A parametricity-based construction of semi-simplicial and semi-cubical sets

Hugo Herbelin

joint work with Ramkumar Ramachandra

LHC 2023

Outline

- Iterating **unary** or **binary** parametricity respectively build **augmented semi-simplicial** and **semi-cubical** sets

- "**Fibred**" vs "**indexed**" presheaves over a direct category (and a word on "Reedy fibrancy")

- An **effective** iterated-parametricity-based "indexed" construction (formalised in Coq)

- Discussion and conclusions

From unary and binary parametricity
to
augmented semi-simplicial and semi-cubical sets

Unary and binary parametricity

For a given typed language, parametricity associates to each type an **observational** characterisation of the behaviour of its inhabitants. E.g.:

Binary parametricity (a form of bisimilarity)

To X : **HSet**, associate $X_\star : X \times X \to \textsf{HSet}$

Example: $f_0 =_{X\to Y} f_1 \quad \triangleq \quad \Pi x_0 x_1 \in X.\, (x_0 =_X x_1 \to f_0\, x_0 =_Y f_1\, x_1)$

Example: $X_0 =_{\textsf{HSet}} X_1 \quad \triangleq \quad X_0 \times X_1 \to \textsf{HSet}$

Unary parametricity (a form of realisability)

To X : **HSet**, associate $X_\star : X \to \textsf{HSet}$

Example : $f \in (X \to Y)_\star \quad \triangleq \quad \Pi x \in X.\, f\, x \in Y$

Example : $X \in \textsf{HSet}_\star \quad \triangleq \quad X \to \textsf{HSet}$

(where, in type theory, which by default is $(\infty, 1)$-categorical, **HSet** is the subtype of "discrete" **Type**, i.e. with proof-irrelevant equality)

Parametricity can be iterated

binary case

$$X \quad : \mathsf{HSet}$$
$$X_\star \quad : X \times X \to \mathsf{HSet}$$
$$X_{\star\star} : \left| \begin{array}{ccc} x : X & \xrightarrow{r : X_\star(x,z)} & z : X \\ p : X_\star(x,y) & & q : X_\star(z,w) \\ y : X & \xrightarrow{s : X_\star(y,w)} & w : X \end{array} \right| \to \mathsf{HSet}$$

$$\ldots$$

unary case

$$X \quad : \mathsf{HSet}$$
$$X_\star \quad : X \to \mathsf{HSet}$$
$$X_{\star\star} \quad : \Pi o : X . (X_\star(o) \times X_\star(o)) \to \mathsf{HSet}$$
$$X_{\star\star\star} : \Pi o : X . \begin{array}{ccc} & a : X_\star(o) & \\ p : X_{\star\star}(o,x,y) & & q : X_{\star\star}(o,x,z) \\ y : X_\star(o) & \xrightarrow{r : X_{\star\star}(o,y,z)} & z : X_\star(o) \end{array} \to \mathsf{HSet}$$

$$\ldots$$

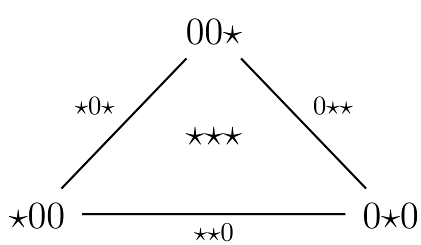
This eventually gives rise to a semi-cubical set structure (binary case) or augmented semi-simplicial set structure (unary case, as reported to us by Altenkirch and Moeneclaey).

The common presentation of augmented semi-simplicial and semi-cubical sets

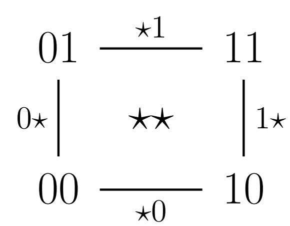
For ν a positive integer, we define the ν-**semi-shape category** to be:

$$
\begin{array}{lll}
\textsf{Obj} & \triangleq & \mathbb{N} \\
\textsf{Hom}(p,n) & \triangleq & \{l \in ([0,\nu) \cup \{\star\})^n \mid \text{number of } \star \text{ in } l = p\} \\
g \circ f & \triangleq & \begin{cases} f & \text{if } g = [] \\ w :: (g' \circ f) & \text{if } g = (w :: g')\text{, where } w \in [0,\nu) \\ a :: (g' \circ f') & \text{if } g = (\star :: g'),\ f = (a :: f')\text{, where } a \in [0,\nu) \cup \{\star\} \end{cases} \\
\textbf{id} & \triangleq & [\star, \ldots, \star]\ n \text{ times for } \textbf{id} \in \textsf{Hom}(n,n)
\end{array}
$$

It defines either the augmented semi-simplicial category $(\nu = 1)$ or the semi-cubical category $(\nu = 2)$



(standard augmented 3-semi-simplex)



(standard 2-semi-cube)

Fibred vs indexed representation of dependencies

Two standard ways to represent a dependent type over a type B

- the dependent "indexed" way (as a family):

$$P : B \to \mathsf{Type}$$

- the fibred way (as in "fibrational" categorical models):

$$(A, f) : \Sigma A : \mathsf{Type}.\,(A \to B)$$

A fundamental equivalence

$$B \to \mathsf{Type} \quad \simeq \quad \Sigma A : \mathsf{Type}.\,(A \to B)$$

$$\begin{array}{rcl} P & \mapsto & (\overbrace{\Sigma b : B.\, P\, b}^{\text{total space of } P}, \pi_1) \\ \lambda b : B.\, \underbrace{\Sigma a : A.\,(f\,a = b)}_{\text{fibre of } b} & \leftarrow\!\shortmid & (A, f) \end{array}$$

Proof relies on univalence (or even without if $\simeq$ is defined in an enough extensional way)

Composing those is not definitional/strict! The true (suspected) cause of Curien-Garner-Hofmann's mismatch between MLTT and LCCC!

Fibred vs indexed parametricity

Consequently, two (conjecturally) equivalent characterisations of the assignment to a type of an iterated family of (relevant) predicates/relations over this type:

iterated fibred	*iterated indexed*
$\vdots$	$\vdots$
$X_{\star\star} : \mathsf{HSet}$	$X_{\star\star} : \Pi xy : X, X_\star(x,y) \to \Pi zw : X.\, X_\star(z,w) \to X_\star(x,z) \times X_\star(y,w) \to \mathsf{HSet}$
$\downarrow\downarrow\downarrow\downarrow$	
$X_\star : \mathsf{HSet}$	$X_\star : X \times X \to \mathsf{HSet}$
$\downarrow\downarrow$	
$X : \mathsf{HSet}$	$X : \mathsf{HSet}$

+ *coherence conditions*

For connections between the two approaches, see e.g. Atkey-Johann-Ghani 2014, Bernardy-Coquand-Moulin 2014, Altenkirch-Kaposi 2015, Tabareau-Tanter-Sozeau 2018, Johann-Sojokova 2017, ...

Correspondingly: Fibred vs indexed representation of augmented semi-simplicial and cubical sets

- Any presheaf over a direct categorical (i.e. with "well-founded face maps") can (a priori) be represented in the indexed way (by induction).

- The indexed and fibred representations do not have the same properties!

- Instantiating an indexed representation requires giving only the sets, not the maps.
- Conversely, indexed presheaves are more difficult to define: the coherence conditions are hard-wired in the definition and proved (once for all) at definition time.
- As models of cubical type theory (which is one of the motivations), indexed presheaves satisfy definitional rules which fibred presheaves (a priori) do not.

In categorical terms, an indexed presheaf is indexed over the "matching object" (i.e. the collection of all faces of lower dimensions).

The image of indexed presheaves within fibred presheaves through the equivalence is called "Reedy fibrant".

Any indexed definition requires a specific choice of definition of a "matching object". This talk reports about a presentation based on parametricity.

Effective construction of augmented semi-simplicial or cubical sets

Iterated meta-induction vs direct construction

Conceptually, iterated parametricity is obtained by iterating the parametricity translation starting from some X : HSet.

$$
\begin{array}{lll}
X & : \mathsf{HSet} & \\
X_\star & : \mathsf{HSet}_\star(X)(X) & (\equiv X \times X \to \mathsf{HSet}) \\
X_{\star\star} & : (\mathsf{HSet}_\star(X)(X))_\star(X_\star)(X_\star) & (\text{applying parametricity rules for } \times, \to \text{ and } \mathsf{HSet}) \\
\ldots & &
\end{array}
$$

This is a *(meta-)induction* on the syntax of types!

How to turn it into a regular induction on the dimension?

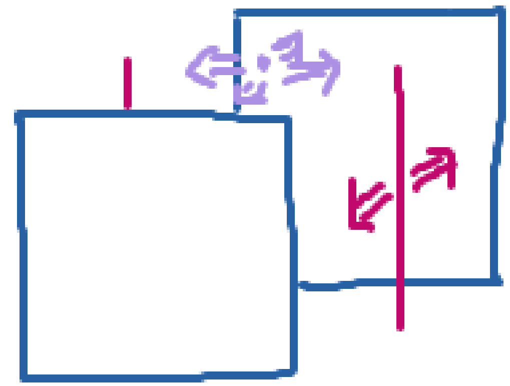
Analysing the recursive structure of types at dimension n

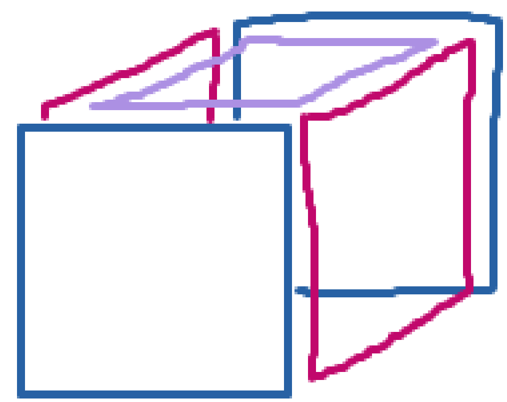
$$X_0 : \underbrace{\mathsf{unit}}_{\mathsf{frame}^{0,0}} \to \mathsf{HSet}$$

$$X_1 : \underbrace{\Sigma * : \mathsf{unit}.\ \underbrace{\begin{pmatrix} \underbrace{X_0(*)}_{\mathsf{painting}^{0,0}} \\ \times \\ \underbrace{X_0(*)}_{\mathsf{painting}^{0,0}} \end{pmatrix}}_{\mathsf{layer}^{1,0}}}_{\mathsf{frame}^{1,1}} \to \mathsf{HSet}$$

$$X_2 : \underbrace{\Sigma a : \underbrace{\left(\Sigma * : \mathsf{unit}.\ \underbrace{\begin{pmatrix} \underbrace{\Sigma b : \begin{pmatrix} X_0(*) \\ \times \\ X_0(*) \end{pmatrix}.\ \underbrace{X_1\ \underbrace{(*, b)}_{\mathsf{restr}^{2,0}_{\mathsf{frame},L}}}_{\mathsf{painting}^{1,1}}}_{\mathsf{painting}^{1,0}} \\ \times \\ \underbrace{\Sigma b : \begin{pmatrix} X_0(*) \\ \times \\ X_0(*) \end{pmatrix}.\ \underbrace{X_1\ \underbrace{(*, b)}_{\mathsf{restr}^{2,0}_{\mathsf{frame},R}}}_{\mathsf{painting}^{1,1}}}_{\mathsf{painting}^{1,0}} \end{pmatrix}}_{\mathsf{layer}^{2,0}} \right)}_{\mathsf{frame}^{2,1}} .\ \underbrace{\begin{pmatrix} \underbrace{X_1\ \underbrace{\left(a.\mathsf{hd}, \begin{pmatrix} a.\mathsf{tl}.L.\mathsf{hd}.L, \\ a.\mathsf{tl}.R.\mathsf{hd}.L \end{pmatrix}\right)}_{\mathsf{restr}^{2,1}_{\mathsf{frame},L}}}_{\mathsf{painting}^{1,1}} \\ \times \\ \underbrace{X_1\ \underbrace{\left(a.\mathsf{hd}, \begin{pmatrix} a.\mathsf{tl}.L.\mathsf{hd}.R, \\ a.\mathsf{tl}.R.\mathsf{hd}.R \end{pmatrix}\right)}_{\mathsf{restr}^{2,1}_{\mathsf{frame},R}}}_{\mathsf{painting}^{1,1}} \end{pmatrix}}_{\mathsf{layer}^{2,1}}}_{\mathsf{frame}^{2,2}} \to \mathsf{HSet}$$

$\ldots$





A full formalisation

Require to simultaneously define the *matching objects*, *restrictions* between them, and *coherences* on composition of restrictions (no higher-order coherences because in **HSet**)

νSet_m		$:$	HSet_{m+1}
νSet_m		$\triangleq$	$\nu\text{Set}_m^{\geq 0}(*)$
$\nu\text{Set}_m^{\geq n}$	$(D : \nu\text{Set}_m^{<n})$	$:$	HSet_{m+1}
$\nu\text{Set}_m^{\geq n}$	D	$\triangleq$	$\Sigma R : \nu\text{Set}_m^{=n}(D).\nu\text{Set}_m^{\geq n+1}(D, R)$

Table 1. Main definition

$\nu\text{Set}_m^{<n}$		$:$	HSet_{m+1}
$\nu\text{Set}_m^{<0}$		$\triangleq$	unit
$\nu\text{Set}_m^{<n'+1}$		$\triangleq$	$\Sigma D : \nu\text{Set}_m^{<n'}. \nu\text{Set}_m^{=n'}(D)$
$\nu\text{Set}_m^{=n}$	$(D : \nu\text{Set}_m^{<n})$	$:$	HSet_m
$\nu\text{Set}_m^{=n}$	D	$\triangleq$	$\text{fullframe}_m^n(D) \to \text{HSet}_m$

Table 2. Truncated ν-sets, the core

fullframe^n	$(D : \nu\text{Set}_m^{<n})$	$:$	HSet_m
fullframe^n	D	$\triangleq$	$\text{frame}^{n,n}(D)$
$\text{frame}^{n,p,p\leq n}$	$(D : \nu\text{Set}_m^{<n})$	$:$	HSet_m
$\text{frame}^{n,0}$	D	$\triangleq$	unit
$\text{frame}^{n,p'+1}$	D	$\triangleq$	$\Sigma d : \text{frame}^{n,p'}(D). \text{layer}^{n,p'}(d)$
$\text{layer}^{n,p,p<n}$	$\{D : \nu\text{Set}_m^{<n}\}$ $(d : \text{frame}^{n,p}(D))$	$:$	HSet_m
$\text{layer}^{n,p}$	$D\ d$	$\triangleq$	$\Pi\omega.\text{painting}^{n-1,p}(D.2)(\text{restr}^{n,p}_{\text{frame},\omega,p}(d))$
$\text{painting}^{n,p,p\leq n}$	$(D : \nu\text{Set}_m^{<n})$ $(E : \nu\text{Set}_m^{=n}(D))$ $(d : \text{frame}^{n,p}(D))$	$:$	HSet_m
$\text{painting}^{n,p,p=n}$	$D\ E\ d$	$\triangleq$	$E(d)$
$\text{painting}^{n,p,p<n}$	$D\ E\ d$	$\triangleq$	$\Sigma l : \text{layer}^{n,p}(d). \text{painting}^{n,p+1}(E)(d, l)$

Table 3. frame, layer, and painting

$\text{restr}^{n,p,p\leq q\leq n-1}_{\text{frame},\epsilon,q}$	$\{D : \nu\text{Set}^{<n}\}$ $(d : \text{frame}^{n,p}(D))$	$:$	$\text{frame}^{n-1,p}(D.1)$
$\text{restr}^{n,0}_{\text{frame},\epsilon,q}$	$D\ *$	$\triangleq$	$*$
$\text{restr}^{n,p'+1}_{\text{frame},\epsilon,q}$	$D\ (d, l)$	$\triangleq$	$(\text{restr}^{n,p'}_{\text{frame},\epsilon,q}(d), \text{restr}^{n,p'}_{\text{layer},\epsilon,q-1}(l))$
$\text{restr}^{n,p,p\leq q\leq n-2}_{\text{layer},\epsilon,q}$	$\{D : \nu\text{Set}^{<n}\}$ $\{d : \text{frame}^{n,p}(D)\}$ $(l : \text{layer}^{n,p}(d))$	$:$	$\text{layer}^{n-1,p}(\text{restr}^{n,p}_{\text{frame},\epsilon,q+1}(d))$
$\text{restr}^{n,p}_{\text{layer},\epsilon,q}$	$D\ d\ l$	$\triangleq$	$\lambda\omega.(\text{restr}^{n-1,p}_{\text{painting},\epsilon,q}(D.2)(l_\omega))$
$\text{restr}^{n,p,p\leq q\leq n-1}_{\text{painting},\epsilon,q}$	$(D : \nu\text{Set}^{<n})$ $(E : \nu\text{Set}^{=n}(D))$ $(d : \text{frame}^{n,p}(D))$ $(c : \text{painting}^{n,p}(E)(d))$	$:$	$\text{painting}^{n-1,p}(D.2)(\text{restr}^{n,p}_{\text{frame},\epsilon,q+1}(d))$
$\text{restr}^{n,p,p=q}_{\text{painting},\epsilon,q}$	$D\ E\ d\ (l, \_)$	$\triangleq$	l_ϵ
$\text{restr}^{n,p,p<q}_{\text{painting},\epsilon,q}$	$D\ E\ d\ (l, c)$	$\triangleq$	$(\text{restr}^{n,p}_{\text{layer},\epsilon,q}(l), \text{restr}^{n,p+1}_{\text{painting},\epsilon,q}(E)(c))$

Table 4. q-th projection of restr, or faces

$\text{coh}^{n,p,p\leq r\leq q\leq n-2}_{\text{frame},\epsilon,\omega,q,r}$	$\{D : \nu\text{Set}^{<n}\}$ $(d : \text{frame}(D))$	$:$	$\text{restr}^{n-1,p}_{\text{frame},\epsilon,q}(\text{restr}^{n,p}_{\text{frame},\omega,r}(d)) = \text{restr}^{n-1,p}_{\text{frame},\omega,r}(\text{restr}^{n,p}_{\text{frame},\epsilon,q+1}(d))$
$\text{coh}^{n,0}_{\text{frame},\epsilon,\omega,q,r}$	$D\ *$	$\triangleq$	$\text{refl}(*)$
$\text{coh}^{n,p'+1}_{\text{frame},\epsilon,\omega,q,r}$	$D\ (d, l)$	$\triangleq$	$(\text{coh}^{n,p'}_{\text{frame},\epsilon,\omega,q,r}(d), \text{coh}^{n,p'}_{\text{layer},\epsilon,\omega,q,r}(l))$
$\text{coh}^{n,p,p<r\leq q\leq n-2}_{\text{layer},\epsilon,\omega,q,r}$	$(D : \nu\text{Set}^{<n})$ $\{d : \text{frame}(D)\}$ $(l : \text{layer}(d))$	$:$	$\text{restr}^{n-1,p}_{\text{layer},\epsilon,q}(\text{restr}^{n,p}_{\text{layer},\omega,r}(l)) = \text{restr}^{n-1,p}_{\text{layer},\omega,r}(\text{restr}^{n,p}_{\text{layer},\epsilon,q+1}(l))$
$\text{coh}^{n,p}_{\text{layer},\epsilon,\omega,q,r}$	$D\ d\ l$	$\triangleq$	$\lambda\theta. \text{coh}^{n-1,p}_{\text{painting},\epsilon,\omega,q-1,r-1}(D.2)(l_\theta)$
$\text{coh}^{n,p,p\leq r\leq q\leq n-2}_{\text{painting},\epsilon,\omega,q,r}$	$(D : \nu\text{Set}^{<n})$ $(E : \nu\text{Set}^{=n}(D))$ $(d : \text{frame}(D))$ $(c : \text{painting}(E)(d))$	$:$	$\text{restr}^{n-1,p}_{\text{painting},\epsilon,q}(D.2)(\text{restr}^{n,p}_{\text{painting},\omega,r}(E)(c)) = \text{restr}^{n-1,p}_{\text{painting},\omega,r}(D.2)(\text{restr}^{n,p}_{\text{painting},\epsilon,q+1}(E)(c))$
$\text{coh}^{n,p,p=r}_{\text{painting},\epsilon,\omega,q,r}$	$D\ E\ d\ (l, \_)$	$\triangleq$	$\text{refl}(\text{restr}^{n-1,p}_{\text{painting},\epsilon,q-1}(D.2)(l_\epsilon))$
$\text{coh}^{n,p,p<r}_{\text{painting},\epsilon,\omega,q,r}$	$D\ E\ d\ (l, c)$	$\triangleq$	$(\text{coh}^{n,p}_{\text{layer},\epsilon,\omega,q,r}(l), \text{coh}^{n,p+1}_{\text{painting},\epsilon,\omega,q,r}(E)(c))$

Table 5. Commutation of q-th projection and r-th projection, or coherence conditions

Fully formalised in Coq (sources at `github.com/artagnon/bonak`)

Comparison with other formalisations

- H.'s indexed semi-simplicial types 2013: based on face maps, complex combinatorics.

- Part-Luo's semi-simplicial types in logically-enriched type theory 2015: based on injective growing functions.

- Annenkov-Capriotti-Kraus' semi-simplicial types in two-level type theory 2017: based on injective growing functions which compose *definitionally*, thus cutting down the combinatorics.

- Chen-Kraus' semi-simplicial types in CwF 2021.

- This talk, parametricity-based: use Coq's strict propositions and "Yoneda trick" on inequalities to get more definitional equalities, complex induction.

Other related formalisations: Sozeau-Tabareau's groupoid model 2013, Tabareau-Tanter-Sozeau's 2-groupoid model 2013, Altenkirch-Boulier-Kaposi-Tabareau's setoid model 2019, Allioux-Finster-Sozeau's opetopic types 2021, ...

Conclusions and open questions

- A rather complex construction but the benefits of an indexed definition:
  - Provide models which interpret dependent types as dependent types, providing more definitional equalities (up to some limits, ongoing work of Sarah Reboullet). See also: Altenkirch-Kaposi 2014, Altenkirch-Kaposi-Shulman 2022.
- Suggest to further study fibred vs indexed presheaves.
- Suggest to rearticulate the traditional opposition between syntax and semantics around factual technical differences rather than cultural differences:
  - indexed is not specific to syntax and fibered to semantics,
  - any algebra can be presented as a free algebra,
  - intrinsic (without proof-terms) vs extrinsic (with proof-terms),
  - with specific choices of (co)limits vs up to isomorphism.
- Can the experience on turning meta-induction into regular induction be used to build semi-simplicial *types* (without **HSet** restriction, ongoing work of Moana Jubert, using Ara-Burroni-Guiraud-Malbos-Métayer-Mimram's tensoring with $\mathbb{O}_1$)?

Technical issues I

Strongly dependent construction where:

- level n requires defining a notion of border at level $n-1$,
- which depends on defining a notion of border restriction map from level $n-1$,
- which depends on a coherence condition between restrictions from levels $n-1$ and $n-2$.

We did not succeed to get full well-founded induction (untractable dependencies):

- instead, we worked on blocks of three levels $n-2$, $n-1$, n at once, that we stepwise slid.

Technical issues II

Strongly dependent construction, proofs of $p \le q$ occur in statements with antagonistic requirements:

- ability to do case analysis on inequality proofs,
- convenience of definitional proof irrelevance on inequality proofs.

For that purpose, we switch between two representations shown equivalent:

- a definitionally proof-relevant inductive formulation in **Type**,
- a definitionally proof-irrelevant definition obtained by a Yoneda construction (i.e. $n \le_{\mathcal{Y}} p \triangleq \Pi q.\, q \le n \to q \le p$) on top of a recursive definition of inequality in strict **Prop**.

An algebra of inequality proofs

In particular, we use three instances of the following algebra of inequality proofs:

$$
\begin{array}{lcll}
\textsf{contra} & : & 0 = n + 1 & \rightarrow \textsf{False} \\
\textsf{init} & : & & 0 \leq n \\
\diamond & : & & n \leq n \\
\updownarrow & : & n \leq m \wedge m \leq p & \rightarrow n \leq p \\
\uparrow & : & n \leq m & \rightarrow n \leq m + 1 \\
\downarrow & : & n + 1 \leq m & \rightarrow n \leq m \\
\Downarrow & : & n + 1 \leq m + 1 & \rightarrow n \leq m \\
\Uparrow & : & n \leq m & \rightarrow n + 1 \leq m + 1
\end{array}
$$

Incidentally suggests to canonically infer inequality proofs using type classes (but not attempted).