mathrm{Idx}\, M_1 \to \mathrm{Type} \\ \mathrm{Slice}\,(\mathrm{Pb}\, M_1\, X_1) = M_2 & X_2 : \mathrm{Idx}\, M_2 \to \mathrm{Type} \end{array}\right\}$$

Intuition: An M-opetopic type is the underlying data of a weak M-algebra.

$$CX =: X_0$$

$$CRX =: X_1$$

$$CRRX =: X_2$$

$$\left\{\begin{array}{l} X_0 : \mathrm{Idx}\, M \to \mathrm{Type} \\ X_1 : \mathrm{Idx}\,(\mathrm{Slice}\,(\mathrm{Pb}\, M\, X)) \to \mathrm{Type} \end{array}\right.$$

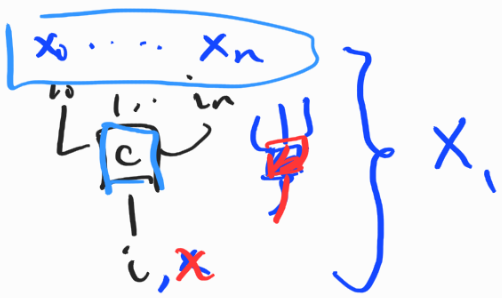
$$\mathrm{Idx}\,(\mathrm{Slice}\,(\mathrm{Pb}\, M\, X)) = \left(\sum_{i : \mathrm{Idx}\, M} \sum_{x : X_i} \sum_{c : \mathrm{Cns}\, i} (p : \mathrm{Pos}\, c) \to X\,(\mathrm{Typ}\, p) \right)$$

(i, x, c, δ)

$$[M]X \xrightarrow{\alpha} X \qquad X : \mathrm{Idx}\, M \to \mathrm{Type}$$

$$\begin{aligned}\alpha : \ & (i : \mathrm{Idx})(c : \mathrm{Cns}\, M) \\ & (\delta : (p : \mathrm{Pos}\, c) \to X\, (\mathrm{Typ}\, p)) \\ & \to \underline{X\, i}\end{aligned}$$

X_1 is the type of "relations" filling the following picture



$$\text{is-multiplicative}\ X_1 := $$
$$(i : \mathrm{Idx}\, M)(c : \mathrm{Cns}\, i)(\delta : (p : \mathrm{Pos}\, c) \to X\, (\mathrm{Typ}\, p))$$
$$\to \text{is-contr}\Big(\sum_{x : X\, i} X_1\, (i, x, c, \delta)\Big)$$

Def An M-opetopic type X is *fibrant* if:

1. is-multiplicative X_1
2. is-fibrant $(R\, X)$

Def An ∞-groupoid is a fibrant Id-opetopic type.

Thm

$$\mathrm{Type} \xrightarrow{\simeq} \infty\text{-groupoid}$$

$\mathbb{M}$ $\qquad \underline{M} \qquad \mathrm{Slc}(M) \qquad \mathrm{Slc}(\mathrm{Slc}\, M) \qquad \mathrm{Slc}(\mathrm{Slc}(\mathrm{Slc}\, M)))$

$\mathrm{Slc}(\mathrm{Id})$

A_∞-Type → fibrant opetopic type over $\underline{\mathrm{Slc}(\mathrm{Id})}$

$\boxed{\cancel{E_\infty}}$ $\qquad \mathbb{M} \supset \mathrm{Id}$

$\mathrm{Slc}(\mathbb{U})$

Σ is not def'n associative

∞-cat $\qquad X : \mathbb{I} \to \mathrm{Type}$

$\boxed{C : \underline{\mathrm{Slc}\,(\mathrm{Pb}\,\mathbb{I}\,X)} \qquad C \text{ is fibrant}}$

$$\underset{m}{M} \rightsquigarrow \mathrm{Slc}\, M$$

$$\begin{tikzcd} & & X \arrow[d] \arrow[dll, bend right] \\ M & X_0 \arrow[r] & x \end{tikzcd}$$

$\mathrm{Type} \leftarrow \cdots \rightarrow \underline{\infty\text{-groupoid}}$

$X : \mathrm{OpetopeType}\ \mathrm{Id}$

$X_0 : \mathrm{Idx}\ \mathrm{Id} \to \mathrm{Type}$

$\mathrm{Idx}\ \mathrm{Id} = \mathbb{1}$

X : OpticType M $\qquad$ $X_0 \, t \, t$