Constructing Inverse Diagrams in Homotopical Type Theory

(an update)

Josh Chen & Nicolai Kraus

EUProofNet WG6 Meeting

4 Apr 2024

Inverse Diagrams

... are functors

$$\underbrace{X : I \longrightarrow \mathcal{C}}_{\text{Inverse}} \quad = \quad \underbrace{\mathcal{D}^{op} \longrightarrow \mathcal{C}}_{\mathcal{D}\text{ direct}}$$

Inverse Diagrams

... are functors

$$\underbrace{X : \mathcal{I} \longrightarrow \mathcal{C}}_{\text{Inverse}} \quad = \quad \underbrace{\mathcal{D}^{op} \longrightarrow \mathcal{C}}_{\mathcal{D}\text{ direct}}$$

e.g. $\mathcal{I} = \omega^{op}$

$$0 \leftarrow 1 \leftarrow 2 \leftarrow 3 \leftarrow \cdots$$

Inverse Diagrams

... are functors

$$X : \underbrace{\mathcal{I}}_{\text{Inverse}} \to \mathcal{C} \quad = \quad \underbrace{\mathcal{D}^{op}}_{\mathcal{D}\text{ direct}} \to \mathcal{C}$$

e.g. $\mathcal{I} = \mathbb{N}_\perp$

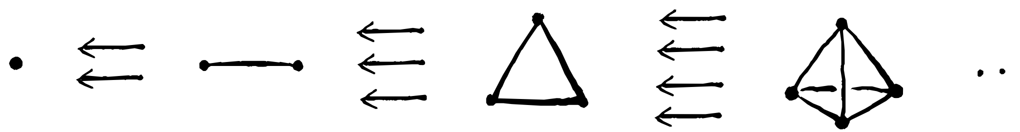
$$\begin{tikzcd}
 & & \perp \arrow[dll] \arrow[dl] \arrow[d] \arrow[dr] \arrow[drr] & & & \cdots \\
0 & 1 & 2 & 3 & 4 & \cdots
\end{tikzcd}$$

Inverse Diagrams

... are functors

$$X : \underbrace{\mathcal{I}}_{\text{Inverse}} \longrightarrow \mathcal{C} \quad = \quad \underbrace{\mathcal{D}^{op}}_{\mathcal{D}\text{ direct}} \longrightarrow \mathcal{C}$$

e.g. $\mathcal{I} = \Delta_+^{op}$



Why?

- Presheaf + other models of HoTT
- Parametricity, canonicity results
- Higher + ∞-categorical structures

[e.g. Kock'05, Shulman'15, Kapulkin–Lumsdaine '21]

$\rightsquigarrow$ Both metatheory and applications.

IDEA:

Coslices of simple I inductively encode dependency contexts

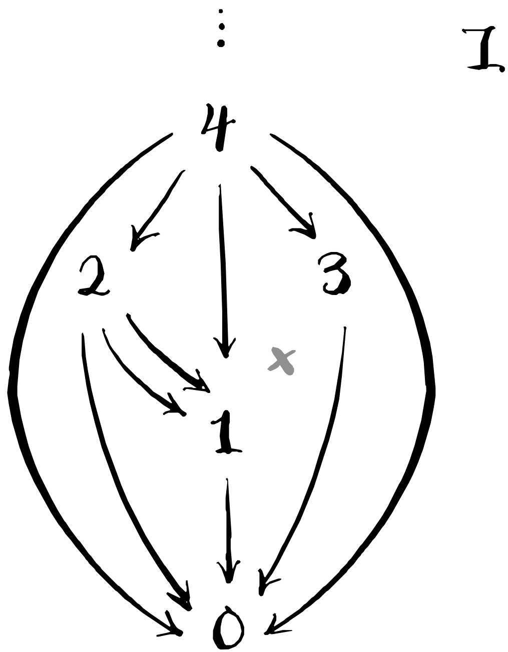
Def. (Countably) Simple category I:

- $I_0 \cong \mathbb{N}$ (countable, ordered)
- $f : I(i,j) \to i > j$ (inverse)
- i/I finite (finite fan-out)

$\Big\}$ "simple" [Makkai '98]

Examples: ω^{op}, Δ_+^{op}, $\square_+^{op}$

Simple $\mathcal{I}$ encode dependency contexts, inductively.



All triangles commute, the square x does not.

i	M_i	Assume type-in-ctxt

Simple I encode dependency contexts, inductively.

$$\begin{array}{ccccc} & & \vdots & & \\ & & & & I \\ & & 4 & & \\ 2 & & & 3 & \\ & & 1 & & \\ & & 0 & & \end{array}$$

All triangles commute, the square x does not.

i	M_i	Assume type-in-ctxt
0	$\cdot$	

Simple I encode dependency contexts, inductively.

$$
\begin{tikzcd}
 & \vdots & & \mathrm{I} \\
 & 4 & & \\
2 & & 3 & \\
 & 1 \arrow[d] & & \\
 & 0 & &
\end{tikzcd}
$$

All triangles commute, the square x does not.

i	M_i	Assume type-in-ctxt
0	$\cdot$	$\vdash \; A_0$
1		

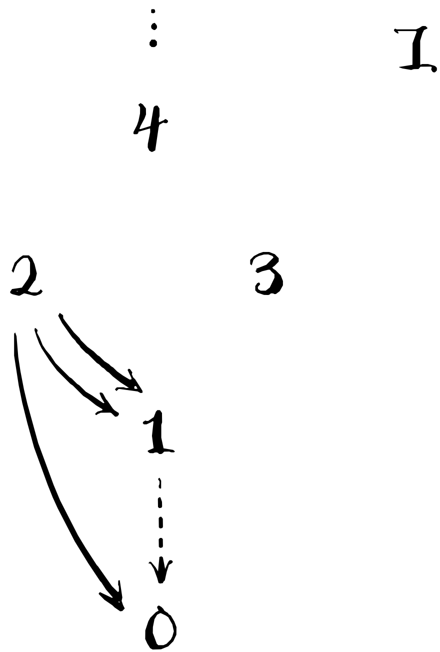
Simple I encode dependency contexts, inductively.

$$\begin{tikzcd}
 & \vdots & & I \\
 & 4 & & \\
2 & & 3 & \\
 & 1 \arrow[d, "x"] & & \\
 & 0 & &
\end{tikzcd}$$

All triangles commute, the square x does not.

i	M_i	Assume type-in-ctxt
0	$\cdot$	$\vdash A_0$
1	$x : A_0$	

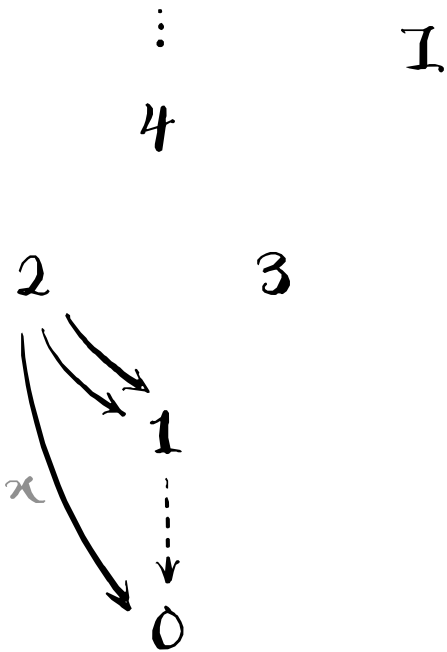
Simple I encode dependency contexts, inductively.



i	M_i	Assume type-in-ctxt
0	$\cdot$	$\vdash A_0$
1	$x : A_0$	$\vdash A_1$
2		

All triangles commute, the square x does not.

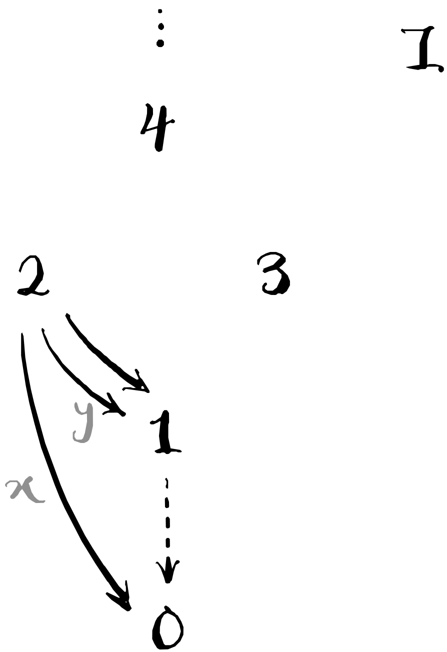
Simple I encode dependency contexts, inductively.



All triangles commute,
the square x does not.

i	M_i	Assume type-in-ctxt
0	$\cdot$	$\vdash A_0$
1	$x : A_0$	$\vdash A_1$
2	$x : A_0$	

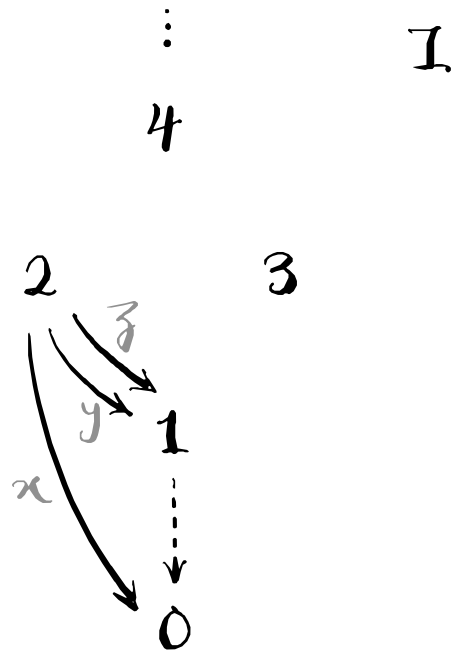
Simple I encode dependency contexts, inductively.



All triangles commute, the square x does not.

i	M_i	Assume type-in-ctxt
0	$\cdot$	$\vdash A_0$
1	$x : A_0$	$\vdash A_1$
2	$x : A_0,$ $y : A_1(x)$	

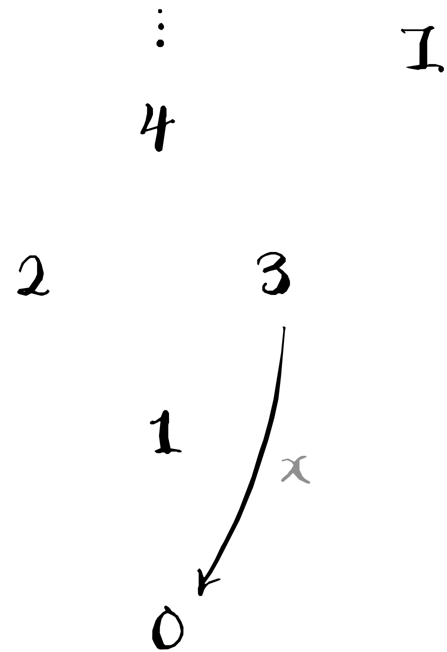
Simple I encode dependency contexts, inductively.



All triangles commute, the square x does not.

i	M_i	Assume type-in-ctxt
0	$\cdot$	$\vdash A_0$
1	$x : A_0$	$\vdash A_1$
2	$x : A_0,$ $y : A_1(x),$ $z : A_1(x)$	

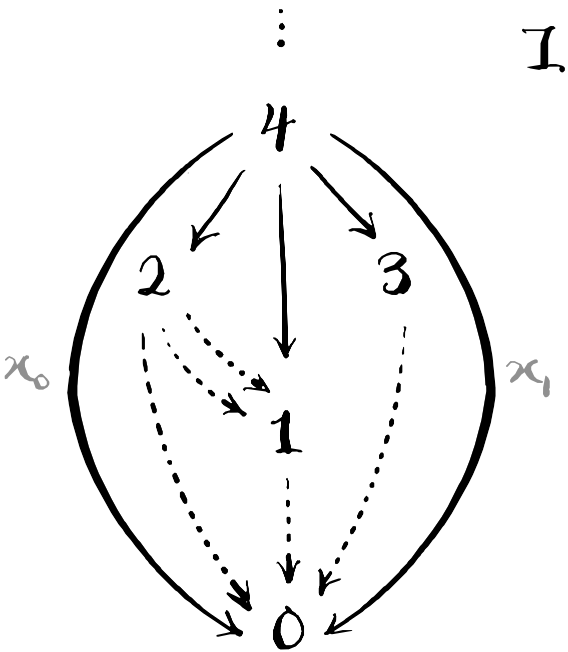
Simple I encode dependency contexts, inductively.



All triangles commute, the square x does not.

i	M_i	Assume type-in-ctxt
0	$\cdot$	$\vdash A_0$
1	$x : A_0$	$\vdash A_1$
2	$x : A_0$, $y : A_1(x)$, $z : A_1(x)$	$\vdash A_2$
3	$x : A_0$	

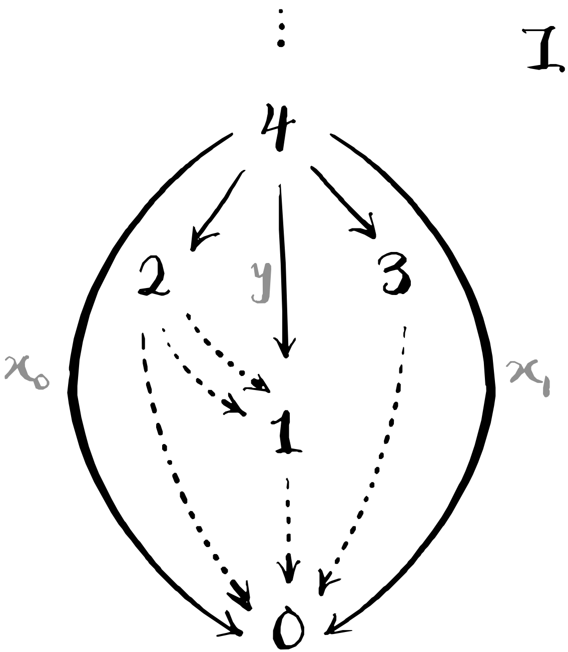
Simple I encode dependency contexts, inductively.



All triangles commute, the square x does not.

i	M_i	Assume type-in-ctxt
0	$\cdot$	$\vdash A_0$
1	$x : A_0$	$\vdash A_1$
2	$x : A_0,$ $y : A_1(x),$ $z : A_1(x)$	$\vdash A_2$
3	$x : A_0$	$\vdash A_3$
4	$x_0, x_1 : A_0$	

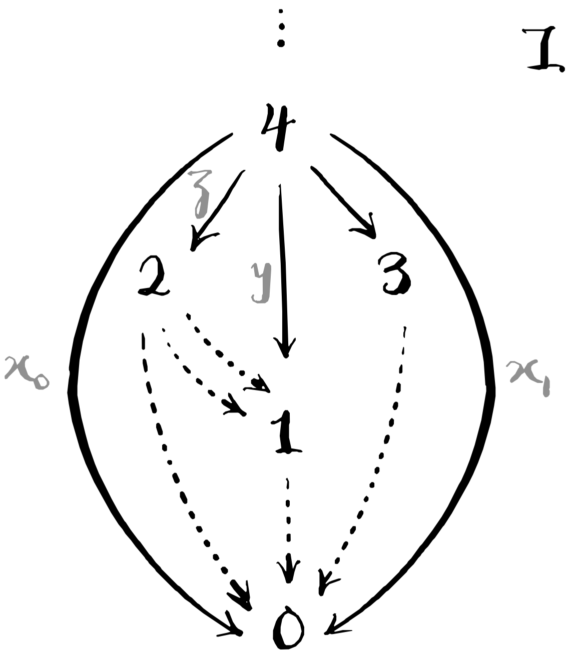
Simple I encode dependency contexts, inductively.



All triangles commute, the square x does not.

i	M_i	Assume type-in-ctxt
0	$\cdot$	$\vdash \; A_0$
1	$x : A_0$	$\vdash \; A_1$
2	$x : A_0$, $y : A_1(x)$, $z : A_1(x)$	$\vdash \; A_2$
3	$x : A_0$	$\vdash \; A_3$
4	$x_0, x_1 : A_0$, $y : A_1(x_0)$	

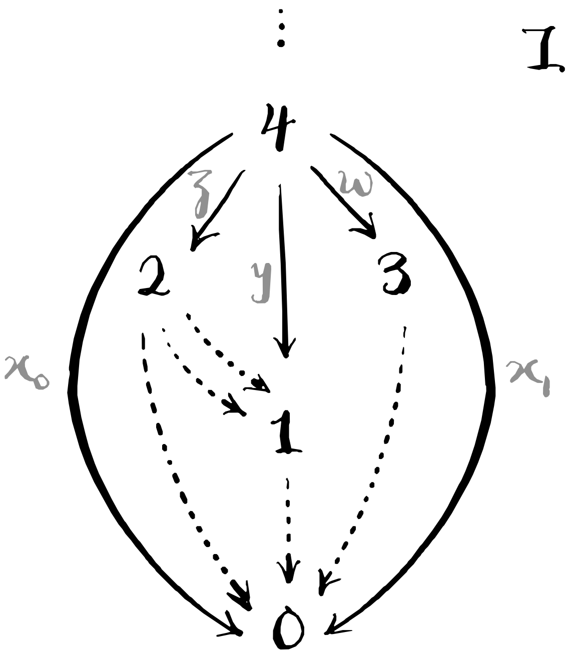
Simple $\mathbb{I}$ encode dependency contexts, inductively.



All triangles commute, the square x does not.

i	M_i	Assume type-in-ctxt
0	$\cdot$	$\vdash A_0$
1	$x : A_0$	$\vdash A_1$
2	$x : A_0,$ $y : A_1(x),$ $z : A_1(x)$	$\vdash A_2$
3	$x : A_0$	$\vdash A_3$
4	$x_0, x_1 : A_0,$ $y : A_1(x_0),$ $z : A_2(x_0, y, y)$	

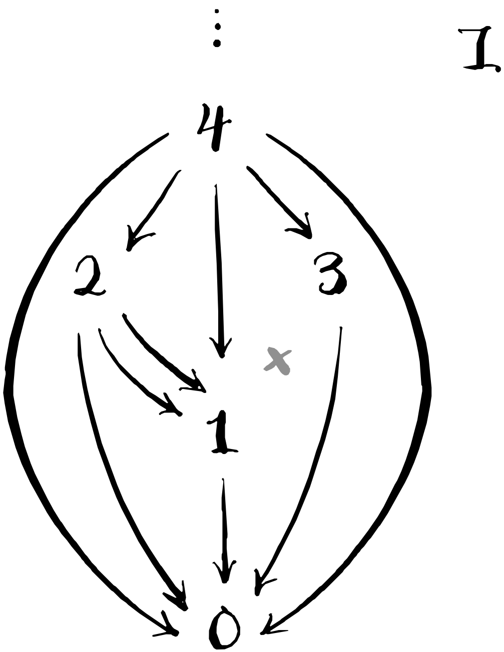
Simple $\mathbb{I}$ encode dependency contexts, inductively.



All triangles commute, the square x does not.

i	M_i	Assume type-in-ctxt
0	$\cdot$	$\vdash A_0$
1	$x : A_0$	$\vdash A_1$
2	$x : A_0,$ $y : A_1(x),$ $z : A_1(x)$	$\vdash A_2$
3	$x : A_0$	$\vdash A_3$
4	$x_0, x_1 : A_0,$ $y : A_1(x_0),$ $z : A_2(x_0, y, y),$ $w : A_3(x_1)$	

Simple I encode dependency contexts, inductively.



All triangles commute, the square x does not.

i	M_i	Assume type-in-ctxt
0	$\cdot$	$\vdash \; A_0$
1	$x : A_0$	$\vdash \; A_1$
2	$x : A_0$, $y : A_1(x)$, $z : A_1(x)$	$\vdash \; A_2$
3	$x : A_0$	$\vdash \; A_3$
4	$x_0, x_1 : A_0$, $y : A_1(x_0)$, $z : A_2(x_0, y, y)$, $w : A_3(x_1)$	$\vdash \; A_4$

Get the image of an inverse diagram

$$X : I \to \mathrm{Type}$$

by taking Σ_i.

i	M_i	Assume type-in-ctxt
0	$\cdot$	$\vdash \; A_0$
1	$x : A_0$	$\vdash \; A_1$
2	$x : A_0,$ $y : A_1(x),$ $z : A_1(x)$	$\vdash \; A_2$

$$
\begin{tikzcd}
2 \arrow[dr, shift left] \arrow[dr, shift right] \arrow[ddr, bend right] & & \\
& 1 \arrow[d] & \\
& 0 &
\end{tikzcd}
\xrightarrow{\quad X \quad}
\begin{tikzcd}
X_2 = \sum (x:A_0)(y:A_1(x))(z:A_1(x)).\, A_2(x,y,z) \arrow[d, shift left, "(x,y,z,-)\mapsto(x,z)"] \arrow[d, shift right, "(x,y,z,-)\mapsto(x,y)"'] \arrow[dd, bend right, "\pi_1"'] \\
X_1 = \sum_{A_0} A_1 \arrow[d, "\pi_1"] \\
X_0 = A_0
\end{tikzcd}
$$

i	M_i	Assume type-in-ctxt
0	$\cdot$	$\vdash \quad A_0$
1	$x : A_0$	$\vdash \quad A_1$
2	$x : A_0,$ $y : A_1(x),$ $z : A_1(x)$	$\vdash \quad A_2$

Matching objects (contexts) of "Reedy fibrant" X

$$X_2 = \sum (x : A_0)(y : A_1(x))(z : A_1(x)).\ A_2(x,y,z)$$

$$\begin{tikzcd}
2 \arrow[d, shift left] \arrow[d, shift right] \arrow[dd, bend right] & & X_2 \arrow[d, shift left] \arrow[d, shift right] \arrow[dd, bend right, "\pi_1"'] \\
1 \arrow[d] & \xrightarrow{\quad X \quad} & X_1 = \sum_{A_0} A_1 \arrow[d, "\pi_1"] \\
0 & & X_0 = A_0
\end{tikzcd}$$

$(x,y,z,-) \mapsto (x,y)$

$(x,y,z,-) \mapsto (x,z)$

Inverse diagrams

$$X : I \longrightarrow \mathcal{C}$$

Frequently:

- (Classical) mathematical metatheory
- $\mathcal{C} = \mathcal{S}et$.

We want:

- Homotopical type theory (incl. MLTT w/o UIP)
- $\mathcal{C}$ a wild category

  (i.e. precategory w/ hom-types)

Inverse diagrams

$$X : I \longrightarrow \mathcal{C}$$

Frequently:

- (Classical) mathematical metatheory
- $\mathcal{C} = \mathrm{Set}$.

We want:

- Homotopical type theory (incl. MLTT w/o UIP)
- $\mathcal{C}$ a wild category

  (e.g. $\mathcal{C} = \mathcal{U}$, wild category of $\mathcal{U}$-small types & functions)

Why?

Internalize!

- ⇝ Internal metatheory of TT w/o UIP
- ⇝ Higher categorical structures in HoTT

Technical goal

Given "nice enough"

- simple inverse category $\mathcal{I}$
- wild category $\mathcal{C}$ w/ a $(\Pi, \mathcal{U})$-cwf structure (*)

in HoTT,

inductively define matching objects M_i
of diagrams $\mathcal{I} \to \mathcal{C}$.

⚙ Technical goal ⚙

Given "nice enough"

- simple inverse category $\mathcal{I}$ (strictly oriented)
- wild category $\mathcal{C}$ w/ a $(\Pi, \mathcal{U})$-cwf structure
  $^{(*)}$
  (... set-truncated? Uniformly coherent?)

in HoTT,

inductively define matching objects M_i of diagrams $\mathcal{I} \to \mathcal{C}$.

M_i are indexed over $i/\mathbb{I}$,

which has a filtration by linear cosieves

2

1

0

i	M_i	Assume
0	$\cdot$	$\vdash A_0$
1	$x : A_0$	$\vdash A_1$
2		

M_i are indexed over $i/\mathbb{I}$,

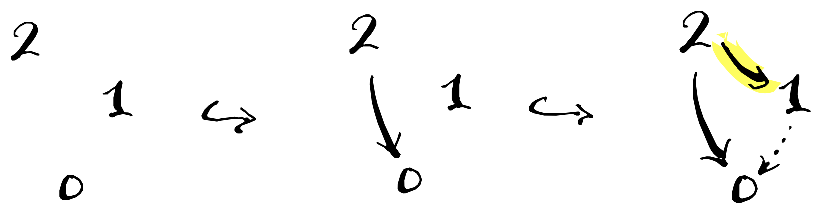
which has a filtration by linear cosieves

$$
\begin{tikzcd}[column sep=small, row sep=small]
2 & & & & 2 \arrow[dd] & \\
& 1 & \hookrightarrow & & & 1 \\
0 & & & & 0 &
\end{tikzcd}
$$

i	M_i	Assume
0	$\cdot$	$\vdash A_0$
1	$x : A_0$	$\vdash A_1$
2	$x : A_0$	

M_i are indexed over i/I,

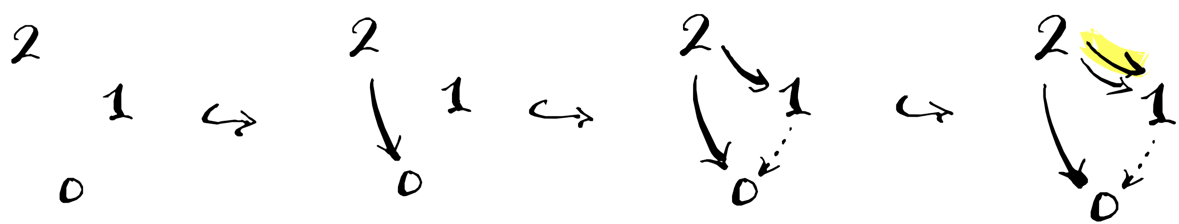
which has a filtration by linear cosieves



i	M_i	Assume
0	$\cdot$	$\vdash A_0$
1	$x : A_0$	$\vdash A_1$
2	$x : A_0$, $y : A_1(x)$	

M_i are indexed over i/I ,

which has a filtration by linear cosieves



i	M_i	Assume
0	$\cdot$	$\vdash A_0$
1	$x : A_0$	$\vdash A_1$
2	$x : A_0$, $y : A_1(x)$, $z : A_1(x)$	

Facts

- i/I has a filtration by linear cosieves

- Linear cosieves of strictly oriented I form a (strict) category $\mathcal{L}_I$ with a split bifibration $p : \mathcal{L}_I \longrightarrow I$

$$\begin{tikzcd} \mathcal{L}_I \arrow[d, "p"'] & (i, S, b) \arrow[r, "\bar{f}"] \arrow[d, mapsto] & (j, T, c) \arrow[d, mapsto] \\ I & i \arrow[r, "f"] & j \end{tikzcd}$$

$b \geq \text{height } S$
$c \geq \text{height } T$

$$\bar{f} = \begin{Bmatrix} f : I(i,j) \\ p : T \subseteq S \cdot f \\ u : b \geq c \end{Bmatrix}$$

Intuition:

Matching objects of diagrams $I \to \mathcal{C}$ should arise as "sufficiently coherent" wild functors $M : \mathcal{L}_I \longrightarrow \mathcal{C}$.

◊ Thm/Construction

(WIP, implementing in Agda)

github.com/jaycech3n/internal-diagrams

Given strictly oriented I and sufficiently coherent wild $\mathcal{C}$ w/ $(\Pi, \mathcal{U})$-cwf structure, there is a (strict) category Sh_I such that

1. Sh_I embeds into $\mathcal{L}_I$ via a functor $\imath$ which
   - is injective on objects
   - hits $(i, i/I, i-1) : (\mathcal{L}_I)_0$ for each $i : I_0$
   - sends arrows to opcartesian lifts of $p : \mathcal{L}_I \to I$
2. by induction on $(Sh_I)_0$ we simultaneously define
   - the type of a generic diagram $X : I \to \mathcal{C}$
   - a wild functor $\mathcal{M} : Sh_I \longrightarrow \mathcal{C}$ such that $\mathcal{M}[\imath^{-1}(i, i/I, i-1)]$ is the matching obj. M_i of X.

Current Questions

(?) "Sufficiently coherent" wild structure

- "Set-truncated" certainly suffices
- So far have not yet needed to truncate... at which point will we be forced to?

(?) Correct induction principle on $(Sh_1)_0$

- Initial mutually inductive def. not structurally decreasing (not accepted by Agda).
- Related to how we choose to define

$$Sh_1 \,,\; \mathcal{L}_1 \text{ and } \iota : Sh_1 \hookrightarrow \mathcal{L}_1 .$$

Appendix

Def. $\mathcal{I}$ simple cat is strictly oriented if:

- $\mathcal{I}(i,j)$ is ordered $(<)$ for all i,j.
- For $f : \mathcal{I}(i,j)$,

$$- \circ f : \mathcal{I}(j,k) \to \mathcal{I}(i,k)$$

  is strictly monotone.

Def. The height of a cosieve S under $i : \mathcal{I}_0$ in $\mathcal{I}$ is the largest $h : \mathcal{I}_0$ such that

$$S \cap \mathcal{I}(i,h)$$

is inhabited.

Def. A cosieve S under i of height h is linear if :

- For all $j : \mathbb{I}_0$ where $j < h$,

  $$\mathbb{I}(i,j) \subset S.$$

- $S \cap \mathbb{I}(i,h)$ is a $<$-prefix of $\mathbb{I}(i,h)$.