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
Université Paris Cité (2020-2022), Unaffiliated r@artagnon.com

Abstract
Semi-simplicial and semi-cubical sets are commonly defined as presheaves over respectively, the semi-simplex or semi-cube category. Homotopy Type Theory then popularized an alternative definition, where the set of n-simplices or n-cubes are instead regrouped into the families of the fibres over their faces, leading to a characterization we call *indexed*. Moreover, it is known that semi-simplicial and semi-cubical sets are related to iterated Reynolds parametricity, respectively in its unary and binary variants. We exploit this correspondence to develop an original uniform indexed definition of both augmented semi-simplicial and semi-cubical sets, and fully formalize it in Coq.

1. Introduction

Fibred vs indexed presentation of semi-simplicial and semi-cubical sets

A family of sets can commonly be represented in two ways: as a family properly speaking, indexed by the elements of a given set S, or as a set T together with a map from T to S, which specifies for each element of T its dependency on S. In the former case, we call it an *indexed* presentation. In the latter case, the set associated to a given element of S is the fibre of this element, so we call it a *fibred* presentation. The two presentations are equivalent and the equivalence can be phrased concisely in the language of homotopy type theory (The Univalent Foundations Program, 2013) as the fibred/indexed equivalence[a].

$$\text{(fibred)} \qquad (\Sigma T : \mathsf{HSet}.(T \to S)) \simeq (S \to \mathsf{HSet}) \qquad \text{(indexed)}$$

Here, HSet represents in homotopy type theory the subset of types within a given universe where equality of any two elements has at most one proof.

A *presheaf* on an category is a family of sets indexed by the object of the category with maps indexed by the morphisms. As such, it lives on the indexed side of the equivalence, contrasting with the fibred side, where we have *discrete Grothendieck fibrations* (Loregian and Riehl, 2020). However, there are situations where a presheaf can also be seen as living on the fibred side of the equivalence. This happens when the indexing category is *direct*, or has a downwards-well-founded collection of non-identity morphisms. Let us consider, for instance, the case of a semi-cubical set (Grandis and Mauri, 2003; Buchholtz and Morehouse, 2017) presented with $2n$ face maps from the set of n-cubes to the set

Competing interests: The authors declare none

[a]In an informal discussion, alternative nomenclatures were proposed: fibration/family equivalence and unbundled/bundled equivalence. The fibred/indexed nomenclature echoes the Grothendieck construction of fibred categories from indexed categories. The most elementary instance of the equivalence, with Type instead of HSet, is sometimes called "Grothendieck construction for dummies", and its proof requires univalence (The Univalent Foundations Program, 2013).

of $(n-1)$-cubes. Formulated in type theory, the corresponding presheaf definition of a semi-cubical set prescribes a family of sets and face maps between them as follows.

$$\begin{tikzcd}
X_0 : \mathsf{HSet} & X_1 : \mathsf{HSet} \arrow[l, shift right, "\partial^R"'] \arrow[l, shift left, "\partial^L"] & X_2 : \mathsf{HSet} \arrow[l, shift right=3, "\partial^{\star R}"'] \arrow[l, shift right=1, "\partial^{\star L}"] \arrow[l, shift left=1, "\partial^{R\star}"] \arrow[l, shift left=3, "\partial^{L\star}"] & \dots
\end{tikzcd}$$

up to cubical faces identities. Here, X_1 can be seen as a family over $X_0 \times X_0$, and X_2 can be seen as a family over $X_1 \times X_1 \times X_1 \times X_1$, in the fibred presentation, together with coherence conditions between the X_1 seen as families over $X_0 \times X_0$. This suggests an alternative indexed presentation of the presheaf as a stratified sequence of families indexed by families of lower rank, taking into account those coherence conditions to prevent duplications. Formulated in type theory, it takes the form:

$$\begin{aligned}
X_0 &: \mathsf{HSet} \\
X_1 &: X_0 \times X_0 \to \mathsf{HSet} \\
X_2 &: \Pi abcd.\, X_1(a,b) \times X_1(c,d) \times X_1(a,c) \times X_1(b,d) \to \mathsf{HSet} \\
&\dots
\end{aligned}$$

The idea for such an indexed presentation of presheaves over a direct category was mentioned at the Univalent Foundations year in the context of defining semi-simplicial types[b]. A few constructions have been proposed since then. The first construction by Voevodsky (2012) relies on the presentation of semi-simplicial sets as a presheaf over increasing injective maps between finite ordinals. The second, by Herbelin (2015)[c] formalized in the Coq proof assistant, relies on the presentation of semi-simplicial sets as a presheaf over face maps. Another by Part and Luo (2015) formalized in an emulation of logic-enriched homotopy type theory in the Plastic proof assistant, and yet another by Altenkirch et al. (2016) formalized in an emulation of a two-level type theory in the Agda proof assistant[d], rely, like in Voevodsky, on the presentation of the semi-simplicial category from increasing injective maps between finite ordinals. The latter constructions, besides being stated as providing semi-simplicial types (thanks to an extension of the type theory), are particularly concise, taking advantage of a definition of increasing injective maps between finite ordinals as type-theoretic functions to inherit the associativity of composition directly from it holding in type theory. This contrasts with the combinatorial construction in Herbelin (2015) where equations over face maps have to be proved by induction.

By taking the sum of each component of an indexed presentation over the indexing set of this component, one obtains back a presheaf in the ordinary sense that has a property of *Reedy fibrancy*, that is whose morphisms are projections in the set-theoretic sense. Such Reedy fibrant presheaves over a direct category have been studied in e.g. Shulman (2015), Kraus and Sattler (2017) and Annenkov et al. (2017, 2023), presenting generic constructions over such presheaves.

The indexed definition of a presheaf over a direct category is technically more involved than the presheaf definition, as it requires hard-wiring in the structure the

[b] ncatlab.org/nlab/show/semi-simplicial+types+in+homotopy+type+theory

[c]In hindsight, the title of the paper "A dependently-typed construction of semi-simplicial types" is somewhat confusing: it implicitly claimed to construct semi-simplicial types, but the construction was done in a type theory with Uniqueness of Identity Proofs. Consequently, what was really obtained was an indexed presentation of semi-simplicial sets. The confusion was however, common at the time.

[d] github.com/nicolaikraus/HoTT-Agda/blob/master/nicolai/SemiSimp/SStypes.agda

dependencies between elements of the sets of the presheaf, including the coherence conditions between these dependencies, such as taking the i-th face of the j-th face of a n-simplex being the same as taking the $(j-1)$-th face of the i-th face (when $j > i$). However, exhibiting a concrete instance of a presheaf in indexed form only requires providing the families, since the responsibility of defining maps and showing the coherence conditions is already accounted for in the definition of the structure.

Reynolds parametricity and its unary and binary variants

In the context of functional programming, Reynolds parametricity (1983) interprets types as relations characterizing the observational behaviour of programs of this type. More generally, families over a product of sets, or *correspondences*, can be used in place of relations. Parametricity can then be iterated, and relying on the fibred presentation of correspondences as spans, it has been noted that iterated Reynolds parametricity has the structure of a cubical set (Altenkirch and Kaposi, 2015; Moulin, 2016; Johann and Sojakova, 2017; Moeneclaey, 2021, 2022). We obtain a *unary* variant of Reynolds *binary* parametricity by using predicates or families instead of relations or correspondences, and this is a form of realizability (Bernardy and Moulin, 2012; Lasson, 2014; Moulin, 2016). Cubical set models which differ only by the arity one (Bernardy et al., 2015) or two (Bezem et al., 2013*a*) were introduced, and this led to a general notion of affine ν-ary cubes in Nuyts and Devriese (2024). In parallel, it has been noted that iterated unary parametricity has the structure of an augmented simplicial set[e]. This suggests that the definition of augmented semi-simplicial sets and semi-cubical sets can in turn be seen as particular instances of the restriction of Nuyts-Devriese's ν-ary cubes to only faces, which we call ν-sets, of presheaves over a ν-semi-shape category made of words of some cardinal $\nu + 1$, where $\nu = 1$ gives augmented semi-simplicial sets and $\nu = 2$ gives semi-cubical sets.

Contribution

The main contribution of the paper is to describe the details of a recipe that uniformly characterizes unary and binary iterated parametricity in indexed form, and to derive from it a new indexed presentation, called indexed *ν-sets*, of augmented semi-simplicial and semi-cubical sets.

Our work is a step in the direction of the program initiated in Altenkirch and Kaposi (2015) to develop parametricity-based models of parametric type theory (Bernardy et al., 2015; Nuyts et al., 2017; Cavallo and Harper, 2020) and cubical type theory (Bezem et al., 2013*a*; Cohen et al., 2018; Angiuli et al., 2021), which are closer to the syntax of type theory, and are likely to better reflect the definitional properties of type theory than presheaf-based cubical sets would. For example, consider the loss of definitional properties when interpreting "indexed" dependent types of type theory as "fibrations" in models such as locally cartesian closed categories (Curien et al., 2014).

Our construction has the unique property of reflecting the structure of parametricity and of yielding both augmented semi-simplicial and semi-cubical sets from the same construction. The approach taken in Part and Luo (2015) and Altenkirch et al. (2016) takes benefit of the definitional compositionality of increasing injective maps, but we do not see how they could be generalized to yield semi-cubical sets.

Our mechanization can be found at github.com/artagnon/bonak. The construction was conceived in Summer 2019, and the mechanization began in late 2019. A sketch of

[e]Private communication with Hugo Moeneclaey and Thorsten Altenkirch.

the construction was presented at the 2020 HoTT-UF workshop, and the completion of the mechanization was reported at the TYPES 2022 conference.

2. Semi-simplicial and semi-cubical sets

In this section, we generalize semi-simplicial and semi-cubical sets to ν-sets, subsuming the earlier definitions. We start with some introductory material on semi-simplicial and semi-cubical sets.

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 "colour". Conversely, semi-simplicial sets can be seen as augmented semi-simplicial sets over the singleton set of a fixed colour. Let us associate dimension -1 to colours; then, points are dimension 0, lines are dimension 1, and so on.

While ordinary semi-simplicial sets are presheaves over the semi-simplex category, augmented semi-simplicial sets are presheaves over $\mathbf{\Delta}_+$. There are different ways to define $\mathbf{\Delta}_+$, up to equivalence, and we use a definition that can be extended to semi-cubical sets in a straightforward manner. In particular, we start numbering objects from 0 instead of -1 so that there is a shift by one compared to the standard numbering of augmented semi-simplicial sets.

Notation 1 (Finite sequences)**.** *We denote finite sequences by $i_1 \dots i_n$ for i_j ranging over some domain. The empty sequence is written ϵ.*

Definition 2 ($\mathbf{\Delta}_+$)**.** *The definition of $\mathbf{\Delta}_+$ is shown below. Note that, if $g \circ f$ is well-defined, then the length of f is less than or equal to that of g. It can be shown that composition is associative and that* id *is neutral.*

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

Definition 3 ($\mathsf{Set}_{\mathbf{\Delta}_+}$)**.** *We define the category of augmented semi-simplicial sets as the functor category:*

$$\mathsf{Set}_{\mathbf{\Delta}_+} := \mathsf{Set}^{\mathbf{\Delta}_+^{\mathsf{op}}}$$

To provide examples, we define the standard augmented n-semi-simplex, taking into account the shift by one in the numbering.

Definition 4 ($\mathbf{\Delta}_+^n$)**.** *The standard augmented* $(n-1)$*-semi-simplex* $\mathbf{\Delta}_+^{n-1}$ *is defined as the Yoneda embedding of* $n \in \mathsf{Obj}(\mathbf{\Delta}_+)$*:*

$$\begin{aligned} \mathbf{\Delta}_+^{n-1} &: \mathsf{Set}_{\mathbf{\Delta}_+} \\ \mathbf{\Delta}_+^{n-1}(p) &:= \mathsf{Hom}(p, n) \\ \mathbf{\Delta}_+^{n-1}(f) &:= \lambda g.\, g \circ f \end{aligned}$$

The standard augmented (-1)-semi-simplex is a singleton made of one colour (in this case, black). Standard augmented n-semi-simplices for $n \geq 0$ have a geometric interpretation, and we illustrate them for dimensions 0, 1, and 2.

Example 5 ($\mathbf{\Delta}_+^0$)**.** The standard augmented 0-semi-simplex can be pictured as a point, coloured black, corresponding to the unique morphism in $\mathsf{Hom}(0, 1)$. This point is the identity in $\mathsf{Hom}(1, 1)$; it is hence shown as a singleton $\star$.

$$\star$$

Example 6 ($\mathbf{\Delta}_+^1$)**.** The standard augmented 1-semi-simplex is drawn as two points, given by $\mathsf{Hom}(1, 2)$, along with a line connecting them, given by $\mathsf{Hom}(2, 2)$. We use black to denote the unique morphisms in $\mathsf{Hom}(0, 1)$ and $\mathsf{Hom}(0, 2)$.

$$\begin{tikzcd} \star 0 \arrow[r, dash, "\star\star"] & 0\star \end{tikzcd}$$

Example 7 ($\mathbf{\Delta}_+^2$)**.** $\mathbf{\Delta}_+^2$ is drawn as three points, given by $\mathsf{Hom}(1, 3)$, three lines connecting them, given by $\mathsf{Hom}(2, 3)$, and a triangular filler given by $\mathsf{Hom}(3, 3)$.

$$\begin{tikzcd} & 00\star \arrow[dl, dash, "\star 0\star"'] \arrow[dr, dash, "0\star\star"] & \\ \star 00 \arrow[rr, dash, "\star\star 0"'] & \star\star\star & 0\star 0 \end{tikzcd}$$

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 colour 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 $\star$. Note that the components may be oriented by letting each n-dimensional component point to the $(n-1)$-dimensional component obtained by replacing the leftmost $\star$ of the n-dimensional component with 0.

Semi-cubical sets

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

Definition 8 ($\square$)**.** *The definition of $\square$ is shown below. The symbols L and R indicate opposite faces of a cube.*

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

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

Definition 9 ($\mathsf{Set}_\square$)**.** *We define the category of semi-cubical sets as the functor category:*

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

Definition 10 ($\square^n$)**.** *The standard semi-cube $\square^n$ is defined as the Yoneda embedding of $n \in$ $\mathsf{Obj}(\square)$:*

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

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

Example 11 ($\square^0$)**.** $\square^0$ is $\mathsf{Hom}(0,0)$, or the singleton set of the empty sequence:

$$\epsilon$$

Example 12 ($\square^1$)**.** $\square^1$ consists of two points, given by $\mathsf{Hom}(0,1)$, and a line, given by $\mathsf{Hom}(1,1)$.

$$\begin{tikzcd} L \arrow[r, dash, "\star"] & R \end{tikzcd}$$

Example 13 ($\square^2$)**.** $\square^2$ consists of four points, given by $\mathsf{Hom}(0,2)$, four lines connecting the four points, given by $\mathsf{Hom}(1,2)$, and a filler, given by $\mathsf{Hom}(2,2)$:

$$\begin{tikzcd} LR \arrow[r, dash, "\star R"] \arrow[d, dash, "L\star"'] & RR \arrow[d, dash, "R\star"] \\ LL \arrow[r, dash, "\star L"'] \arrow[ur, phantom, "\star\star"] & RL \end{tikzcd}$$

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 prism 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 prism is obtained by postfixing with $\star$. Note that the components can be oriented by letting each n-dimensional component go from the $(n-1)$-dimensional component obtained by replacing the leftmost $\star$ with L, to the one obtained by replacing the leftmost $\star$ with R.

ν-***sets***

Let us call ν-sets, the generalization of augmented semi-simplicial sets and semi-cubical sets obtained by building on an arbitrary alphabet ν, so that the following holds:

Cardinal of ν	1	2
Interpretation	Augmented semi-simplicial sets	Semi-cubical sets

To obtain this, we extend $\mathbf{\Delta}_+$ and $\square$ in a straightforward manner into a category which we call ⬡.

Definition 14 (⬡)**.** *The definition of ν-semi-shape category is shown below. Note that, if $g \circ f$ is well-defined, then the length of f is less than or equal to that of g. It can be shown that composition is associative and that* id *is neutral.*

$$\mathsf{Obj}_{⬡} := \mathbb{N}$$

$$\mathsf{Hom}_{⬡}(p, n) := \{l \in (\nu \sqcup \{\star\})^n \mid \textit{number of } \star \textit{ in } l = p\}$$

$$g \circ f := \begin{cases} f & \textit{if } g = \epsilon \\ a\,(g' \circ f) & \textit{if } g = a\,g', \textit{where } a \in \nu \\ a\,(g' \circ f') & \textit{if } g = \star\,g',\ f = a\,f', \textit{where } a \in \nu \textit{ or } a = \star \end{cases}$$

$$\mathsf{id} := \star \ldots \star\ n \textit{ times for } \mathsf{id} \in \mathsf{Hom}_{⬡}(n, n)$$

A ν-set is thus a contravariant 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 (Reynolds, 1983) and more generally for type theory (Bernardy et al., 2010; Bernardy and Lasson, 2011; Atkey et al., 2014; Bernardy et al., 2015).

3. Type theory

Martin-Löf's Type theory (1975; 1984) 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, Coq and Lean. In type theory, propositions are types and proofs are programs. Type theory includes *definitional equality*, by which all propositions and proofs are quotiented.

Type theory is a flexible formalism supporting different models. Some models come from homotopy theory, and are based on simplicial sets or related structures (Hofmann and Streicher, 1994; Kapulkin and Lumsdaine, 2021; Bezem et al., 2013*b*; Cohen et al., 2015): in these models, equality is interpreted as paths, and they support the univalence principle stating that equality of types mimics equivalence of types, leading to the development of Homotopy Type Theory (The Univalent Foundations Program, 2013).

Types are organized in a hierarchy of universes written Type_m for m a natural number. The main types in type theory are the type of dependent pairs, written $\Sigma a : A.\, B(a)$, the type 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 equalities, written $t = u$. As a notation, the type of dependent pairs when B is not dependent on A is shortened into $A \times B$ and the type of dependent functions when B is not dependent on A is written $A \to B$. We assume our type theory to also include a distinguished singleton type, written unit, and with inhabitant $*$, the type of boolean values, and the type of natural numbers. We 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. We also assume that our type theory includes a coinductively-defined notion of dependent streams described in Appendix A.

A type-theoretic notion of sets can be recovered in each universe as HSet_m, denoting the subtype of Type_m for which paths are degenerated, using Uniqueness of Identity Proofs (UIP). Technically, this is expressed as a structure equipping a domain Dom with the property UIP:

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

In HSet_m, the following properties hold:

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

By notation, Type and HSet mean Type_m and HSet_m at some unspecified universe level m.

We are also interested in *extensional* type theory, a type theory with the following reflection rule, where $=$ is propositional equality in some type and $\equiv$ is definitional equality (Martin-Löf, 1984):

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

Note that the reflection rule implies UIP so that HSet and Type are equivalent in extensional type theory. The rule also implies functional extensionality. Extensional type theory is logically equivalent to intensional type theory extended with UIP and functional extensionality (Hofmann, 1995).

4. Relating to parametricity

Recall from the introduction, the form taken by the indexed presentation of a semi-cubical set:

$$\begin{aligned}
&X_0 : \mathsf{HSet}\\
&X_1 : X_0 \times X_0 \to \mathsf{HSet}\\
&X_2 : \Pi abcd.\, X_1(a,b) \times X_1(c,d) \times X_1(a,c) \times X_1(b,d) \to \mathsf{HSet}\\
&\ldots
\end{aligned}$$

Here, the process of construction of the type of X_1 from that of X_0, and of the type of X_2 from that of X_1, is similar to iteratively applying a binary parametricity translation. The binary parametricity which we consider interprets a closed type A by a family $A_\star$ over $A \times A$, and this can be seen as a graph whose vertices are in A. Each type constructor is associated with the construction of a graph. To start with, the type of types HSet is interpreted as the family of type of families $\mathsf{HSet}_\star$, which takes A_L and A_R in HSet and returns the type $A_L \times A_R \to \mathsf{HSet}$ of families over A_L and A_R. Also, for A interpreted by $A_\star$ and $B(a)$, for $a : A$, interpreted by $B_\star((a_L, a_R), a_\star)$ with $a_\star : A_\star(a_L, a_R)$, a dependent function type $\Pi a : A.\, B(a)$ is interpreted as the graph $(\Pi a : A.\, B(a))_\star$ that takes two functions f_L and f_R of type $\Pi a : A.\, B(a)$, and expresses that these functions map related arguments in A to related arguments in B:

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

Similarly, a product type $A \times B$ is interpreted as the graph $(A \times B)_\star$ that relates two tuples (a_L, b_L) and (a_R, b_R) in $A \times B$ as follows:

$$(A \times B)_\star((a_L, b_L), (a_R, b_R)) \triangleq A_\star(a_L, a_R) \times B_\star(b_L, b_R)$$

In particular, for $X : \mathsf{HSet}$, applying our parametricity translation is about associating to X an inhabitant $X_\star$ of $\mathsf{HSet}_\star(X, X)$ i.e. of $X \times X \to \mathsf{HSet}$. In turn, applying the translation again to $X_\star : X \times X \to \mathsf{HSet}$ is about associating to $X_\star$ an inhabitant $X_{\star\star}$ of $(X \times X \to \mathsf{HSet})_\star(X_\star, X_\star)$ i.e. of:

$$\begin{aligned}
&\Pi((x_{LL}, x_{LR}), (x_{RL}, x_{RR})) : ((X \times X) \times (X \times X)).\\
&(X_\star(x_{LL}, x_{LR}) \times X_\star(x_{RL}, x_{RR}) \to X_\star(x_{LL}, x_{RL}) \times X_\star(x_{LR}, x_{RR}) \to \mathsf{HSet})
\end{aligned}$$

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 the composition of binary parametricity with the diagonal on types and type families, applied to an initial $X : \mathsf{HSet}$:

$$\begin{aligned}
&X_0 \triangleq X \quad : \mathsf{HSet}\\
&X_1 \triangleq X_\star \quad : \mathsf{HSet}_\star(X, X)\\
&X_2 \triangleq X_{\star\star} : (\mathsf{HSet}_\star(X, X))_\star(X_\star, X_\star)\\
&\ldots
\end{aligned}$$

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, $X_{n+1} : (S_n)_\star(X_n, X_n)$ 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.

5. Our construction

In this section, we describe our parametricity-based construction of ν-sets in indexed form at two levels of formality.

Sections 5.1 and 5.3 describe the construction at an informal level of discourse:

(1) In 5.1, we present it in informal extensional type theory where equational reasoning is left implicit, and we give an intuition for the construction in 5.2.
(2) While reasoning in extensional type theory is similar to reasoning in set theory regarding how equality is handled, extensional type theory has two limitations. The first limitation is that it enforces the principle of Uniqueness of Identity Proofs and this is inconsistent with the Univalence principle, thus making it inexpressible in Homotopy Type Theory. The second limitation is that we want the construction to be formalizable in the Coq proof assistant whose underlying type theory is intensional. Section 5.3 thus rephrases the construction in (informal) intensional type theory. Since ν-sets are 0-truncated types, we compensate for the absence of UIP by assuming a "local UIP", requiring types to be HSet.

Sections 5.4, 5.5, and 5.6 describe additional issues to be addressed in order to get a fully formal construction:

(1) The well-foundedness of the induction requires a special termination evidence which will be discussed in section 5.4.
(2) The construction is indexed over integers and holds under some constraints on the range of these integers. There is a standard formalization dilemma in this kind of situation: either the constraints on the range are embedded in the construction so that the construction makes sense only on the corresponding range, or the construction is made first on a more general domain than needed but restricted to a smaller domain at the time of use. We adopted the former approach, requiring the construction to be dependent on proofs of inequalities on natural numbers. We discuss how we deal with such dependencies in section 5.5.
(3) A number of standard groupoid properties of equality as well as type isomorphisms have been left implicit in the informal definition. This is discussed in section 5.6.

5.1 The construction in informal type theory

A ν-set in indexed form is a sequence of families of HSet, that is HSet_m for some universe level m. We call such sequence a ν-set at level m, whose type thus lives in HSet_{m+1}.

Table 1 describes the type of a ν-set at level m as a dependent stream of type families representing the limit of n-truncated ν-sets: using the notations of Section 3, the recursive equation $\nu\mathsf{Set}_m^{\geq n} D \triangleq \Sigma R : \nu\mathsf{Set}_m^{=n}(D).\, \nu\mathsf{Set}_m^{\geq n+1}(D, R)$ from the table formally corresponds to the stream $\mathit{Stream}_{\Sigma n.\, \nu\mathsf{Set}_m^{<n},\, \lambda(n,D).\, \nu\mathsf{Set}_m^{=n}(D),\, \lambda((n,D),R).(n+1,(D,R))}(n, D)$. Therefore, $\nu\mathsf{Set}_m^{\geq n}$ denotes an infinite sequence $X_n, X_{n+1}, \ldots$ dependent on a $(<n)$-truncated ν-set, $\nu\mathsf{Set}_m^{<n}$, so that, when n is 0, it denotes a full ν-set, written $\nu\mathsf{Set}_m$. This is

made possible because the (< 0)-truncated ν-set, $\nu\mathsf{Set}_m^{<0}$, is degenerated: it is an empty family, and there is thus only one (< 0)-truncated ν-set, namely the canonical inhabitant $\star$ of unit.

The definition of the type of a n-truncated ν-set is in turn described in table 2. In the infinite sequence of type families representing a ν-set, the n-th component is a type dependent over a fullframe. It is recursively defined in table 3, using the auxiliary definitions of frame, layer and painting. A fullframe describes a boundary of a standard form (simplex, cube), which we decompose into layer, and a painting corresponds to a filled frame. Notice that the type layer relies on an operator of frame restriction $\mathsf{restr}_{\mathsf{frame}}$ which is defined in table 4, and this restriction operator is in turn defined using auxiliary definitions $\mathsf{restr}_{\mathsf{layer}}$ and $\mathsf{restr}_{\mathsf{painting}}$.

Notably, the definition of $\mathsf{restr}_{\mathsf{layer}}$ relies on an equality expressing the commutation of the composition of two $\mathsf{restr}_{\mathsf{frame}}$. The proof of this commutation is worth being made explicit, which we do in table 5 using proof-term notations. The proof requires an induction on the dimension and on the structure of frame, layer, and painting. This is what $\mathsf{coh}_{\mathsf{frame}}$ does using auxiliary proofs $\mathsf{coh}_{\mathsf{layer}}$ and $\mathsf{coh}_{\mathsf{painting}}$. Even though it looks independent of the definitions from the other tables, $\mathsf{coh}_{\mathsf{frame}}$ 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, $\mathsf{restr}_{\mathsf{frame}}$, and $\mathsf{coh}_{\mathsf{frame}}$ has to be defined in one go by induction on p. Also, each of painting, $\mathsf{restr}_{\mathsf{painting}}$, and $\mathsf{coh}_{\mathsf{painting}}$ 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, $\mathsf{restr}_{\mathsf{layer}}$ and $\mathsf{coh}_{\mathsf{layer}}$ are just abbreviations. The exact way this mutual recursion is eventually formalized is explained in section 5.4.

Note that for a fixed constant n, relying on the evaluation rules of type theory, the coherence conditions degenerate to a reflexivity proof, so that the construction builds an effective sequence of types not mentioning coherences anymore.

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

Table 1. Main definition

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

Table 2. Truncated ν-sets, the core

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

Table 3. frame, layer, and painting

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

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

$\mathsf{coh}^{n,p,p\leq r\leq q\leq n-2}_{\mathsf{frame},\epsilon,\omega,q,r}$	$\{D : \nu\mathsf{Set}^{<n}\}$ $(d : \mathsf{frame}(D))$	$:$	$\mathsf{restr}^{n-1,p}_{\mathsf{frame},\epsilon,q}(\mathsf{restr}^{n,p}_{\mathsf{frame},\omega,r}(d))$ $= \mathsf{restr}^{n-1,p}_{\mathsf{frame},\omega,r}(\mathsf{restr}^{n,p}_{\mathsf{frame},\epsilon,q+1}(d))$
$\mathsf{coh}^{n,0}_{\mathsf{frame},\epsilon,\omega,q,r}$	$D\ *$	$\triangleq$	$\mathsf{refl}(*)$
$\mathsf{coh}^{n,p'+1}_{\mathsf{frame},\epsilon,\omega,q,r}$	$D\ (d,l)$	$\triangleq$	$(\mathsf{coh}^{n,p'}_{\mathsf{frame},\epsilon,\omega,q,r}(d), \mathsf{coh}^{n,p'}_{\mathsf{layer},\epsilon,\omega,q-1,r-1}(l))$
$\mathsf{coh}^{n,p,p\leq r\leq q\leq n-3}_{\mathsf{layer},\epsilon,\omega,q,r}$	$(D : \nu\mathsf{Set}^{<n})$ $\{d : \mathsf{frame}(D)\}$ $(l : \mathsf{layer}(d))$	$:$	$\mathsf{restr}^{n-1,p}_{\mathsf{layer},\epsilon,q}(\mathsf{restr}^{n,p}_{\mathsf{layer},\omega,r}(l))$ $= \mathsf{restr}^{n-1,p}_{\mathsf{layer},\omega,r}(\mathsf{restr}^{n,p}_{\mathsf{layer},\epsilon,q+1}(l))$
$\mathsf{coh}^{n,p}_{\mathsf{layer},\epsilon,\omega,q,r}$	$D\ d\ l$	$\triangleq$	$\lambda\theta.\, \mathsf{coh}^{n-1,p}_{\mathsf{painting},\epsilon,\omega,q,r}(D.2)(l_{\theta})$
$\mathsf{coh}^{n,p,p\leq r\leq q\leq n-2}_{\mathsf{painting},\epsilon,\omega,q,r}$	$(D : \nu\mathsf{Set}^{<n})$ $(E : \nu\mathsf{Set}^{=n}(D))$ $(d : \mathsf{frame}(D))$ $(c : \mathsf{painting}(E)(d))$	$:$	$\mathsf{restr}^{n-1,p}_{\mathsf{painting},\epsilon,q}(D.2)(\mathsf{restr}^{n,p}_{\mathsf{painting},\omega,r}(E)(c))$ $= \mathsf{restr}^{n-1,p}_{\mathsf{painting},\omega,r}(D.2)(\mathsf{restr}^{n,p}_{\mathsf{painting},\epsilon,q+1}(E)(c))$
$\mathsf{coh}^{n,p,p=r}_{\mathsf{painting},\epsilon,\omega,q,r}$	$D\ E\ d\ (l,\_)$	$\triangleq$	$\mathsf{refl}(\mathsf{restr}^{n-1,p}_{\mathsf{painting},\epsilon,q-1}(D.2)(l_{\epsilon}))$
$\mathsf{coh}^{n,p,p<r}_{\mathsf{painting},\epsilon,\omega,q,r}$	$D\ E\ d\ (l,c)$	$\triangleq$	$(\mathsf{coh}^{n,p}_{\mathsf{layer},\epsilon,\omega,q,r}(l), \mathsf{coh}^{n,p+1}_{\mathsf{painting},\epsilon,\omega,q,r}(E)(c))$

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

5.2 Intuition for our formal construction

There is a fullframe for each dimension n, written $\mathsf{fullframe}^n$, and every X_n is uniformly assigned a type of the form $\mathsf{fullframe}^n \to \mathsf{HSet}$. Here, $\mathsf{fullframe}^n$ is a "telescope" collecting all arguments of the type of X_i in section 4 as a nesting of Σ-types.

To illustrate how to recursively build $\mathsf{fullframe}^n$, let us begin by setting $\mathsf{fullframe}^0 \triangleq \mathsf{unit}$, so that the type HSet given to X_0 in section 4 can be equivalently formulated as $\mathsf{unit} \to \mathsf{HSet}$. Then, more generally, let each $\mathsf{fullframe}^n$ consist of n layers, written $\mathsf{layer}^{n,p}$ with $p < n$, that we stack in order, starting from unit, and writing $\mathsf{frame}^{n,p}$ for the p first layers of a $\mathsf{fullframe}^n$, so that $\mathsf{fullframe}^n$ is $\mathsf{frame}^{n,n}$. For instance, X_1 is made of one layer, so that it can be written as a Σ-type of an inhabitant of unit and $\mathsf{layer}^{1,0}$. Then, X_2 is similarly made of two layers.

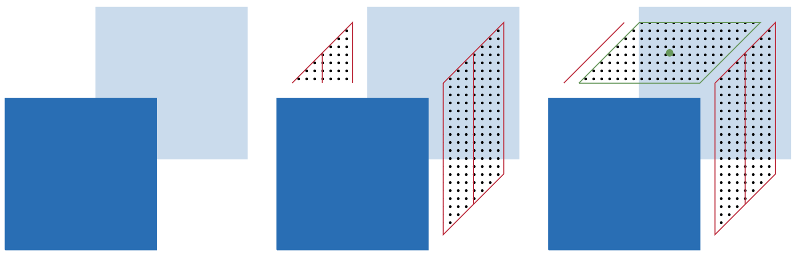
$$X_0 : \underbrace{\mathsf{unit}}_{\mathsf{frame}^{0,0}} \to \mathsf{HSet}$$

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

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

...

Let us now illustrate the construction of $\mathsf{fullframe}^3$, necessary to build the type of X_3.



The figure on the left is $\mathsf{frame}^{3,1}$, in the middle is $\mathsf{frame}^{3,2}$, and on the right is $\mathsf{frame}^{3,3}$, which is full. Further, $\mathsf{frame}^{3,1}$ is made of one layer, $\mathsf{layer}^{3,0}$, shown as the front and back faces (blue boxes), $\mathsf{frame}^{3,2}$ is made of one additional layer, $\mathsf{layer}^{3,1}$, shown as the left and right faces (red boxes), $\mathsf{frame}^{3,3}$ is made of one more layer, $\mathsf{layer}^{3,2}$, shown as the top face (green box).

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

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

So far, we have not paid attention to the fact that we have a dependent type, shown as Σ. To be more precise, note that $\mathsf{fullframe}^{n}$ depends on all X_i up to $n-1$. So, we need to package up X_i, for $i < n$, into a nesting of Σ-types, constituting the type of a n-truncated ν-set, which we wrote $\nu\mathsf{Set}^{<n}$. This allows us to give the type $\nu\mathsf{Set}^{<n} \to \mathsf{HSet}$ to $\mathsf{fullframe}^{n}$. Then, for $D : \nu\mathsf{Set}^{<n}$, representing an initial prefix of $X_0, X_1, \dots X_{n-1}$, the indexed set X_n has type $\mathsf{fullframe}^{n}(D) \to \mathsf{HSet}$. Thus, $\mathsf{frame}^{n,p}$, $\mathsf{layer}^{n,p}$ and $\mathsf{painting}^{n,p}$ also depend on D. We can then reformulate the previous equation with its dependency on D. In particular, X_n is just the last component of D, that is $D.\mathsf{tl}$.

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

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

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

When $\nu = 2$, using L and R to represent the sides, the formation of layers from paintings amounts to:

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

The operation $\mathsf{restr}^{n,p}_{\mathsf{frame},\epsilon,q}$ restricts the p first layers of a frame, and the construction is by recursion on the structure of a frame d. This necessitates the definitions $\mathsf{restr}^{n,p}_{\mathsf{layer},\epsilon,q}(d)(l)$ and $\mathsf{restr}^{n,p}_{\mathsf{painting},\epsilon,q}(d)(c)$, for l a layer, and c a $\mathsf{painting}$. The key case is $\mathsf{restr}^{n,p}_{\mathsf{painting},\epsilon,p}(d)(c)$, where c, a $\mathsf{painting}^{n,p}$, necessarily has the form of $((c_L, c_R), \_)$. Here, $\mathsf{restr}^{n,p}_{\mathsf{painting},L,p}$ picks out c_L, a $\mathsf{painting}^{n-1,p}$, $\mathsf{restr}^{n,p}_{\mathsf{painting},R,p}$ picks out the c_R, also a $\mathsf{painting}^{n-1,p}$, and the last component, shown as _, a $\mathsf{painting}^{n,p+1}$, is discarded. There is one last difficulty, which we illustrate by writing down expected and actual types.

Given c_ω of type

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

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

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

while we expect a component of type

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

Hence, we need a coherence condition to commute the restrictions. Coherence conditions similar to this necessitate what are shown as, $\mathsf{coh}_{\mathsf{frame}}$, $\mathsf{coh}_{\mathsf{layer}}$ and $\mathsf{coh}_{\mathsf{painting}}$ in table 5. These are by induction on the structure of frame, layer and $\mathsf{painting}$. Note that, for the construction in intensional type theory, we further need a 2-dimensional coherence condition, $\mathsf{coh2}_{\mathsf{frame}}$, for $\mathsf{coh}_{\mathsf{layer}}$, which is explained in the next section.

5.3 From extensional to intensional type theory

In this section, we intend to get rid of the reflection rule and make explicit the equational reasoning step needed to rephrase the construction in intensional type theory. For readability purposes, we make only explicit in this section the key coherence conditions of the construction. Other cases of equality reasoning would have to be made explicit to fully obtain a construction in intensional type theory, but these steps are standard enough to be omitted at this stage. See section 5.6 for the details.

The need for transport along a proof of commutation of $\mathsf{restr}_{\mathsf{frame}}$ in the definition of $\mathsf{restr}_{\mathsf{layer}}$ is made explicit in table 4′, where the arrow over $\mathsf{coh}_{\mathsf{frame}}$ indicates the direction of rewrite.

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

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

$\mathsf{coh}^{n,p,p\leq r\leq q\leq n-2}_{\mathsf{frame},\epsilon,\omega,q,r}$	$\{D : \nu\mathsf{Set}^{<n}\}$ $(d : \mathsf{frame}^{n,p}(D))$	$:$	$\mathsf{restr}^{n-1,p}_{\mathsf{frame},\epsilon,q}(\mathsf{restr}^{n,p}_{\mathsf{frame},\omega,r}(d))$ $= \mathsf{restr}^{n-1,p}_{\mathsf{frame},\omega,r}(\mathsf{restr}^{n,p}_{\mathsf{frame},\epsilon,q+1}(d))$
$\mathsf{coh}^{n,0}_{\mathsf{frame},\epsilon,\omega,q,r}$	$D\ *$	$\triangleq$	$\mathsf{refl}(*)$
$\mathsf{coh}^{n,p'+1}_{\mathsf{frame},\epsilon,\omega,q,r}$	$D\ (d, l)$	$\triangleq$	$(\mathsf{coh}^{n,p'}_{\mathsf{frame},\epsilon,\omega,q,r}(d), \mathsf{coh}^{n,p'}_{\mathsf{layer},\epsilon,\omega,q-1,r-1}(l))$
$\mathsf{coh}^{n,p,p\leq r\leq q\leq n-3}_{\mathsf{layer},\epsilon,\omega,q,r}$	$(D : \nu\mathsf{Set}^{<n})$ $\{d : \mathsf{frame}^{n,p}(D)\}$ $(l : \mathsf{layer}^{n,p}(d))$	$:$	$\overrightarrow{\mathsf{coh}^{n,p}_{\mathsf{frame},\epsilon,\omega,q+1,r+1}(d)}(\mathsf{restr}^{n-1,p}_{\mathsf{layer},\epsilon,q}(\mathsf{restr}^{n,p}_{\mathsf{layer},\omega,r}(l)))$ $= \mathsf{restr}^{n-1,p}_{\mathsf{layer},\omega,r}(\mathsf{restr}^{n,p}_{\mathsf{layer},\epsilon,q+1}(l))$
$\mathsf{coh}^{n,p}_{\mathsf{layer},\epsilon,\omega,q,r}$	$D\ d\ l$	$\triangleq$	$\lambda\theta.(\overrightarrow{\mathsf{coh2}^{n,p}_{\mathsf{frame},\epsilon,\omega,\theta,q,r}(d)})(\mathsf{ap}\ (\overrightarrow{\mathsf{coh}^{n-1,p}_{\mathsf{frame},\omega,\theta,r,p}(\mathsf{restr}^{n,p}_{\mathsf{frame},\epsilon,q+2}(d))}))$ $(\mathsf{ap}\ (\overrightarrow{\mathsf{restr}^{n-2,p}_{\mathsf{frame},\omega,r}(\mathsf{coh}^{n,p}_{\mathsf{frame},\epsilon,\theta,q+1,p})}))\ \mathsf{coh}^{n-1,p}_{\mathsf{painting},\epsilon,\omega,q,r}(D.2)(l_\theta)$
$\mathsf{coh}^{n,p,p\leq r\leq q\leq n-2}_{\mathsf{painting},\epsilon,\omega,q,r}$	$(D : \nu\mathsf{Set}^{<n})$ $(E : \nu\mathsf{Set}^{=n}(D))$ $(d : \mathsf{frame}^{n,p}(D))$ $(c : \mathsf{painting}^{n,p}(E)(d))$	$:$	$\overrightarrow{\mathsf{coh}^{n,p}_{\mathsf{frame},\epsilon,\omega,q,r}(d)}(\mathsf{restr}^{n-1,p}_{\mathsf{painting},\epsilon,q}(D.2)(\mathsf{restr}^{n,p}_{\mathsf{painting},\omega,r}(E)(c)))$ $= \mathsf{restr}^{n-1,p}_{\mathsf{painting},\omega,r}(D.2)(\mathsf{restr}^{n,p}_{\mathsf{painting},\epsilon,q+1}(E)(c))$
$\mathsf{coh}^{n,p,p=r}_{\mathsf{painting},\epsilon,\omega,q,r}$	$D\ E\ d\ (l, \_)$	$\triangleq$	$\mathsf{refl}(\mathsf{restr}^{n-1,p}_{\mathsf{painting},\epsilon,q-1}(D.2)(l_\epsilon))$
$\mathsf{coh}^{n,p,p<r}_{\mathsf{painting},\epsilon,\omega,q,r}$	$D\ E\ d\ (l, c)$	$\triangleq$	$(\mathsf{coh}^{n,p}_{\mathsf{layer},\epsilon,\omega,q,r}(l), \mathsf{coh}^{n,p+1}_{\mathsf{painting},\epsilon,\omega,q,r}(E)(c))$

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

The proof of $\mathsf{coh}_{\mathsf{frame}}$ itself requires making explicit several rewrites which were invisible in extensional type theory. The commutation of $\mathsf{restr}_{\mathsf{layer}}$ lives in a type referring to $\mathsf{coh}_{\mathsf{frame}}$, so we need a transport along the commutation of $\mathsf{restr}_{\mathsf{frame}}$ in the statement of $\mathsf{coh}_{\mathsf{layer}}$. The proof of $\mathsf{coh}_{\mathsf{layer}}$ is the most involved proof of the construction, as it requires a higher-dimensional coherence condition, $\mathsf{coh2}_{\mathsf{frame}}$, whose exact formulation is as follows.

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

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 construction is done in HSet, the term is trivially discharged.

Notice that each $\mathsf{restr}_{\mathsf{layer}}$ in the type of $\mathsf{coh}_{\mathsf{layer}}$ is hiding a $\mathsf{coh}_{\mathsf{frame}}$ rewrite: this makes a sum total of three $\mathsf{coh}_{\mathsf{frame}}$ rewrites on the left-hand side, and two $\mathsf{coh}_{\mathsf{frame}}$ rewrites on the right-hand side. In the proof term of $\mathsf{coh}_{\mathsf{layer}}$, $\mathsf{coh}_{\mathsf{painting}}$ has one $\mathsf{coh}_{\mathsf{frame}}$ rewrite on its left-hand side. This, combined with the two terms of the form ap $\mathsf{coh}_{\mathsf{frame}}$, matches our expectation of three $\mathsf{coh}_{\mathsf{frame}}$ on the left-hand side, and two $\mathsf{coh}_{\mathsf{frame}}$ on the right-hand side. Then, $\mathsf{coh2}_{\mathsf{frame}}$ can be seen as expressing the commutation of these $\mathsf{coh}_{\mathsf{frame}}$ terms.

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

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 Herbelin (2015); Altenkirch et al. (2016); Kraus (2021) for a discussion on the need for recursive higher-dimensional coherence conditions in formulating higher-dimensional structures in type theory.

5.4 Well-foundedness of the construction

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, and had terms dependent on the proofs of the case distinction that $n' \leq n$ implies $n' < n$ or $n' = n$, but these proofs did not come with enough definitional properties to be usable in practice. Hence, we chose a different route: in practice, since $\mathsf{restr}^{n}_{\mathsf{frame}}$ depends on frame^{n} and frame^{n-1}, while $\mathsf{coh}^{n}_{\mathsf{frame}}$ depends on frame^{n}, frame^{n-1}, and frame^{n-2}, we only need to keep track of three consecutive dimensions. Hence, what we build by induction at level n, is a structure made not only of the definitions shown in the tables 3, 4′, and 5′, but also of frame, layer, painting at levels $n-1$ and $n-2$, as well as $\mathsf{restr}_{\mathsf{frame}}$, $\mathsf{restr}_{\mathsf{layer}}$, and $\mathsf{restr}_{\mathsf{painting}}$ at level $n-1$, together with helper equations.

5.5 Dependencies in inequality proofs

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 use an intermediate "recursive definition" phrased as:

```
Fixpoint leR (n m : nat) : SProp :=
match n, m with
| 0, _ => STrue
| S n, 0 => SFalse
| S n, S m => leR n m
end.
```

Here, SProp is a definitionally proof-irrelevant impredicative universe morally[f] living at the bottom of the universe hierarchy (Gilbert et al., 2019). By placing the definition in SProp, we have definitional equality of inequality proofs. However, for the purpose of unification, this definition does not go far enough. Consider the unification problems:

```
leR_trans ?p leR_refl = ?p
leR_trans leR_refl ?p = ?p
```

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

```
Definition leY n m :=
  forall p, leR p n -> leR p m.
```

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 existential variables, and we have to provide them explicitly.

The second variant of $\leq$, the "inductive variant", is phrased as:

```
Inductive leI : nat -> nat -> Type :=
| leI_refl n : n <~ n
| leI_down {n p} : p.+1 <~ n -> p <~ n
where "n <~ m" := (leI n m) : nat_scope.
```

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.

5.6 Groupoid properties of equality and basic type isomorphisms

The construction relies on groupoid properties of equality which are left implicit in table 5′. The use of the equivalence between $u = v$ and $\Sigma(p : u.\mathsf{hd} = v.\mathsf{hd}).(u.\mathsf{tl} = v.\mathsf{tl})$ for u and v in a Σ-type is left implicit in the same table. Also implicit is the use of the equivalence 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, or functional extensionality, holds by default in extensional type theory. These have to be made explicit in the formalization.

[f]In Coq, it is however a stand-alone universe unrelated to the universe hierarchy.

As a final remark, note that as a consequence of η-conversion for finite enumerated types, the requirement of functional extensionality disappears when ν is finite. However, this is a conversion which Coq does not implement, and the alternative would be to replace $\Pi a : \nu.\, B$ by a "flat" iterated product $B(1) \times B(2) \times \ldots \times B(\nu)$.

6. Future work

The construction could be extended with degeneracies as well as with permutations (Grandis and Mauri, 2003). Dependent ν-sets could also be defined, opening the way to construct Π-types and Σ-types of ν-sets. A ν-set of ν-sets representing a universe could also be defined as sketched in a talk at the HoTT-UF workshop for the bridge case (2020). More generally, we believe these lines of work would eventually provide alternative models to parametric type theories (Nuyts et al., 2017; Cavallo and Harper, 2020) where equality of types, now a family rather than the total space of a fibration, is not only definitionally isomorphic to bridges (Bernardy et al., 2015), but definitionally the same as bridges.

By equipping the universe construction with a structure of equivalences, as suggested along the lines of Altenkirch and Kaposi (2015), we also suspect the construction to be able to serve as a basis for syntactic models of various versions of cubical type theory (Bezem et al., 2013*a*; Cohen et al., 2018; Angiuli et al., 2021), saving the detour via the fibred approach inherent to usual presheaf models. In particular, we conjecture being able to justify univalence holding definitionally. Our approach would also firmly ground cubical type theory in iterated parametricity.

Although prior approaches to constructing the indexed presentation of a presheaf over a direct category rely on it being evident by inspection that the fibred and indexed presentations are equivalent, no formal proof has been given, and this is a direction for future work. In our construction, we can check by computation of the first levels that it indeed computes the expected sets.

References

Altenkirch, T., Capriotti, P., and Kraus, N. 2016. Extending homotopy type theory with strict equality. In **Talbot, J. and Regnier, L.**, editors, *25th EACSL Annual Conference on Computer Science Logic, CSL 2016, August 29 - September 1, 2016, Marseille, France*, volume 62 of *LIPIcs*, pp. 21:1–21:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

Altenkirch, T. and Kaposi, A. 2015. Towards a cubical type theory without an interval. In **Uustalu, T.**, editor, *21st International Conference on Types for Proofs and Programs, TYPES 2015, May 18-21, 2015, Tallinn, Estonia*, volume 69 of *LIPIcs*, pp. 3:1–3:27. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

Angiuli, C., Brunerie, G., Coquand, T., Harper, R., Hou (Favonia), K.-B., and Licata, D. R. 2021. Syntax and models of cartesian cubical type theory. *Math. Struct. Comput. Sci.*, 31(4):424–468.

Annenkov, D., Capriotti, P., and Kraus, N. 2017. Two-level type theory and applications. *CoRR*, abs/1705.03307.

Annenkov, D., Capriotti, P., Kraus, N., and Sattler, C. 2023. Two-level type theory and applications. *Mathematical Structures in Computer Science*, 33(8):688–743.

Atkey, R., Ghani, N., and Johann, P. 2014. 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*.

Bernardy, J., Coquand, T., and Moulin, G. 2015. A presheaf model of parametric type theory. *Electr. Notes Theor. Comput. Sci.*, 319:67–82.

Bernardy, J., Jansson, P., and Paterson, R. 2010. Parametricity and dependent types. In **Hudak, P. and Weirich, S.**, editors, *Proceeding of the 15th ACM SIGPLAN international conference on Functional programming, ICFP 2010, Baltimore, Maryland, USA, September 27-29, 2010*, pp. 345–356. ACM.

Bernardy, J. and Lasson, M. 2011. 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*, volume 6604 of *Lecture Notes in Computer Science*, pp. 108–122. Springer.

Bernardy, J.-P. and Moulin, G. 2012. A computational interpretation of parametricity. In *2012 27th Annual IEEE Symposium on Logic in Computer Science*, pp. 135–144. IEEE.

Bezem, M., Coquand, T., and Huber, S. 2013a. A model of type theory in cubical sets. In **Matthes, R. and Schubert, A.**, editors, *19th International Conference on Types for Proofs and Programs, TYPES 2013, April 22-26, 2013, Toulouse, France*, volume 26 of *LIPIcs*, pp. 107–128. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik.

Bezem, M., Coquand, T., and Huber, S. 2013b. A model of type theory in cubical sets. In **Matthes, R. and Schubert, A.**, editors, *19th International Conference on Types for Proofs and Programs, TYPES 2013, April 22-26, 2013, Toulouse, France*, volume 26 of *LIPIcs*, pp. 107–128. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik.

Buchholtz, U. and Morehouse, E. 2017. Varieties of cubical sets. In *International Conference on Relational and Algebraic Methods in Computer Science*, pp. 77–92. Springer.

Cavallo, E. and Harper, R. 2020. Internal Parametricity for Cubical Type Theory. In **Fernández, M. and Muscholl, A.**, editors, *28th EACSL Annual Conference on Computer Science Logic (CSL 2020)*, volume 152 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pp. 13:1–13:17, Dagstuhl, Germany. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

Cohen, C., Coquand, T., Huber, S., and Mörtberg, A. 2015. Cubical type theory: A constructive interpretation of the univalence axiom. In **Uustalu, T.**, editor, *21st International Conference on Types for Proofs and Programs, TYPES 2015, May 18-21, 2015, Tallinn, Estonia*, volume 69 of *LIPIcs*, pp. 5:1–5:34. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

Cohen, C., Coquand, T., Huber, S., and Mörtberg, A. 2018. Cubical Type Theory: A Constructive Interpretation of the Univalence Axiom. In **Uustalu, T.**, editor, *TYPES 2015*, volume 69 of *LIPIcs*, pp. 5:1–5:34. Schloss Dagstuhl.

Curien, P.-L., Garner, R., and Hofmann, M. 2014. Revisiting the categorical interpretation of dependent type theory. *Theoretical Computer Science*, 546:99–119. Models of Interaction: Essays in Honour of Glynn Winskel.

Gilbert, G., Cockx, J., Sozeau, M., and Tabareau, N. 2019. Definitional proof-irrelevance without K. *Proc. ACM Program. Lang.*, 3(POPL):3:1–3:28.

Grandis, M. and Mauri, L. 2003. Cubical sets and their site. *Theory and Applications of Categories*, 11:185–211.

Herbelin, H. 2015. A dependently-typed construction of semi-simplicial types. *Mathematical Structures in Computer Science*, 25(5):1116–1131.

Herbelin, H. 2020. Investigations into syntactic iterated parametricity and cubical type theory. Available at `https://hott-uf.github.io/2020/HoTTUF_2020_paper_22.pdf`.

Hofmann, M. 1995. *Extensional concepts in intensional type theory*. PhD thesis, University of Edinburgh.

Hofmann, M. and Streicher, T. 1994. The groupoid model refutes uniqueness of identity proofs. In *Proceedings of the Ninth Annual Symposium on Logic in Computer Science (LICS '94), Paris, France, July 4-7, 1994*, pp. 208–212. IEEE Computer Society.

Johann, P. and Sojakova, K. 2017. Cubical categories for higher-dimensional parametricity. *CoRR*, abs/1701.06244.

Kapulkin, K. and Lumsdaine, P. L. 2021. The simplicial model of Univalent Foundations (after Voevodsky). *Journal of the European Mathematical Society*, 23(6):2071–2126.

Kraus, N. 2021. Internal ∞-categorical models of dependent type theory : Towards 2LTT eating HoTT. In *36th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2021, Rome, Italy, June 29 - July 2, 2021*, pp. 1–14. IEEE.

Kraus, N. and Sattler, C. 2017. Space-valued diagrams, type-theoretically. *arXiv preprint arXiv:1704.04543*.

Lasson, M. 2014. *Réalisabilité et Paramétricité dans les Systèmes de Types Purs*. PhD thesis, Université de Lyon.

Loregian, F. and Riehl, E. 2020. Categorical notions of fibration. *Expositiones Mathematicae*, 38(4):496–514.

Martin-Löf, P. 1975. An intuitionistic theory of types, predicative part. In *Logic Colloquium*, pp. 73–118. North Holland.

Martin-Löf, P. 1984. *Intuitionistic type theory*, volume 1 of *Studies in proof theory*. Bibliopolis.

Moeneclaey, H. 2021. 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*, pp. 1–11. IEEE.

Moeneclaey, H. 2022. *Cubical models are cofreely parametric*. PhD thesis, Université Paris Cité.

Moulin, G. 2016. *Internalizing parametricity*. PhD thesis, Department of Computer Science and Engineering, Chalmers University of Technology.

Nuyts, A. and Devriese, D. 2024. Transpension: The Right Adjoint to the Pi-type. *Logical Methods in Computer Science*, Volume 20, Issue 2.

Nuyts, A., Vezzosi, A., and Devriese, D. 2017. Parametric quantifiers for dependent type theory. *Proc. ACM Program. Lang.*, 1(ICFP).

Part, F. and Luo, Z. 2015. Semi-simplicial types in logic-enriched homotopy type theory. *CoRR*, abs/1506.04998.

Reynolds, J. C. 1983. Types, abstraction and parametric polymorphism. In **Mason, R. E. A.**, editor, *Information Processing 83, Proceedings of the IFIP 9th World Computer Congress, Paris, France, September 19-23, 1983*, pp. 513–523. North-Holland/IFIP.

Shulman, M. 2015. Univalence for inverse diagrams and homotopy canonicity. *Mathematical Structures in Computer Science*, 25(5):1203–1277.

The Univalent Foundations Program 2013. *Homotopy Type Theory: Univalent Foundations of Mathematics*. Institute for Advanced Study.

Voevodsky, V. 2012. Semi-simplicial types. Briefly described in Section 8 of Herbelin (2015).

Appendix A. Definition of dependent stream

type formation

$$\frac{\Gamma \vdash A : \mathsf{Type}_m \qquad \Gamma, a : A \vdash B(a) : \mathsf{Type}_m \qquad \Gamma, a : A, b : B(a) \vdash f(a,b) : A \qquad \Gamma \vdash u : A}{\Gamma \vdash \mathit{Stream}_{A,B,f}\; u : \mathsf{Type}_m}$$

introduction and eliminations

$$\frac{\begin{array}{l}\Gamma \vdash A : \mathsf{Type}_m \qquad \Gamma, a : A \vdash B(a) : \mathsf{Type}_m \qquad \Gamma, a : A, b : B(a) \vdash f(a,b) : A \qquad \Gamma \vdash u : A \\ \Gamma, a : A \vdash D(a) : \mathsf{Type}_m \qquad \Gamma \vdash w : D(u) \\ \Gamma, a : A, d : D(a) \vdash v(a,d) : B(a) \qquad \Gamma, a : A, d : D(a) \vdash s(a,d) : D(f(a, v(a,d)))\end{array}}{\Gamma \vdash \mathsf{cofix}^{u,w}_{a,d,g}\{\mathit{this} := v(a,d); \mathit{next} := g(f(a,v(a,d)), s(a,d))\} : \mathit{Stream}_{A,B,f}\; u}$$

$$\frac{\Gamma \vdash t : \mathit{Stream}_{A,B,f}\; u}{\Gamma \vdash t.\mathit{this} : B(u)}$$

$$\frac{\Gamma \vdash t : \mathit{Stream}_{A,B,f}\; u}{\Gamma \vdash t.\mathit{next} : \mathit{Stream}_{A,B,f}\; f(u, t.\mathit{this})}$$

computation

$$\frac{\begin{array}{l}\Gamma \vdash A : \mathsf{Type}_m \qquad \Gamma, a : A \vdash B(a) : \mathsf{Type}_m \qquad \Gamma, a : A, b : B(a) \vdash f(a,b) : A \qquad \Gamma \vdash u : A \\ \Gamma, a : A \vdash D(a) : \mathsf{Type}_m \qquad \Gamma \vdash w : D(u) \\ \Gamma, a : A, d : D(a) \vdash v(a,d) : B(a) \qquad \Gamma, a : A, d : D(a) \vdash s(a,d) : D(f(a, v(a,d)))\end{array}}{\Gamma \vdash \mathsf{cofix}^{u,w}_{a,d,g}\{\mathit{this} := v(a,d); \mathit{next} := g(f(a,v(a,d)), s(a,d))\}.\mathit{this} \equiv v(u,w) : B(u)}$$

$$\frac{\begin{array}{l}\Gamma \vdash A : \mathsf{Type}_m \qquad \Gamma, a : A \vdash B(a) : \mathsf{Type}_m \qquad \Gamma, a : A, b : B(a) \vdash f(a,b) : A \qquad \Gamma \vdash u : A \\ \Gamma, a : A \vdash D(a) : \mathsf{Type}_m \qquad \Gamma \vdash w : D(u) \\ \Gamma, a : A, d : D(a) \vdash v(a,d) : B(a) \qquad \Gamma, a : A, d : D(a) \vdash s(a,d) : D(f(a, v(a,d)))\end{array}}{\begin{array}{r}\Gamma \vdash \mathsf{cofix}^{u,w}_{a,d,g}\{\mathit{this} := v(a,d); \mathit{next} := g(f(a,v(a,d)), s(a,d))\}.\mathit{next} \equiv \\ \mathsf{cofix}^{f(u,v(u,w)),s(u,w))}_{a,d,g}\{\mathit{this} := v(a,d); \mathit{next} := g(f(a,v(a,d)), s(a,d))\} \\ : \mathit{Stream}_{A,B,f}\; f(u, v(u,w))\end{array}}$$

where $\mathsf{cofix}^{u,w}_{a,d,g}\{\mathit{this} := v(a,d); \mathit{next} := g(f(a,v(a,d)), s(a,d))\}$ is a notation for the instantiation on parameter u and internal value w of the corecursive definition of a stream over an arbitrary a generated by a recipe dependent on an arbitrary internal value $d : D(a)$ with first component given by $v(a,d)$ and second component given by $g(f(a,v(a,d)), s(a,d))$ where g, typed as $\Gamma, a : A, d : D(a) \vdash g(a,d) : \mathit{Stream}_{A,B,f}\; (f(a,d))$, formally represents the recursive call, and where $s(a,d)$ tells how the internal value evolves.