



HAL
open science

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

Hugo Herbelin, Ramkumar Ramachandra

► **To cite this version:**

Hugo Herbelin, Ramkumar Ramachandra. A parametricity-based formalization of semi-simplicial and semi-cubical sets. 2023. hal-03963929v1

HAL Id: hal-03963929

<https://inria.hal.science/hal-03963929v1>

Preprint submitted on 2 Mar 2023 (v1), last revised 31 Dec 2023 (v2)

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution - ShareAlike 4.0 International License

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

Hugo Herbelin
and Ramkumar Ramachandra

Université Paris Cité, Inria, CNRS, IRIF, Paris
Hugo.Herbelin@inria.fr and r@artagnon.com

Abstract—Constructions such as semi-simplicial and semi-cubical sets can be defined in the “usual way” as presheaves over respectively, the semi-simplex or semi-cube category, which we call *fibred* definitions, but also defined like in e.g. Voevodsky [1] or in previous work [2], as a dependently-typed construction, which we call *indexed*.

This paper describes a uniform indexed characterization of both augmented semi-simplicial and semi-cubical sets arising respectively as unary and binary iterated parametricity-based constructions.

Additionally, our construction is fully formalized in Coq’s dependent type theory.

I. INTRODUCTION

In the context of functional programming, Reynolds’ parametricity [3] interprets types as relations characterizing the observational behavior of programs of this type. Parametricity can be iterated, and it has been noted that iterated Reynolds’ parametricity, when using more generally graphs than relations, has a cubical flavor [4], [5], [6], [7], [8]. We obtain a *unary* variant of Reynolds’ *binary* parametricity by using predicates or families instead of relations or graphs, in which case, we obtain a form of realizability [9], [10], [6]. It has then been noted that iterated unary parametricity has an augmented simplicial flavor ^a.

We exploit the connection between iterated unary parametricity and augmented simplicial sets, and between iterated binary parametricity and cubical sets to present a uniform construction of both augmented semi-simplicial and semi-cubical sets obtained by applying simple rules characterizing iterated parametricity. In contrast to the standard presheaf presentation of augmented semi-simplicial and semi-cubical sets [11], [12], [13], our uniform construction, which we call *ν -sets*, is *indexed*. That is, instead of having the set of augmented semi-simplices or semi-cubes in dimension $n + 1$ *fibred* over the set of augmented semi-simplices or semi-cubes in dimension n , we consider families of augmented semi-simplices and semi-cubes indexed over their faces.

The outline of the paper is as follows. First, we recall the definition of the augmented semi-simplicial and

semi-cubical categories using a combinatorial presentation, that directly leads to our generalization to ν -sets. We then proceed to explain our indexed construction, with the help of diagrams. Before spelling out the details of our formalization in type-theoretic language, we have a section on reasoning with sets inside type theory. We end with some of the finer details of our mechanization.

Our precise mechanization in Coq is available at github.com/artagnon/bonak. The construction was conceived in Summer 2019, and the mechanization began in late 2019. A sketch of the construction was presented at the 2020 HoTT-UF workshop and the completion of the mechanization was reported at the TYPES 2022 conference.

II. SEMI-SIMPLICIAL AND SEMI-CUBICAL SETS

A. Augmented semi-simplicial sets

Augmented semi-simplicial sets are defined similarly to semi-simplicial sets, except that the connected components are additionally dependent on a “color”. Conversely, semi-simplicial sets can be seen as augmented semi-simplicial sets over a singleton set of colors.

Let us associate dimension 0 to colors; then, points are dimension 1, lines are dimension 2, and so on. There is hence a shift by one when compared to semi-simplicial sets. We can then draw augmented semi-simplicial sets like semi-simplicial sets, except for this shift by one.

While ordinary semi-simplicial sets are presheaves over the semi-simplex category $\hat{\Delta}$, augmented semi-simplicial sets are presheaves over, what we will define as, Δ_+ . There are different ways to define Δ_+ , up to equivalence, and we use a definition that will later straightforwardly extend to semi-cubical sets.

II-A.1 Notation [Finite sequences] We denote finite sequences by $\{i_1, \dots, i_n\}$ for i_j ranging over some domain. In particular, the empty sequence is written $\{\}$ and we define $i :: \{j_1, \dots, j_n\}$ to be $\{i, j_1, \dots, j_n\}$.

II-A.2 Definition [Δ_+] The definition of Δ_+ is shown in Fig. 1. Note that, if $g \circ f$ is well-defined, then the length of f is less than that of g . It can be shown that composition is associative and that id is neutral.

^aPrivate communication with Hugo Moeneclaey and Thorsten Altenkirch

$$\begin{aligned}
\text{Obj}(\Delta_+) &:= \mathbb{N} \\
\text{Hom}(p, n) &:= \{l \in \{0, \star\}^n \mid \text{number of } \star \text{ in } l = p\} \\
g \circ f &:= \begin{cases} f & \text{if } g = \{\} \\ 0 :: (g' \circ f) & \text{if } g = (0 :: g') \\ a :: (g' \circ f') & \text{if } g = (\star :: g'), f = (a :: f'), \text{ where } a = 0 \text{ or } \star \end{cases} \\
\text{id} &:= \{\star, \dots, \star\} \text{ } n \text{ times for } \text{id} \in \text{Hom}(n, n)
\end{aligned}$$

Fig. 1: Definition of Δ_+

II-A.3 Definition $[\text{Set}_{\Delta_+}]$ We define the category of augmented semi-simplicial sets as the functor category:

$$\text{Set}_{\Delta_+} := \text{Set}^{\Delta_+^{\text{op}}}$$

To provide examples, we define the standard augmented n -semi-simplex.

II-A.4 Definition $[\Delta_+^n]$ The standard augmented n -semi-simplex Δ_+^n is defined as what is called the Yoneda embedding of $n \in \text{Obj}(\Delta_+)$:

$$\begin{aligned}
\Delta_+^n &: \text{Set}_{\Delta_+} \\
\Delta_+^n(p) &:= \text{Hom}(p, n) \\
\Delta_+^n(f) &:= \lambda g. g \circ f
\end{aligned}$$

The standard augmented 0-semi-simplex is a singleton of one color. Standard augmented n -semi-simplices have a geometric interpretation, and we illustrate them for dimensions 1, 2, and 3.

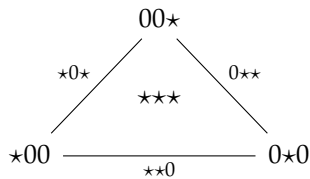
II-A.5 Example $[\Delta_+^1]$ In dimension 1, the standard augmented semi-simplex can be pictured as a point, colored black, corresponding to the unique morphism in $\text{Hom}(0, 1)$. This point is the identity in $\text{Hom}(1, 1)$; it is hence shown as a singleton \star .

\star

II-A.6 Example $[\Delta_+^2]$ In dimension 2, the standard augmented semi-simplex is drawn as two points, given by $\text{Hom}(1, 2)$, along with a line connecting them, given by $\text{Hom}(2, 2)$. We use the color black to denote the unique morphisms in $\text{Hom}(0, 1)$ and $\text{Hom}(0, 2)$.

$$\star 0 \text{ --- } \star\star \text{ --- } 0\star$$

II-A.7 Example $[\Delta_+^3]$ Δ_+^3 is drawn as three points, given by $\text{Hom}(1, 3)$, three lines connecting them, given by $\text{Hom}(2, 3)$, and a triangular filler given by $\text{Hom}(3, 3)$.



More generally, the standard augmented $(n + 1)$ -semi-simplex can be obtained by taking a copy of the standard augmented n -semi-simplex serving as a base and gluing on top of it another copy lifted by one dimension. In the second copy, the color becomes an extra point, the points become lines connecting the points of the base to the extra point and so on. In particular, the components of the base are those of the standard augmented n -semi-simplex postfixed by 0 while the components of the lifted copy are postfixed by \star .

B. Semi-cubical sets

Semi-cubical sets are defined like augmented semi-simplicial sets except that Δ_+ is replaced by \square in which we take sequences of L, R and \star , instead of sequences of 0 and \star .

II-B.1 Definition $[\square]$ The definition of \square is shown in Fig. 2. The symbols L and R indicate opposite faces of a cube.

Again, if $g \circ f$ is well-defined, then the length of f is less than that of g . It can be shown that composition is associative and that id is neutral.

II-B.2 Definition $[\text{Set}_{\square}]$ We define the category of semi-cubical sets as the functor category:

$$\text{Set}_{\square} := \text{Set}^{\square^{\text{op}}}$$

II-B.3 Definition $[\square^n]$ The standard semi-cube \square^n is defined as the Yoneda embedding of $n \in \text{Obj}(\square)$:

$$\begin{aligned}
\square^n &: \text{Set}_{\square} \\
\square^n(p) &:= \text{Hom}(p, n) \\
\square^n(f) &:= \lambda g. g \circ f
\end{aligned}$$

Standard n -semi-cubes have a geometric interpretation, which we illustrate for dimensions 0, 1, and 2.

II-B.4 Example $[\square^0]$ \square^0 is $\text{Hom}(0, 0)$, or the singleton set of the empty sequence:

$\{\}$

II-B.5 Example $[\square^1]$ \square^1 consists of two points, given by $\text{Hom}(0, 1)$ and a line, given by $\text{Hom}(1, 1)$.

$$L \text{ --- } \star \text{ --- } R$$

$$\begin{aligned}
\text{Obj}(\square) &:= \mathbb{N} \\
\text{Hom}(p, n) &:= \{l \in \{L, R, \star\}^n \mid \text{number of } \star \text{ in } l = p\} \\
g \circ f &:= \begin{cases} f & \text{if } g = \{\} \\ a :: (g' \circ f) & \text{if } g = (a :: g'), \text{ where } a = L \text{ or } R \\ a :: (g' \circ f') & \text{if } g = (\star :: g'), f = (a :: f'), \text{ where } a = L, R, \text{ or } \star \end{cases} \\
\text{id} &:= \{\star, \dots, \star\} \text{ } n \text{ times}
\end{aligned}$$

Fig. 2: Definition of \square

II-B.6 Example \square^2 consists of four points, given by $\text{Hom}(0, 2)$, four lines connecting the four points, given by $\text{Hom}(1, 2)$, and a filler, given by $\text{Hom}(2, 2)$:

$$\begin{array}{ccc}
LR & \xrightarrow{\star R} & RR \\
L\star & \Big| & \Big| R\star \\
& \star\star & \\
LL & \xrightarrow{\star L} & RL
\end{array}$$

More generally, the standard $(n+1)$ -semi-cube can be obtained by taking two copies of the standard n -semi-cube serving as bottom and top face and connecting them on their border by a cylinder obtained as a third copy stretched in the new dimension. The bottom and top faces are obtained from the standard n -semi-cube by postfixing with respectively L and R while the cylinder is obtained by postfixing with \star .

C. Generalization to ν -sets

Let us call ν -sets, the generalization of augmented semi-simplicial sets and semi-cubical sets. To obtain this, we extend Δ_+ and \square in a straightforward manner into a category which we call ν -semi-shape category. The morphisms of the ν -semi-shape category are sequences of elements of a set ν of arbitrary cardinal, extended with \star , so that the following holds.

Value of ν	1	2
Interpretation	Augmented semi-simplicial types	Semi-cubical types

A ν -set is thus a functor ϕ from the ν -semi-shape category to Set and we call n - ν -semi-shape an element of $\phi(n)$. As in the augmented semi-simplicial and semi-cubical cases, the standard $(n+1)$ - ν -semi-shape is obtained by connecting together ν copies of the standard n - ν -semi-shape with an extra copy stretched in the new dimension. We clarify in the next sections how this process of construction is similar to the parametricity translation developed for functional programming [3] and more generally for type theory [14], [15], [16], [17].

III. OUR SETTING

We now explain how we recover the notion of sets in type theory, explain the difference between the fibered and indexed representations, and supply an example of indexed semi-cubical sets.

A. Working with sets in type theory

Martin-Löf's Type theory [18], [19] is a logical formalism based on the notion of a *type* rather than that of a *set*. It can be seen as a foundation of mathematics alternative to set theory and is the core of several tools for the formalization of mathematics such as Agda [20], Coq [21] or Lean [22]. In type theory, propositions are types and proofs are programs. A particularity of type theory is also that types and programs, henceforth propositions and proofs also, are considered modulo an equational theory called definitional equality.

Type theory is a flexible formalism supporting different models. Some models are based on topological spaces, where equality is interpreted as path, and substitutivity of equality as transport. This model supports the univalence principle stating that equality of types mimics equivalence of types, leading to the development of Homotopy Type Theory [23]. In type theory, types are organized in a hierarchy of universes written Type_m for m a natural number. Main types in type theory are the types of dependent pairs, written $\Sigma a : A. B(a)$, the types of dependent functions, written $\Pi a : A. B(a)$, for A a type and $B(a)$ a type dependent on the inhabitant a of A , and the type of propositional equality, written $t = u$. We assume our type theory to also include a distinguished singleton type, written unit , and with inhabitant \star , the type of boolean values, called bool , and the type of natural numbers. We also write hd and tl the projections of dependent pairs, and refl for reflexivity. Logical propositions being types themselves, we use Π to represent universal quantification and Σ to represent existential quantification.

A notion of sets can be recovered in each universe as HSet_m , denoting the subtype of Type_m for which paths are degenerated, what can be expressed by the property of Uniqueness of Identity Proofs (UIP). Technically, this

is expressed as a structure equipping a domain Dom with the property UIP:

$$\begin{aligned} \text{Dom} &: \text{Type}_m \\ \text{UIP} &: \Pi xy : \text{Dom}. \Pi pq : x = y. p = q \end{aligned}$$

In HSet_m , the following properties hold:

- (i) UIP holds on the unit type, bool type, as well as all types of finite cardinal ν .
- (ii) UIP propagates to Σ -types.
- (iii) UIP propagates to Π -types, with some additional functional extensionality axioms.

Unless otherwise specified, we fix a universe level m and abbreviate Type_m as Type and HSet_m as HSet .

B. Fibered versus indexed representation

It is well-known in set theory that families of small sets can be represented in a fibered or indexed way. In the fibered way, a family over T is represented as a pair of a set S , and a function from S to T . The function classifies the original set into fibers. In contrast, in the indexed way, a family over T is represented as a functional relation with domain T , which itself, by the replacement axiom, is a set.

In category theory, the correspondence between the fibered and indexed way manifests as the correspondence between Grothendieck fibrations and pseudofunctors, of which a particular instance is the correspondence between discrete fibrations and presheaves, or even between discrete fibrations over sets and families of sets.

The latter correspondence can be expressed in Homotopy Type Theory, for $T : \text{Type}$, as an equivalence between $\Sigma S : \text{Type}. S \rightarrow T$ and $T \rightarrow \text{Type}$, which can be proved using univalence. This correspondence also holds on HSet , for $T : \text{HSet}$:

$$\Sigma S : \text{HSet}. S \rightarrow T \simeq T \rightarrow \text{HSet}$$

C. Example: indexed semi-cubical sets

Let us now come back to semi-cubical sets. A semi-cubical set can be represented as a sequence of fibrations, together with appropriate coherence conditions.

$$X_0 : \text{HSet} \begin{array}{l} \xleftarrow{\delta^R} \\ \xleftarrow{\delta^L} \end{array} X_1 : \text{HSet} \begin{array}{l} \xleftarrow{\delta^{*L}} \\ \xleftarrow{\delta^{*R}} \\ \xleftarrow{\delta^{R*}} \\ \xleftarrow{\delta^{L*}} \end{array} X_2 : \text{HSet} \quad \dots$$

If we think of X_1 fibered over two copies of X_0 in an indexed way, we get X_1 of type $X_0 \rightarrow X_0 \rightarrow \text{HSet}$. Continuing the process, we can think of X_2 fibered over four copies of X_1 themselves fibered over two copies of X_0 as a dependent type. We have a priori a dependency over eight copies of X_0 but these are actually four due to the coherence conditions. Based on these coherence

conditions, the types of X_0, X_1, X_2 happen to have the form:

$$\begin{aligned} X_0 &: \text{HSet} \\ X_1 &: X_0 \rightarrow X_0 \rightarrow \text{HSet} \\ X_2 &: \Pi abcd. X_1(a)(b) \rightarrow X_1(a)(c) \rightarrow X_1(b)(d) \\ &\quad \rightarrow X_1(c)(d) \rightarrow \text{HSet} \\ &\dots \end{aligned}$$

Now, one may wonder how to express the type of X_n uniformly as a function of n . Connecting semi-cubical sets to binary parametricity will give us a solution which we describe in the next section.

IV. OUR CONSTRUCTION

In this section, we gradually build up an intuition for our construction, followed by the details.

A. Relating the indexed representation to parametricity

The process of construction of the type of X_1 from that of X_0 , and from the type of X_2 to that of X_1 in the last section, is similar to applying a binary parametricity translation and expecting the resulting translation to be inhabited. The parametricity which we consider interprets a type A by a graph A_* over this type, and a term $t : A$ as an edge in $A_*(t)(t)$. In particular, HSet is interpreted as the graph HSet_* , which takes A_L and A_R in HSet and returns the type $A_L \rightarrow A_R \rightarrow \text{HSet}$ of graphs over A_L and A_R . Also, for A interpreted by A_* and B interpreted by B_* , a dependent function type $\Pi a : A. B$ is interpreted as the graph $(\Pi a : A. B)_*$ that takes two functions f_L and f_R of type $\Pi a : A. B$, and expresses that these functions map related arguments in A to related arguments in B :

$$\begin{aligned} (\Pi a : A. B)_*(f_L)(f_R) &\triangleq \\ \Pi a_L : A. \Pi a_R : A. \Pi a_* : A_*(a_L)(a_R). B_*(f_L(a_L))(f_R(a_R)) \end{aligned}$$

In particular, for $X : \text{HSet}$, applying the parametricity translation is about canonically associating to X an inhabitant X_* of $\text{HSet}_*(X)(X)$ i.e. of $X \rightarrow X \rightarrow \text{HSet}$. In turn, applying the parametricity translation to $X_* : X \rightarrow X \rightarrow \text{HSet}$ is about canonically associating to X_* an inhabitant X_{**} of $(X \rightarrow X \rightarrow \text{HSet})_*(X_*)(X_*)$ i.e. of:

$$\begin{aligned} \Pi x_{LL} : X. \Pi x_{LR} : X. X_*(x_{LL})(x_{LR}) \\ \rightarrow \Pi x_{RL} : X. \Pi x_{RR} : X. X_*(x_{RL})(x_{RR}) \\ \rightarrow X_*(x_{LL}, x_{RL}) \rightarrow X_*(x_{LR})(x_{RR}) \rightarrow \text{HSet} \end{aligned}$$

which hints us at how the sequence X_0, X_1, X_2 can be seen as a sequence of inhabitants of the iteration of binary parametricity applied to an initial $X : \text{HSet}$:

$$\begin{aligned} X_0 &\triangleq X &: & \text{HSet} \\ X_1 &\triangleq X_* &: & \text{HSet}_*(X)(X) \\ X_2 &\triangleq X_{**} &: & (\text{HSet}_*(X)(X))_*(X_*)(X_*) \\ &\dots && \end{aligned}$$

$$\begin{array}{l}
X_0 : \underbrace{\text{unit}}_{\text{frame}^{0,0}} \rightarrow \text{HSet} \\
X_1 : \Sigma * : \text{unit}. \left(\underbrace{\begin{array}{c} X_0(*) \\ \text{painting}^{0,0} \\ \times \\ X_0(*) \\ \text{painting}^{0,0} \end{array}}_{\text{layer}^{1,0}} \right) \rightarrow \text{HSet} \\
\phantom{X_1 : \Sigma * : \text{unit}.} \underbrace{}_{\text{frame}^{1,1}} \\
X_2 : \Sigma a : \Sigma * : \text{unit}. \left(\underbrace{\begin{array}{c} \Sigma b : \left(\begin{array}{c} X_0(*) \\ \times \\ X_0(*) \end{array} \right) \cdot X_1 \underbrace{(*, b)}_{\text{restr}_{\text{frame},L}^{2,0}} \\ \text{painting}^{1,1} \\ \times \\ \Sigma b : \left(\begin{array}{c} X_0(*) \\ \times \\ X_0(*) \end{array} \right) \cdot X_1 \underbrace{(*, b)}_{\text{restr}_{\text{frame},R}^{2,0}} \\ \text{painting}^{1,1} \\ \text{painting}^{1,0} \\ \text{layer}^{2,0} \end{array}}_{\text{frame}^{2,1}} \cdot \underbrace{\begin{array}{c} X_1 \left(a.\text{hd}, \left(\begin{array}{c} a.\text{tl}.L.\text{hd}.L, \\ a.\text{tl}.R.\text{hd}.L \end{array} \right) \right) \\ \text{restr}_{\text{frame},L}^{2,1} \\ \times \\ X_1 \left(a.\text{hd}, \left(\begin{array}{c} a.\text{tl}.L.\text{hd}.R, \\ a.\text{tl}.R.\text{hd}.R \end{array} \right) \right) \\ \text{restr}_{\text{frame},R}^{2,1} \\ \text{painting}^{1,1} \\ \text{layer}^{2,1} \end{array}}_{\text{layer}^{2,1}} \right) \rightarrow \text{HSet} \\
\phantom{X_2 : \Sigma a : \Sigma * : \text{unit}.} \underbrace{}_{\text{frame}^{2,2}} \\
\dots
\end{array}$$

Fig. 3: Intuition for formal construction ($\nu = 2$)

This tells us how the informal type given to X_2 in the previous section could be rephrased so that it comes as the instance of a general recipe characterizing the type of all X_i .

Notice, however, that the recipe obtained so far, $X_{n+1} : (S_n)_*(X_n)(X_n)$ for $X_n : S_n$, applies parametricity on the *syntax* of the type of X_n . It does not directly yield a characterization of S_n as a function from n . Reformulating the recipe as an explicit recursive construction, without requiring an interpretation of the syntax of types, is the main outcome of this work, together with the mechanization and the uniform treatment of augmented semi-simplicial and semi-cubical sets by means of the generalization to ν -sets.

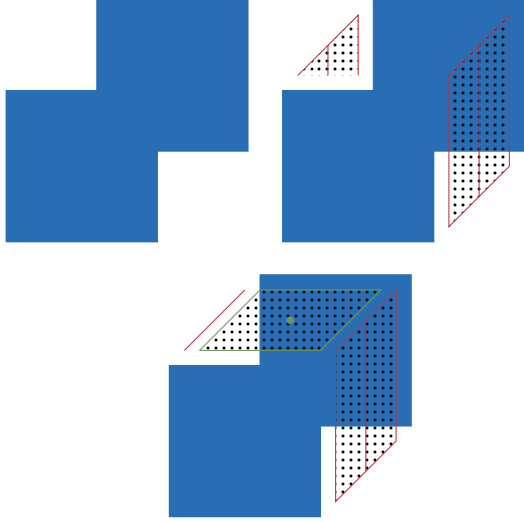
B. Intuition for our formal construction

To assign types to X_0, X_1, X_2, \dots in the above indexed representation, we give a recursive definition relying on building blocks, which we call **frame**, **layer**, and **painting**.

A **frame** is a boundary of a standard form (simplex, cube, etc.), which we decompose into **layer**, and a **painting** corresponds to a filled **frame**. Some **frame** are full and we call them **fullframe**.

We assign every X_n the type $\text{fullframe}^n \rightarrow \text{HSet}$ uniformly, applying to the description above the isomorphism between $A \rightarrow B \rightarrow C$ and $A \times B \rightarrow C$, or between $\Pi a : A. (Ba \rightarrow C)$ and $(\Sigma a : A. Ba) \rightarrow C$. Let us now illustrate how we build fullframe^n . To begin, let us set $\text{fullframe}^0 = \text{unit}$, so that the type HSet of X_0 can be equivalently formulated as $\text{unit} \rightarrow \text{HSet}$. Then, more generally, we let each fullframe^n consist of n layers, written $\text{layer}^{n,p}$ with $p < n$, that we stack in order, starting from the unit type and writing $\text{frame}^{n,p}$ for the p first layers of a fullframe^n , so that fullframe^n is $\text{frame}^{n,n}$. For instance, X_1 is made of one layer, so that it can be written as a Σ -type of an inhabitant of the unit and $\text{layer}^{1,0}$, as shown in Fig. 3. Fig. 3 also mentions how the type of X_2 is structured.

Let us now illustrate the construction of fullframe^3 , necessary to build the type of X_3 .



The figure on the left is $\text{frame}^{3,1}$, on the right is $\text{frame}^{3,2}$, and on the bottom is $\text{frame}^{3,3}$, which is full. Further, $\text{frame}^{3,1}$ is made of one layer, $\text{layer}^{3,0}$, shown in blue, $\text{frame}^{3,2}$ is made of one additional layer, $\text{layer}^{3,1}$, shown in red, $\text{frame}^{3,3}$ is made of one more layer, $\text{layer}^{3,2}$, shown in green.

We illustrated here the cubical case, that is $\nu = 2$, but, in general, a $\text{layer}^{n,p}$ is a product of ν $\text{painting}^{n-1,p}$. A $\text{painting}^{n,0}$ is a n -dimensional object corresponding to a filled fullframe^n . More generally, a $\text{painting}^{n,p}$ is an n -dimensional object which has the form of a $\text{painting}^{n-p,0}$, thus of $(n-p)$ -dimensional form, but shifted and living in dimensions p to n . Such $\text{painting}^{n,p}$ fills a space framed by a partial $\text{frame}^{n,p}$ so that, together, they form a filled fullframe^n . For instance, in the picture, each of the two $\text{painting}^{2,0}$ of $\text{layer}^{3,0}$ is a filled blue square, each of the two $\text{painting}^{2,1}$ of $\text{layer}^{3,1}$ is the line shown in red, stretched into a partial square filling the partial frames made of respectively the left and right border of the blue square, and each of the two $\text{painting}^{2,2}$ of $\text{layer}^{3,2}$ is a point shown in green, stretched into a partial square filling the full frames made respectively of the upper and lower borders of the blue and red squares. A $\text{painting}^{n,p}$ complements a $\text{frame}^{n,p}$ by adding layers needed to form a fullframe^n and by filling the resulting fullframe^n with an inhabitant of X_n . Layers are added from dimension n to dimension p that is opposite to the order from 0 to p the $\text{frame}^{n,p}$ are built, as shown below.

$$\begin{aligned} \text{frame}^{n,p} &\triangleq \Sigma a_n : (\dots (\Sigma * : \text{unit. layer}^{n,0}) \dots). \text{layer}^{n,p} \\ \text{painting}^{n,p} &\triangleq \Sigma l_p : \text{layer}^{n,p}. (\dots (\Sigma l_n : \text{layer}^{n,n-1}. X_n) \dots) \end{aligned}$$

So far, we have not paid attention to the fact that we have a dependent type, shown as a Σ . Let us be more precise about this requirement. First, fullframe^n depends on all X_i up to $n-1$. So, we need to package up X_i , for

$i < n$, into a type, which we call $\nu\text{Set}^{<n}$. This allows us to give the type $\nu\text{Set}^{<n} \rightarrow \text{HSet}$ to fullframe^n . Then, for $D : \nu\text{Set}^{<n}$, representing an initial prefix of X_0, X_1, \dots, X_{n-1} , the indexed set X_n has type $\text{fullframe}^n(D) \rightarrow \text{HSet}$. Thus, $\text{frame}^{n,p}$, $\text{layer}^{n,p}$ and $\text{painting}^{n,p}$ also depend on D . We can then refine the previous equation by showing the dependencies on D . In particular, X_n is just $D.\text{tl}$.

$$\begin{aligned} \text{frame}^{n,p}(D) &\triangleq \\ \Sigma a_n : (\dots (\Sigma * : \text{unit. layer}^{n,0}(D)) \dots). \text{layer}^{n,p}(D) \\ \text{painting}^{n,p}(D) &\triangleq \\ \Sigma l_p : \text{layer}^{n,p}(D). (\dots (\Sigma l_n : \text{layer}^{n,n-1}(D). D.\text{tl}) \dots) \end{aligned}$$

An extra refinement arises from the fact that each addition of a layer to a frame has to be glued onto the border of the partial frame built so far. So, each $\text{layer}^{n,p}$ has to depend on $\text{frame}^{n,p}$. We also need a way to characterize the border of the ν $\text{painting}^{n-1,p}$ that compose a $\text{layer}^{n,p}$, and this is $\text{restr}_{\text{frame},\epsilon,p}^{n,p}$ for all $\epsilon < \nu$. For instance, on the picture, the left and right $\text{painting}^{2,1}$, shown in red, are laid on respectively the left and right borders of the blue boxes, and hence needs to depend on $\text{frame}^{3,1}$. The left and right borders of the two blue boxes are extracted as $\text{restr}_{\text{frame},L}^{2,1}(D)(d)$ and $\text{restr}_{\text{frame},R}^{2,1}(D)(d)$. We can then refine the previous equation by showing the dependencies on d , as shown in Fig. 4.

When $\nu = 2$, the formation of layers from paintings amounts to:

$$\text{layer}^{n,p}(D)(d) \triangleq \text{painting}^{n-1,p}(D.\text{hd})(\text{restr}_{\text{frame},L,p}^{n,p}(d)) \times \text{painting}^{n-1,p}(D.\text{hd})(\text{restr}_{\text{frame},R,p}^{n,p}(d))$$

The operation $\text{restr}_{\text{frame},\epsilon,q}^{n,p}$ indicates restriction on q layers of a frame, and the induction is on p , from 0 to q . In particular, $\text{restr}_{\text{frame},\epsilon,p}^{n,p}$ is a "full restriction". We define $\text{restr}_{\text{frame},\epsilon,p}^{n,p}(d)$ by recursion on the structure of a frame d , which necessitates definitions of $\text{restr}_{\text{layer},\epsilon,q}^{n,p}(d)(l)$ and $\text{restr}_{\text{painting},\epsilon,q}^{n,p}(d)(c)$, for l a layer and c a painting. The key case is $\text{restr}_{\text{painting},\epsilon,p}^{n,p}(d)(c)$, where c , a $\text{painting}^{n,p}$, has necessarily the form of $((c_L, c_R), _)$: $\text{restr}_{\text{painting},L,p}^{n,p}$ picks out c_L , a $\text{painting}^{n-1,p}$, $\text{restr}_{\text{painting},R,p}^{n,p}$ picks out the c_R , also a $\text{painting}^{n-1,p}$, and $_$ a $\text{painting}^{n,p+1}$, is discarded. There is one more difficulty, which we illustrate by writing down expected and actual types.

Given c_ω of type

$$c_\omega : \text{painting}^{n-1,p}(D.\text{hd})(\text{restr}_{\text{frame},\omega,q}^{n-1,p}(d))$$

$\text{restr}_{\text{layer},\epsilon}^{n,p}(d)(c_L, c_R)$ produces a layer, in which the ω -component has the type

$$\text{painting}^{n-2,p}(D.\text{hd})(\text{restr}_{\text{frame},\epsilon,q}^{n-1,p}(\text{restr}_{\text{frame},\omega,p}^{n,p}(d)))$$

$$\begin{aligned}
\text{frame}^{n,p}(D) &\triangleq & \Sigma d : (\dots (\Sigma * : \text{unit. layer}^{n,0}(D)(*) \dots). \text{layer}^{n,p}(D)(d)) \\
\text{painting}^{n,p}(D)(d) &\triangleq & \Sigma l_p : \text{layer}^{n,p}(D)(d). (\dots (\Sigma l_n : \text{layer}^{n,n-1}(D)(d, l_p, \dots, l_{n-1}). D.\text{tl}(d, l_p, \dots, l_n)) \dots) \\
&& \text{where } (d, l_p, \dots, l_q) \text{ abbreviates } ((\dots (d, l_p), \dots), l_q)
\end{aligned}$$

Fig. 4: Refinement of definition of `frame` and `painting`

while we expect a term of type

$$\text{painting}^{n-2,p}(D.\text{hd}.\text{hd})(\text{restr}_{\text{frame},\omega,p}^{n-1,p}(\text{restr}_{\text{frame},\epsilon,\eta+1}^{n,p}(d)))$$

Hence, we need a coherence condition to commute the restrictions. Coherence conditions similar to this necessitate, what are shown as, $\text{coh}_{\text{frame}}$, $\text{coh}_{\text{layer}}$ and $\text{coh}_{\text{painting}}$ in tables in the next section. These are by induction on the structure of `frame`, `layer` and `painting`. Note that, for the construction in IV-D, we further need a 2-dimensional coherence condition, $\text{coh2}_{\text{frame}}$, for $\text{coh}_{\text{layer}}$.

C. Formal construction in ETT

We now present the construction in *extensional* type theory; i.e. in a type theory with the following reflection rule, where $=$ is propositional equality in some type and \equiv is definitional equality [19]:

$$\frac{\Gamma \vdash p : t = u}{\Gamma \vdash t \equiv u}$$

For the presentation corresponding to the formalization in *intensional* type theory, see IV-D.

The definition is dispatched over tables I, II, III, IV and V. Table I describes a ν -set in indexed form, as a coinductive limit of n -truncated ν -sets. The n -truncated ν -sets are themselves described in II. The n th component is a family over a `fullframe`. The type `fullframe` is recursively defined in III, using the auxiliary definitions of `layer` and `painting`. The type `layer` and `painting` are dependent over $\text{restr}_{\text{frame}}$, and these restrictions are defined on IV. These restrictions are defined using auxiliary definitions of $\text{restr}_{\text{layer}}$ and $\text{restr}_{\text{painting}}$.

Notably, the definition of $\text{restr}_{\text{layer}}$ relies on a definitional equality expressing the commutation of the composition of $\text{restr}_{\text{frame}}$. This commutation is not provable by computation so we have to prove it propositionally before using the reflection rule. Proving this itself requires an induction on the dimension, and on the structure of `frame`, `layer`, and `painting`. This is what $\text{coh}_{\text{frame}}$ proves, as shown in the table V, using auxiliary definitions $\text{coh}_{\text{layer}}$ and $\text{coh}_{\text{painting}}$. Even though it looks independent of the other tables, $\text{coh}_{\text{frame}}$ has to be proved mutually with the definitions of `frame`, `layer`, `painting`, and their corresponding restrictions. More precisely, for a fixed n , the block of `frame`, $\text{restr}_{\text{frame}}$, and $\text{coh}_{\text{frame}}$ has to be mutually defined by induction on p . Also, each of `painting`, $\text{restr}_{\text{painting}}$, and $\text{coh}_{\text{painting}}$ is built

by induction from p to n . The `painting` block at n relies on the `frame` block at n , but, the converse dependency is only on lower n , so this is well-founded. Note that `layer`, $\text{restr}_{\text{layer}}$ and $\text{coh}_{\text{layer}}$ are just abbreviations. We leave however implicit the exact way this mutual recursion can be formalized as this stage.

Most components of the construction takes inequality constraints as parameters, and we have left implicit that they are satisfied in the tables. Comparison of inequality proofs depends on the definition of inequality, which we leave implicit, only assuming that we can pick a definition for which proofs of $m \leq n$ are provably unique and thus definitionally equal by the reflection rule.

The construction takes benefit of various provable equalities over proofs of equality being definitional by the reflection rule. This includes in particular the groupoid properties of equality. Notably, uniqueness of identity proofs holds in extensional type theory, so that any type is automatically an `HSet`. Also, we left implicit in table V the use of the isomorphism between $u = v$ and $\Sigma(p : u.\text{hd} = v.\text{hd}).(u.\text{tl} = v.\text{tl})$ for u and v in a Σ -type. In the same table, we also left implicit the use of the isomorphism between $f = g$ and $\Pi a : A. f(a) = g(a)$ for f and g in $\Pi a : A. B$, where it should be recalled that the right-to-left map, that is function extensionality, holds by default in extensional type theory.

Note that for a fixed constant n , the coherence conditions evaluate to a reflexivity proof, so that the construction evaluates to an effective sequence of types of iterated relations not mentioning them anymore. For instance, it produces figure 3 for $n = 2$ and $\nu = 2$.

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

TABLE I: Coinductive structure

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

TABLE II: Truncated sets, the core

D. Formal construction in ITT

Since we have formalized the proof in the Coq proof assistant, which implements an intensional type theory, we now present the tables in *intensional* type theory [19], [21], leaving implicit some details, which we will explain here, and other details, which we will explain in IV-E.

Tables VI, VII, and VIII are unchanged, except that we make explicit all arguments, including those that can be inferred from the context. The need for transport along a proof of commutation of $\text{restr}_{\text{frame}}$ in the definition of $\text{restr}_{\text{layer}}$ is made explicit as shown in table IX, where the arrow over $\text{coh}_{\text{frame}}$ indicates the direction of rewrite. The proof of $\text{coh}_{\text{frame}}$ is described on X, and it requires making explicit several rewritings which were invisible in extensional type theory. The commutation of $\text{restr}_{\text{layer}}$ lives in a type referring to $\text{coh}_{\text{frame}}$, so we need a transport along the commutation of $\text{restr}_{\text{frame}}$ in the statement of $\text{coh}_{\text{layer}}$. The proof of $\text{coh}_{\text{layer}}$ is the most involved proof in the construction. This is where the higher-dimensional coherence condition is needed. The exact formulation of this coherence condition is as follows.

$$\begin{aligned} & \text{coh}_{\text{frame},\omega,\theta,r,p}^{n,p}(\text{restr}_{\text{frame},\epsilon,\eta+2}^{n,p}(d)) \bullet \\ & \text{ap } \text{restr}_{\text{frame},\omega,r}^{n,p}(\text{coh}_{\text{frame},\epsilon,\theta,\eta+1,p}^{n,p}(d)) \bullet \\ & \text{coh}_{\text{frame},\epsilon,\omega,\eta,r}^{n,p}(\text{restr}_{\text{frame},\theta,p}^{n,p}(d)) = \\ & \text{ap } \text{restr}_{\text{frame},\theta,p}^{n,p}(\text{coh}_{\text{frame},\epsilon,\omega,\eta+1,r+1}^{n,p}(d)) \bullet \\ & \text{coh}_{\text{frame},\epsilon,\theta,\eta,p}^{n,p}(\text{restr}_{\text{frame},\omega,r+1}^{n,p}(d)) \bullet \\ & \text{ap } \text{restr}_{\text{frame},\epsilon,\eta}^{n,p}(\text{coh}_{\text{frame},\omega,\theta,r,p}^{n,p}(d)) \end{aligned}$$

where ap applies a function on two sides of an equality, and \bullet is transitivity of equality. This property of equality proofs holds in HSet , and since our formalization is done in HSet , the term is trivially discharged.

Notice that each $\text{restr}_{\text{layer}}$ in the type of $\text{coh}_{\text{layer}}$ is hiding a $\text{coh}_{\text{frame}}$ rewrite: this makes a sum total of three $\text{coh}_{\text{frame}}$ rewrites on the left-hand side, and two $\text{coh}_{\text{frame}}$ rewrites on the right-hand side. In the proof term of $\text{coh}_{\text{layer}}$, $\text{coh}_{\text{painting}}$ has one $\text{coh}_{\text{frame}}$ rewrite on its left-hand side and zero on the right-hand side. This combined with the two terms of the form $\text{ap } \text{coh}_{\text{frame}}$ matches our expectation of three $\text{coh}_{\text{frame}}$ on the left-hand side and two $\text{coh}_{\text{frame}}$ on the right-hand side. Then, $\text{coh}_{\text{frame}}$ can roughly be seen as a commutation of these $\text{coh}_{\text{frame}}$ terms.

Finally, let us explain $\text{coh}_{\text{painting}}$. The base case $p = r$ is the key case of the commutation of $\text{restr}_{\text{frame}}$, when one of the $\text{restr}_{\text{painting}}$ collapses, and the remaining equation holds trivially. The case of $p < r$ follows the structure of $\text{restr}_{\text{painting}}$ by induction.

Like in V, we leave implicit in X, the use of the isomorphism between the type of equalities of dependent pairs in a Σ -type with the Σ -type of equalities of the

components of the pair, as well as the isomorphism between equality of functions with point-wise equality. Contrary to the ETT case, functional extensionality does not hold, so we assume it. However, the requirement of functional extensionality disappears if ν is finite. As for basic groupoid properties of equalities, which no longer hold definitionally, we do not mention them for simplicity.

The way recursion is implemented is still left implicit at this stage. See IV-E for the details.

Depending on how inequality on natural numbers is defined, different equalities may hold or not definitionally, such as transitivity of inequality or the computational properties of induction over inequality proofs. Details are given in IV-E.

Remark: If we were not working in HSet , but in HGpd , we would need to prove one more higher-dimensional coherence, and if we were working in Type , we would need to prove arbitrarily many higher-dimensional coherences. Here, HGpd is the subset of types A such that for all x and y in A , $x = y$ is in HSet . See [2], [24], [25] for a discussion on the need for recursive higher-dimensional coherence conditions in formulating higher-dimensional structures in type theory.

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

TABLE VI: Coinductive structure

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

TABLE VII: Truncated sets, the core

E. Details on the mechanization

Since the construction shown in the previous sections is by induction on n , and dependencies are on lower n and $p < n$, one would imagine formalizing this using well-founded induction in dependent type theory. We initially tried this approach: we had terms dependent on the proofs of the case distinction of $n' \leq n$ implies $n' < n$ or $n' = n$, and these proofs did not have definitional computational rule; it started to be necessary to reason propositionally on the computational property of the case distinction, and it eventually turned out to be unmanageable. Hence, we chose a different route: in practice, since $\text{restr}_{\text{frame}}^n$ depends on frame^n and

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

TABLE III: frame, layer, and painting

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

TABLE IV: q -th projection of restr, or faces

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

TABLE V: Commutation of q -th projection and r -th projection, or coherence conditions

frame^{n-1} , while $\text{coh}_{\text{frame}}^n$ depends on frame^n , frame^{n-1} , and frame^{n-2} , we only need to keep track of three “levels”, and we built separate data structures for the

levels, with dependencies. More concretely, we build the ten definitions shown in the tables by induction, and this is part of the definition of a larger record. The other fields

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

TABLE VIII: frame, layer, and painting

of the record are `frame`, `layer`, `painting` at levels $n - 1$ and $n - 2$, `restrframe`, `restrlayer`, and `restrpainting` at level $n - 1$, and equations to recall the definitions of these objects at lower levels.

The entire construction relies on inequalities over natural numbers, and we use two different definitions of \leq addressing different concerns in our formalization. In order to build our first variant, we present an intermediate “recursive definition”, phrased as:

Here `SProp` is a definitionally proof-irrelevant impredicative universe at the bottom of the universe hierarchy [26]. By placing the definition in `SProp`, we have definitional equality of all inequality proofs. For the purpose of unification, however, this definition does not go far enough. Consider the unification problems:

where `leR_trans` is transitivity, `leR_refl` is reflexivity, and `?p` is an existential variable. These two problems definitionally hold in `SProp`, but equalizing them does not solve the existential. For unification to be useful in inferring existentials, we present our first variant of \leq , which we dub as the “Yoneda variant”:

This definition is an improvement over `leR` since reflexivity is now definitionally the neutral element of transitivity, and associativity of transitivity also holds definitionally. Although it significantly eases our proof, there are some instances where unification is unable to solve the existentials, and we have to provide them explicitly.

The second variant of \leq , the “inductive variant”, is phrased as:

Compared to `leY`, `leI` has no proof-irrelevance properties. This definition is specially crafted for `painting`, where we have to reason inductively from $p \leq n$ to n . In our usage, we have lemmas `leY_of_leI` and `leI_of_leY` in order to equip `leY` with the induction scheme of `leI`. The resulting induction scheme has computational rules holding propositionally.

V. FUTURE WORK

In the cubical case, we expect the construction to eventually provide a model of (some version of) parametric type theory [27], [28] by adding degeneracies, a hierarchy of universes (as sketched e.g. in a talk by Herbelin at the HoTT-UF workshop for the bridge case [29]), as well as reasoning modulo permutations [12].

By equipping the universe construction with a structure of equivalences, as suggested along the lines of Altenkirch and Kaposi [5], we expect the construction to be able to serve as a basis for syntactic models of various versions of cubical type theory [30], [31], [32]. A key point is that, thanks to the indexed approach, we expect univalence to hold by definition. If these intuitions are correct, this would definitively root cubical type theory on top of iterated parametricity translation.

REFERENCES

- [1] V. Voevodsky, “Semi-simplicial types,” Nov 2012, shortly described in Section 8 of [2].
- [2] H. Herbelin, “A dependently-typed construction of semi-simplicial types,” *Mathematical Structures in Computer Science*, vol. 25, no. 5, pp. 1116–1131, 2015.
- [3] J. C. Reynolds, “Definitional interpreters for higher-order programming languages,” in *Proceedings of the ACM annual conference—Volume 2*, 1972, pp. 717–740.
- [4] P. Johann and K. Sojakova, “Cubical categories for higher-dimensional parametricity,” *CoRR*, vol. abs/1701.06244, 2017. [Online]. Available: <http://arxiv.org/abs/1701.06244>

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

TABLE IX: q -th projection of restr, or faces

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

TABLE X: Commutation of q -th projection and r -th projection, or coherence conditions

- [5] T. Altenkirch and A. Kaposi, "Towards a cubical type theory without an interval," in *21st International Conference on Types for Proofs and Programs, TYPES 2015, May 18-21, 2015, Tallinn, Estonia*, ser. LIPIcs, T. Uustalu, Ed., vol. 69. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2015, pp. 3:1–3:27. [Online]. Available: <https://doi.org/10.4230/LIPIcs.TYPES.2015.3>
- [6] G. Moulin, "Internalizing parametricity," Ph.D. dissertation, Department of Computer Science and Engineering, Chalmers University of Technology, 2016.
- [7] H. Moeneclaey, "Parametricity and semi-cubical types," in *36th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2021, Rome, Italy, June 29 - July 2, 2021*. IEEE, 2021, pp. 1–11. [Online]. Available: <https://doi.org/10.1109/LICS52264.2021.9470728>
- [8] —, "Cubical models are cofreely parametric," Ph.D. dissertation, Université Paris Cité, 2022.
- [9] J.-P. Bernardy and G. Moulin, "A computational interpretation of parametricity," in *2012 27th Annual IEEE Symposium on Logic in Computer Science*. IEEE, 2012, pp. 135–144.
- [10] M. Lasson, "Réalisation et paramétricité dans les systèmes de types purs," Ph.D. Thesis, Université de Lyon, Nov. 2014.
- [11] G. Friedman, "An elementary illustrated introduction to simplicial sets," *arXiv preprint arXiv:0809.4221*, 2008. [Online]. Available: <https://arxiv.org/abs/0809.4221v7>
- [12] M. Grandis and L. Mauri, "Cubical sets and their site," *Theory and Applications of Categories*, vol. 11, pp. 185–211, 2003.
- [13] U. Buchholtz and E. Morehouse, "Varieties of cubical sets," in *International Conference on Relational and Algebraic Methods in Computer Science*. Springer, 2017, pp. 77–92.
- [14] J. Bernardy, P. Jansson, and R. Paterson, "Parametricity and dependent types," in *Proceeding of the 15th ACM SIGPLAN international conference on Functional programming, ICFP 2010, Baltimore, Maryland, USA, September 27-29, 2010*, P. Hudak and S. Weirich, Eds. ACM, 2010, pp. 345–356. [Online]. Available: <http://doi.acm.org/10.1145/1863543.1863592>
- [15] J. Bernardy and M. Lasson, "Realizability and parametricity in pure type systems," in *Foundations of Software Science and Computational Structures - 14th International Conference, FOSSACS 2011, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2011, Saarbrücken, Germany, March 26-April 3, 2011. Proceedings*, ser. Lecture Notes in Computer Science, vol. 6604. Springer, 2011, pp. 108–122. [Online]. Available: https://doi.org/10.1007/978-3-642-19805-2_8
- [16] R. Atkey, N. Ghani, and P. Johann, "A relationally parametric model of dependent type theory," in *The 41st Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL '14, San Diego, CA, USA, January 20-21, 2014*, 2014.
- [17] J. Bernardy, T. Coquand, and G. Moulin, "A presheaf model of parametric type theory," *Electr. Notes Theor. Comput. Sci.*, vol. 319, pp. 67–82, 2015. [Online]. Available: <https://doi.org/10.1016/j.entcs.2015.12.006>
- [18] P. Martin-Löf, "An intuitionistic theory of types, predicative part," in *Logic Colloquium*. North Holland, 1975, pp. 73–118.
- [19] P. Martin-Löf, *Intuitionistic type theory*, ser. Studies in proof theory. Bibliopolis, 1984, vol. 1.
- [20] The Agda Team, *Agda User Manual, Release 2.6.4*, Jan 2023. [Online]. Available: <https://readthedocs.org/projects/agda/downloads/pdf/latest>
- [21] The Coq Development Team, *The Coq Reference Manual, version 8.16.1*, Oct 2022. [Online]. Available: <https://coq.inria.fr/distrib/current/refman/>
- [22] L. M. de Moura, S. Kong, J. Avigad, F. van Doorn, and J. von Raumer, "The lean theorem prover (system description)," in *Automated Deduction - CADE-25 - 25th International Conference on Automated Deduction, Berlin, Germany, August 1-7, 2015, Proceedings*, ser. Lecture Notes in Computer Science, A. P. Felty and A. Middeldorp, Eds., vol. 9195. Springer, 2015, pp. 378–388. [Online]. Available: https://doi.org/10.1007/978-3-319-21401-6_26
- [23] T. Altenkirch, P. Capriotti, and N. Kraus, "Extending homotopy type theory with strict equality," in *25th EACSL Annual Conference on Computer Science Logic, CSL 2016, August 29 - September 1, 2016, Marseille, France*, ser. LIPIcs, J. Talbot and L. Regnier, Eds., vol. 62. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016, pp. 21:1–21:17. [Online]. Available: <https://doi.org/10.4230/LIPIcs.CSL.2016.21>
- [24] N. Kraus, "Internal ∞ -categorical models of dependent type theory : Towards 2lft eating hott," in *36th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2021, Rome, Italy, June 29 - July 2, 2021*. IEEE, 2021, pp. 1–14. [Online]. Available: <https://doi.org/10.1109/LICS52264.2021.9470667>
- [25] G. Gilbert, J. Cockx, M. Sozeau, and N. Tabareau, "Definitional proof-irrelevance without K," *Proc. ACM Program. Lang.*, vol. 3, no. POPL, pp. 3:1–3:28, 2019. [Online]. Available: <https://doi.org/10.1145/3290316>
- [26] A. Nuyts, A. Vezzosi, and D. Devriese, "Parametric quantifiers for dependent type theory," *Proc. ACM Program. Lang.*, vol. 1, no. ICFP, aug 2017. [Online]. Available: <https://doi.org/10.1145/3110276>
- [27] E. Cavallo and R. Harper, "Parametric cubical type theory," *CoRR*, vol. abs/1901.00489, 2019. [Online]. Available: <http://arxiv.org/abs/1901.00489>
- [28] H. Herelin, "Investigations into syntactic iterated parametricity and cubical type theory," 2020. [Online]. Available: https://hott-uf.github.io/2020/HoTTUF_2020_paper_22.pdf
- [29] M. Bezem, T. Coquand, and S. Huber, "A model of type theory in cubical sets," in *19th International Conference on Types for Proofs and Programs, TYPES 2013, April 22-26, 2013, Toulouse, France*, ser. LIPIcs, R. Matthes and A. Schubert, Eds., vol. 26. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2013, pp. 107–128. [Online]. Available: <https://doi.org/10.4230/LIPIcs.TYPES.2013.107>
- [30] C. Cohen, T. Coquand, S. Huber, and A. Mörtberg, "Cubical Type Theory: A Constructive Interpretation of the Univalence Axiom," in *TYPES 2015*, ser. LIPIcs, T. Uustalu, Ed., vol. 69. Schloss Dagstuhl, 2018, pp. 5:1–5:34. [Online]. Available: <http://drops.dagstuhl.de/opus/volltexte/2018/8475>
- [31] C. Angiuli, G. Brunerie, T. Coquand, R. Harper, K. H. (Favonia), and D. R. Licata, "Syntax and models of cartesian cubical type theory," *Math. Struct. Comput. Sci.*, vol. 31, no. 4, pp. 424–468, 2021. [Online]. Available: <https://doi.org/10.1017/S0960129521000347>
- [32] The Univalent Foundations Program, *Homotopy Type Theory: Univalent Foundations of Mathematics*. Institute for Advanced Study, 2013. [Online]. Available: <https://homotopytypetheory.org/book/>