Lawvere-Tierney sheafification in Homotopy Type Theory

Kevin Quirin
École des Mines de Nantes, France
and
Nicolas Tabareau
Inria, France

Sheafification is a popular tool in topos theory which allows the extension of the internal logic of a topos with new principles. One of its most famous applications is the transformation of a given topos into a boolean topos using the dense topology, a construction which corresponds in essence to Gödel's double negation translation. The same construction has not yet been developed in Martin-Löf type theory due to a mismatch between topos theory and type theory. This mismatch has been fixed recently by the introduction of homotopy type theory, an extension of Martin-Löf type theory with new principles inspired by category theory and homotopy theory, and corresponding closely with the theory of higher toposes. In this paper, we give a computer-checked construction of Lawvere-Tierney sheafification in homotopy type theory.

1. INTRODUCTION

Sheafification [MM92] is a very powerful geometric construction which arose initially in topology and algebraic geometry but which, nonetheless, quickly found important applications to mathematical logic. In topos theory, sheafification can be seen as a way to transform one topos into another. It is used, for example, to build, from any topos $\mathcal{T}$, a boolean topos (*i.e.* one which validates the excluded middle principle) satisfying the axiom of choice and negating the continuum hypothesis [MM92, Theorem VI.2.1]. This is, in fact, an adaptation of a slightly older set theoretic method called *forcing* which transforms a model $\mathfrak{M}$ of ZFC into a model $\mathfrak{M}[G]$ of ZFC, satisfying new principles. Its most famous application is the proof of consistency of ZFC with the negation of the continuum hypothesis by Paul Cohen [Coh66], answering (neither negatively nor positively) Hilbert's first problem. Indeed, Gödel proved in 1938 the consistency of ZFC with continuum hypothesis [Göd38] using the constructible model $\mathfrak{L}$. The central idea of forcing is to add to the theory ZFC partial information about the witness of $\neg$CH. Then, supposing that ZFC is coherent, it is provable that ZFC together with a finite number of approximation of the desired object is still consistent. Finally, the compactness theorem allows one to prove the consistency of ZFC with *all* approximations, *i.e.* with a witness of $\neg$CH.

Subsequently, forcing was adapted to topos theory by Myles Tierney [Tie72a] making use of the notion of sheaves. Note that, in topos theory, there are two different kind of sheaves: Grothendieck sheaves, which only exist on a presheaf topos, and Lawvere-Tierney sheaves. One can show that Lawvere-Tierney sheaves, when considered on a presheaf topos, are exactly the Grothendieck sheaves; thus, Lawvere-Tierney sheaves can be seen as a generalization of Grothendieck sheaves.

Given a topos $\mathcal{T}$, one can build another topos – the topos of sheaves $\mathrm{Sh}(\mathcal{T})$ – together with a geometric embedding from $\mathrm{Sh}(\mathcal{T})$ to $\mathcal{T}$ whose left adjoint is called *sheafification*. Depending on the sheaves we choose to treat, the topos $\mathrm{Sh}(\mathcal{T})$ may satisfy new principles. The construction of the geometric embedding is done in [MM92, Section V.3], and briefly recalled in section 4.

Type theory is known to have a close relationship with topos theory, prompting one to wonder if similar techniques might not be developed for type theory. The answer to this question has been given recently by the advent of homotopy type theory [UFP13], an extension of Martin-Löf type theory with principles inspired by (higher) category theory and homotopy theory, such as higher inductive types [LS13, UFP13] and Voevodsky's univalence principle [KL12], which says that for any types T and U, the canonical map

$$(T = U) \to (T \simeq U)$$

transporting equalities to equivalences, is itself an equivalence. This new point of view on type theory has revealed how types carry homotopical structure. For instance, mere propositions (or Type_{-1}) are just types with an irrelevant equality and sets (or Type_0) are types with a propositional equality and so on for Type_n. The development of univalence has also shed some light on the difficulty of making AC and EM coexist in type theory. Indeed, it has been shown that a naive (non-propositional) version of EM is inconsistent with univalence.

When restricted to mere propositions and sets, homotopy type theory corresponds quite closely to topos theory but the mismatch starts when considering higher homotopy types. Fortunately, a higher version of topos theory has been developed recently, synthesized in the monograph of Lurie on higher topos theory [Lur09]. Even if the connection between homotopy type theory and higher topos theory has not yet been made perfectly precise, it is commonly believed that the former constitutes an internal language for the latter (see [Shu15b] for a more detail discussion on this topic).

Lurie has adapted a considerable number of the tools of topos theory to a higher setting. In particular, the theory of sheaves has been lifted to higher topos theory. As the notion of higher topos appears to correspond very closely to homotopy type theory, this provides a new hope that tackling the problem of extending the power of homotopy type theory using sheafification is actually possible.

Nevertheless, the adaptation of the sheafification in higher topos theory to homotopy type theory is not completely straightforward because the construction in higher toposes is restricted to the initial Grothendieck setting which is still very topologically oriented, and hence not very amenable to formalization in type theory. It seems more promising to use a synthetic notion of sheafification, called Lawvere-Tierney sheafification [Tie72b, MM92], but this construction has not been considered yet in the setting of higher topos theory. This raises two issues that this paper addresses: (i) how to lift the notion of Lawvere-Tierney sheafification to higher topos theory and (ii) whether is it possible to formalize this new definition in homotopy type theory.

This paper presents a definition of the sheafification functor in the setting of homotopy type theory. As Lawvere-Tierney sheaves have not, to our knowledge, been considered in the higher setting, the contribution of this article is twofold:

(i) we propose a construction which coincides with Lawvere-Tierney sheafification when restricted to hSets by connecting it to the work on higher modalities [UFP13], (ii) we formalize all the definitions and theorems internally inside the Coq proof assistant [CDT15].

Plan of the paper

Section 2 introduces the necessary preliminaries on Homotopy Type Theory while Section 3 presents the version of higher modalities that we use. Section 4 recalls the definition of sheafification in toposes. Section 5 presents the main contribution of this article, the definition of sheafification in Homotopy Type Theory, and Section 6 discusses its formalization inside the Coq proof assistant.

Related Work

Similar questions have been considered in relation to the Curry-Howard isomorphism, that is, the possibility of extending a programming language close to type theory with new logical or computational principles while keeping consistency automatically. For instance, much effort has been invested in providing a computational content for the law of excluded middle in order to define a constructive version of classical logic. This has lead to various calculi, most notably the $\lambda\mu$-calculus of Parigot [Par93]. However, this line of work has not appeared to be fruitful in defining a new version of type theory with classical principles. Other authors have tried to extend the continuation-passing-style (CPS) transformation to type theory. They have been faced, however, with the difficulty that the CPS transformation is incompatible with (full) dependent sums [BU02], emphasizing the fragile link between the axiom of choice and the law of excluded middle in type theory. Nevertheless the axiom of choice has been shown to be realizable by computational meaning in a classical setting by techniques turning around the notion of (modified) bar induction [BBC98], Krivine's realizability [Kri03] and even more recently with restriction on elimination of dependent sums and lazy evaluation [Her12]. The work on forcing in type theory [JTS12, JLP$^+$16] also gives a computational meaning to a type theory enriched with new logical or computational principles. Actually, this construction is entirely complementary to forcing in type theory, as forcing corresponds to the presheaf construction while Lawvere-Tierney sheafification corresponds to the topological transformation that allows the passage from the presheaf construction to the sheaf construction.

2. PRELIMINARIES ON HOMOTOPY TYPE THEORY

In this section, we review some basic definitions in homotopy type theory which will be central to our formalization but not specific to sheafification. Some of the definitions and theorems of Section 2.1 appear in [UFP13] (or are direct applications of results which do) while others are specific to our formalization.

As a prerequisite, we encourage the reader to be familiar with type theory and in particular the point of view developed in [UFP13]. Nevertheless, we recall most of the central definitions that we use so that the paper is sufficiently self-contained. Given a type T and a type family $U : T \to \text{Type}$, we denote $\prod_{x:T} Ux$ for the dependent product, $\sum_{x:T} Ux$ for the dependent sum, and π_1, π_2 for the first and second projection of a dependent pair (denoted $(a; b)$). The identity path will be denoted 1. We use informal mathematical language instead of type theory whenever it is possible, to ease in reading without making our statements imprecise. In particular, (higher) inductive types are defined using itemization to avoid an overhead of notation. Throughout the paper, Type must be seen in an universe-polymorphic way.

Section 2.1 will present homotopy levels and object classifiers, section 2.2 introduces a theory of colimits in homotopy type theory, illustrated by an important example in section 2.3.

2.1 Homotopy Types and Classifying Objects

One of the most direct applications of homotopical notions to type theory is the introduction of homotopy types. Using the analogy that points in a space correspond to elements of a type and that paths between two points correspond to elements of the corresponding identity type (which defines equality in type theory), an n-type is simply a type for which equality becomes trivial above level n. Voevodsky has realized that this notion admits a compact inductive definition internal to type theory, given by

Definition 2.1. Is-n-type is defined by induction on $n \geqslant -2$:

—Is-(-2)-type X if X is a contractible type, *i.e.* X is pointed by $c : X$, and every other point in X is equal to c.

—Is-$(n+1)$-type $X \overset{def}{=} \prod_{x,y:X} \text{Is-}n\text{-type}(x = y)$.

Then, $\text{Type}_n \overset{def}{=} \sum_{X:\text{Type}} \text{Is-}n\text{-type}\, X$.

When $n = -1$, we will use IsHProp and HProp instead of Is-(-1)-type and Type_{-1}.

From any type T, the type $\|T\|_n : \text{Type}_n$ can be constructed as the HIT generated by

—a function $|\cdot|_n : T \to \|A\|_n$,

—a proof of Is-n-type $\|T\|_n$,

satisfying the following universal property:

Lemma 2.2. *For any* $A : \text{Type}$ *and* $B : \text{Type}_n$*, if* $f : A \to B$ *then there is an induced* $g : \|A\|_n \to B$ *such that* $g(|a|_n) = f(a)$ *for any* $a : A$.

We refer the reader to [UFP13, 7.3] for more details on truncations.

The homotopy fiber of a function f at element b is defined as $\text{fib}_f(b) \overset{def}{=} \sum_{a:A} f(a) = b$. A function f has n-truncated homotopy fibers (or simply is an n-truncated function) when $\text{fib}_f(b)$ is in Type_n for any b. Again, we define some syntactic sugar. A function f is

—an *embedding* if f is (-1)-truncated

—a *surjection* if every fiber of f is merely inhabited (i.e $\|\text{fib}_f(y)\|$ holds for all y).

One can show [UFP13, Lemma 7.6.4] that any map f factors uniquely through $\text{Im}(f) \overset{def}{=} \sum_{y:B} \|\text{fib}_f(y)\|$ as a surjection followed by an embedding.

Following [RS15], it is possible to show that, for any homotopy level n and any type B, Type_n classifies subobjects of B with n-truncated homotopy fibers in the sense that there is an equivalence

$$\chi : \sum_{A:\mathrm{Type}} \sum_{f:A\to B} \prod_{b\in B} \text{Is-}n\text{-type } \mathrm{fib}_f(b) \xrightarrow{\sim} (B \to \mathrm{Type}_n)$$

such that the usual subobject classifier diagram ([UFP13, Theorem 4.8.4]) is a pullback. Therefore, in our construction, we will represent a subobject of a type B with n-truncated homotopy fibers either as a map $f : A \to B$ such that $\text{Is-}n\text{-type}\,\mathrm{fib}_f(b)$, or as a map $B \to \mathrm{Type}_n$.

2.2 Colimits in Homotopy Type Theory

One desireable construction we would like to consider is the Čech nerve (Section 2.3). In order to do so, this section presents a definition of colimits in a type theoretic setting. Following the definition of graphs and diagrams defined in [AKL15], we recall the definition of colimits of diagrams overs graphs presented in [RS15].

A colimit of a diagram D over a graph G is given by a type P that defines a cocone on D, plus the universal property that for any type X, the canonical map that transforms a function $f : P \to X$ to a cocone of D on X is an isomorphism.

Definition 2.3. Let G be a graph, and D be a diagram on G. Let P : Type together with

—a map $q_i : D_0(i) \to P$ for any vertex $i : G_0$, *i.e.* $q : \prod_{i:G_0} D_0(i) \to P$

—for any vertices $i, j : G_0$ and all edges $\phi : G_1(i,j)$, a path $p^{\phi}_{i,j} : q_j \circ D_1(\phi) = q_i$, *i.e.* $p : \prod_{i,j:G_0} \prod_{\phi:G_1(i,j)} q_j \circ D_1(\phi) = q_i$.

Then P is the *colimit* of D if for any other X : Type,

$$\mathrm{IsEquiv}\left(\lambda f : P \to X, \left(\lambda i,\ f \circ q_i\,;\, \lambda i\, j\, \phi,\ f(p^{\phi}_{i,j})\right)\right).$$

Using higher inductive types, every diagram D on a graph G admits a colimit in homotopy type theory.

In 5.2.1, we will need to know how colimits behave with respect to truncations. An answer is given by the following lemma.

Lemma 2.4. *Let D be a diagram, m a truncation index, and $P : \mathrm{Type}_m$ a colimit of D. Then, if $\|D\|_m$ is the same diagram as D where every type is m-truncated, P is a m-colimit*[1] *of $\|D\|_m$.*

The proof is quite straightforward: a cocone over D into P can be changed equivalently into a cocone over $\|D\|_m$ into $\|P\|_m$, using the elimination principle of truncations, and then we can show that the following diagram commutes for any $X : \mathrm{Type}_m$

$$\begin{tikzcd}
\|P\|_m \to X \arrow[r] \arrow[d, "\wr"] & \mathrm{cocone}(\|D\|_m, X) \\
P \to X \arrow[r, "\sim"] & \mathrm{cocone}(D, X) \arrow[u, "\wr"']
\end{tikzcd}$$

[1] P is a m-colimit if P satisfies the same property as in 2.3 when we replace Type by Type_m

2.3 On Giraud-Rezk-Lurie axioms

The Giraud-Rezk-Lurie axioms are the ∞-version of Giraud's axioms that characterize a topos. Namely, there are four axioms on a $(\infty, 1)$-category that have been shown to be equivalent to $(\infty, 1)$-topos axioms [Lur09, Chapter 6]. A consequence which we would like to use here is the fact that a surjection (*i.e.* (-1)-connected function) is the colimit of its Čech nerve. In [Bou16], the authors propose an analogue of this property: they give, for any map f, a diagram $C(f)$ whose colimit is $\mathrm{Im}(f)$.

This property will be essential in the proof that the construction $\square_{n+1}$, defined in Section 5.2.1, gives rise to a modality.

Definition 2.5. Let $f : X \to Y$ be a map. The coequalizer T_f of the kernel pair of f is the higher inductive type given by

—$t :\ X \to T_f$
—$\alpha :\ \forall a\, b : X,\ f(a) = f(b) \to t(a) = t(b)$
—$\alpha_1 :\ \forall a : X,\ \alpha(a, a, 1) = 1$

We view T_f as the coequalizer of $\sum_{a,b:X} f(a) = f(b) \underset{\pi_2}{\overset{\pi_1}{\rightrightarrows}} X$ preserving the identity. We call $\tilde{f}$ the map $T_f \to Y$ given by induction.

Then, the considered diagram $C(f)$ is the mapping telescope of the iterations of T.

Definition 2.6. Let f be a map from X to Y. Then the *iterated kernel pair* of f is given by the diagram $C(f) := X \xrightarrow{t} T_f \xrightarrow{t} T_{\tilde{f}} \longrightarrow \cdots$

Let's recall the main theorem:

Theorem 2.7 (Colimit of $C(f)$ [Bou16]). *For any morphism $f : X \to Y$, the colimit of $C(f)$ is $\mathrm{Im}(f)$, the image of f.*

3. MODALITIES

This section presents modalities in Homotopy Type Theory as defined in [UFP13] and later developed in [SS14, Shu15a]. We have added proofs of various properties of modalities when they are not already present in the literature. A truncated version of modalities, specific to our work, is then presented together with a discussion of the formalization.

Definition 3.1. A left exact modality is the data of

(1) A predicate $P : \mathrm{Type} \to \mathrm{HProp}$
(2) For every type A, a type $\bigcirc A$ such that $P(\bigcirc A)$
(3) For every type A, a map $\eta_A : A \to \bigcirc A$

such that

(4) For every types A and B, if $P(B)$ then

$$\begin{cases} (\bigcirc A \to B) & \to & (A \to B) \\ f & \mapsto & f \circ \eta_A \end{cases}$$

is an equivalence.

(5) for any A : Type and $B : A \to$ Type such that $P(A)$ and $\prod_{x:A} P(Bx)$, then $P\left(\sum_{x:A} B(x)\right)$

(6) for any A : Type and $x, y : A$, if $\bigcirc A$ is contractible, then $\bigcirc(x = y)$ is contractible.

Conditions (1) to (4) define a *reflective subuniverse*, (1) to (5) a *modality*.

Remark 3.2. The inverse of $- \circ \eta_A$ from point (4) will be denoted $\bigcirc_{\mathrm{rec}} : (A \to B) \to (\bigcirc A \to B)$, and its computation rule $\bigcirc^{\beta}_{\mathrm{rec}} : \prod_{f:A\to B} \prod_{x:A} \bigcirc_{\mathrm{rec}}(f)(\eta_A x) = fx$.

If $\bigcirc$ is a modality, the type of modal types will be denoted $\mathrm{Type}^{\bigcirc}$. Let us fix a left-exact modality $\bigcirc$ for the rest of this section. A modality acts functorialy on Type, in the sense that

LEMMA 3.3 (FUNCTORIALITY OF MODALITIES). *Let A, B : Type and $f : A \to B$. Then there is a map $\bigcirc f : \bigcirc A \to \bigcirc B$. Moreover*

—*For all A, B : Type and $f : A \to B$, $\bigcirc f \circ \eta_A = \eta_B \circ f$.*

—*For all X : Type, $Y, Z : \mathrm{Type}^{\bigcirc}$, $f : X \to Y$ and $g : Y \to Z$,*

$$g \circ \bigcirc_{rec}(f) = \bigcirc_{rec}(g \circ f).$$

—*For all X, Y : Type, $Z : \mathrm{Type}^{\bigcirc}$, $f : X \to Y$ and $g : Y \to Z$,*

$$\bigcirc_{rec}(g) \circ \bigcirc f = \bigcirc_{rec}(g \circ f).$$

—*For all X, Y, Z : Type, $f : X \to Y$ and $g : Y \to Z$,*

$$\bigcirc(g \circ f) = \bigcirc g \circ \bigcirc f.$$

—*If* IsEquiv f, *then* IsEquiv $\bigcirc f$.

PROOF. We define $\bigcirc f$ by

$$\bigcirc f \stackrel{def}{=} \bigcirc_{\mathrm{rec}}(\eta_B \circ f).$$

Then

—By the computation principle of $\bigcirc_{\mathrm{rec}}$, the first point is obvious.

—As Z is modal and both functions are $\bigcirc X \to Z$, it suffices to show that

$$g \circ \bigcirc_{\mathrm{rec}} g \circ \eta_X = \bigcirc_{\mathrm{rec}}(g \circ f) \circ \eta_X.$$

But both sides are equal to $g \circ f$ using computational rules.

—We will show that each side is equal to

$$\varphi \stackrel{def}{=} \bigcirc_{\mathrm{rec}}((\bigcirc_{\mathrm{rec}} g) \circ (\eta_Y \circ f)).$$

The left-hand side is equal to φ using the previous point, applied to $\eta_Y \circ f$. For the right-hand side, it suffices to show that $g \circ f = \bigcirc_{\mathrm{rec}}(g) \circ \eta_Y \circ f$, which is exactly the computation rule of $\bigcirc_{\mathrm{rec}}$ composed with f.

—This is a particular case of the previous point, applied to f and $\eta_Z \circ g$.

—If f is an equivalence, an obvious inverse for $\bigcirc f$ is $\bigcirc(f^{-1})$.

□

Proposition 3.4. *Any left-exact modality* $\bigcirc$ *satisfies the following properties*[2].

- *(R) A is modal if and only if* η_A *is an equivalence.*
- *(R)* $\mathbf{1}$ *is modal.*
- *(R)* $\mathrm{Type}^{\bigcirc}$ *is closed under dependent products,* i.e. $\prod_{x:A} B\,x$ *is modal as soon as all* $B\,x$ *are modal.*
- *(R) For any types A and B, the map*

$$\bigcirc(A \times B) \to \bigcirc A \times \bigcirc B$$

is an equivalence.
- *(R) If A is modal, then for all* $x, y : A$, $(x = y)$ *is modal.*
- *(M) For every type A and* $B : \bigcirc(A) \to \mathrm{Type}^{\bigcirc}$, *then*

$$- \circ \eta_A : \begin{array}{ccc} \prod_{z:\bigcirc A} B\,z & \longrightarrow & \prod_{a:A} B(\eta_A\, a) \\ f & \longmapsto & f \circ \eta_A \end{array}$$

is an equivalence.
- *(M) If* $A, B : \mathrm{Type}$ *are modal, then so are* $\mathrm{Is}\text{-}n\text{-}\mathrm{type}\, A$, $A \simeq B$ *and* $\mathrm{IsEquiv}\, f$ *for all* $f : A \to B$.
- *(L) If* $X, Y : \mathrm{Type}$ *and* $f : X \to Y$, *then the map*

$$\bigcirc\,(\mathrm{fib}_f(y)) \to \mathrm{fib}_{\bigcirc f}(\eta_B y)$$

is an equivalence, and the following diagram commutes

$$\begin{tikzcd} \mathrm{fib}_f(y) \arrow[r, "\eta"] \arrow[d, "\gamma"'] & \bigcirc\,(\mathrm{fib}_f(y)) \arrow[dl] \\ \mathrm{fib}_{\bigcirc f}(\eta_B y) & \end{tikzcd}$$

Remark 3.5. Again, the inverse of $- \circ \eta_A$ will be denoted $\bigcirc_{\mathrm{ind}} : \prod_{a:A} B(\eta_A a) \to \prod_{z:\bigcirc A} B\,x$, and its computation rule $\bigcirc^{\beta}_{\mathrm{ind}} : \prod_{f:\prod_{a:A} B(\eta_A\, a)} \prod_{x:A} \bigcirc_{\mathrm{ind}}(f)(\eta_A x) = f x$

Proof. —If η_A is an equivalence, then $A \simeq \bigcirc A$, so A is modal.
Now if A is modal, then we have $\bigcirc_{\mathrm{rec}}(\mathrm{id}) : \bigcirc A \to A$, and one can easily check that it is an inverse to η_A.

—Given the previous proof, it suffices to prove that $\eta_{\mathbf{1}}$ is an equivalence. The only way to inhabit $\bigcirc\mathbf{1} \to \mathbf{1}$ is with $\lambda\, x, \star$. It is straightforward to check that this forms an equivalence.

—This is [UFP13, Theorem 7.7.2].

—This is [UFP13, Corollary 7.7.3].

[2]Properties needing only a reflective subuniverse are annotated by (R), a modality by (M), a left-exact modality by (L)

—Again, it suffice to show that $\eta_{x=y}$ is an equivalence. We begin by showing that

$$(\lambda\_ : \bigcirc(x = y), x) = (\lambda\_ : \bigcirc(x = y), y).$$

As A is modal, η_A is an equivalence, as well as $\mathrm{ap}_{\eta_A} : x = y \to \bigcirc(x = y)$. Thus, it suffices to show that

$$(\lambda\_ : \bigcirc(x = y), x) \circ \mathrm{ap}_{\eta_A}) = (\lambda\_ : \bigcirc(x = y), y) \circ \mathrm{ap}_{\eta_A},$$

and the latter is obvious using functional extensionality. Now, applying the just proved equality to any $u : \bigcirc(x = y)$ yields $x = y$. One can prove that this defines an inverse to $\eta_{x=y}$.

—This is [UFP13, Theorem 7.7.7].

—We show that Is-n-type A is modal by induction on the truncation level n.
If $n = -2$, we have Is-n-type $A \simeq \sum_{a:A} \prod_{b:A} b = a$. The latter is modal using stability by dependent sums, dependent products and paths type.
Now, if for every A, Is-n-type A is modal, then Is-$(n+1)$-type A is equivalent to

$$\prod_{x,y:A} \text{Is-}n\text{-type}\, x = y.$$

Again, using stability by dependent products and the induction hypothesis, the latter is modal.
The facts that $A \simeq B$ and IsEquiv f for any modal types A, B and map $f : A \to B$ are modal are technical, but don't involve new methods. They can be found in the formalization.

—It is straightforward to define a map

$$\phi : \sum_{x:X} fx = y \to \sum_{x:\bigcirc X} \bigcirc fx = \eta_Y y,$$

using η functions. We will use the following lemma to prove that the function induced by ϕ defines an equivalence:

Lemma 3.6. *Let X : Type, Y : Type$^{\bigcirc}$ and $f : X \to Y$. If for all $y : Y$, $\bigcirc(\mathrm{fib}_f(y))$ is contractible, then the function $\bigcirc X \to Y$ induced by f is an equivalence.*

Hence we just need to check that every $\bigcirc$-fiber $\bigcirc(\mathrm{fib}_\phi(x;p))$ is contractible. Technical transformations allow one to prove

$$\mathrm{fib}_\phi(x;p) \simeq \mathrm{fib}_s(y;p^{-1})$$

for

$$s : \begin{array}{ccc} \mathrm{fib}_{\eta_X}(x) & \longrightarrow & \mathrm{fib}_{\eta_Y}(\bigcirc f\, x) \\ (a,q) & \longmapsto & (f\, a, -) \end{array}$$

But left-exactness allows to characterize the contractibility of fibers:

Lemma 3.7. *Let A, B : Type. Let $f : A \to B$. If $\bigcirc A$ and $\bigcirc B$ are contractible, then so is $\bigcirc(\mathrm{fib}_f(b))$ for any $b : B$.*

Thus, we just need to prove that $\bigcirc(\mathrm{fib}_{\eta_X}(a))$ and $\bigcirc(\mathrm{fib}_{\eta_Y}(b))$ are contractible. But one can check that η maps always satisfy this property. Finally, $\bigcirc(\mathrm{fib}_s(y;p^{-1}))$ is contractible, so $\bigcirc(\mathrm{fib}_\phi(x;p))$ also, and the result is proved.

□

Let us finish these properties by the following proposition, giving an equivalent characterization of left-exactness.

PROPOSITION 3.8. *Let $\bigcirc$ be a modality. Then $\bigcirc$ is left-exact if and only if $\bigcirc$ preserves path spaces,* i.e.

$$\prod_{A:\mathrm{Type}} \prod_{x,y:A} \mathrm{IsEquiv}(\bigcirc(\mathrm{ap}_{\eta_A}))$$

where $\bigcirc(\mathrm{ap}_{\eta_A}) : \bigcirc(x = y) \to \eta_A x = \eta_A y$.

PROOF. We will rather prove something slightly more general, using an encode-decode proof [UFP13, Section 8.9]; we will characterize, for a type A and a fixed inhabitant $x : A$ the type

$$\eta_A x = y$$

for any $y : \bigcirc A$.

Let $\mathrm{Cover} : \bigcirc A \to \mathrm{Type}^{\bigcirc}$ be defined by induction by

$$\mathrm{Cover}(y) \stackrel{def}{=} \bigcirc_{\mathrm{rec}}(\lambda y,\, \bigcirc(x = y)).$$

Note that for any $y : \bigcirc A$, $\mathrm{Cover}(y)$ is always modal. We will show that $\eta_A x = y \simeq$ *Cover*(y). Now, let $\mathrm{Encode} : \prod_{y:\bigcirc A} \eta_A x = y \to \mathrm{Cover}(y)$ be defined by

$$\mathrm{Encode}(y,p) \stackrel{def}{=} \mathrm{transport}^{p}_{\mathrm{Cover}} \left(\mathrm{transport}^{\bigcirc^{\beta}_{\mathrm{rec}}((\lambda z,\, \bigcirc(x=z)),x)}_{\mathrm{idmap}}(\eta_{x=x} 1)\right)$$

and $\mathrm{Decode} : \prod_{y:\bigcirc A} \mathrm{Cover}(y) \to \eta_A x = y$ by

$$\mathrm{Decode} \stackrel{def}{=} \bigcirc_{\mathrm{ind}} \left(\lambda y\, p,\, \bigcirc(\mathrm{ap}_{\eta_A}) \left(\mathrm{transport}^{\bigcirc^{\beta}_{\mathrm{rec}}((\lambda z,\, \bigcirc(x=y)),y)}_{\mathrm{idmap}}\, p\right)\right)$$

Then one can show, using $\bigcirc$-induction and path-induction, that for any $y : \bigcirc A$, $\mathrm{Encode}(y, -)$ and $\mathrm{Decode}(y, -)$ are each other inverses. Then, taking $y' = \eta_A y$, we have just shown that $\eta_A x = \eta_A y \simeq \mathit{Cover}(\eta_A y)$, which is itself equivalent, by $\bigcirc^{\beta}_{\mathrm{rec}}$, to $\bigcirc(x = y)$. It is straightforward to check that the composition $\bigcirc(x = y) \to \mathrm{Cover}(\eta_A y) \to \eta_A x = \eta_A y$ is exactly $\bigcirc(\mathrm{ap}_{\eta_A})$.

Now, let us prove the backward implication. Let A be a type such that $\bigcirc A$ is contractible, and $x, y : A$. As $\eta_A x, \eta_A y : \bigcirc A$, we know that $\eta_A x = \eta_A y$ is contractible. But as $\eta_A x = \eta_A y \simeq \bigcirc(x = y)$ by assumption, $\bigcirc(x = y)$ is also contractible. □

As this whole paper deals with truncation levels, it should be interesting to see how they are changed under a modality. We already know that if a type T is (-2)-truncated, *i.e.* contractible, then it is unchanged by the reflector:

$$\bigcirc T \simeq \bigcirc \mathbf{1} \simeq \mathbf{1} \simeq T.$$

Thus, Type_{-2} is closed by any reflective subuniverse. Now, let T : HProp. To check that $\bigcirc T$ is an h-proposition, it suffices to check that

$$\prod_{x,y:\bigcirc T} x = y$$

For any $x : \bigcirc T$, the type $\prod_{y:\bigcirc T} x = y$ is modal, as all $x = y$ are; by the same argument, $\prod_{x:\bigcirc T} x = y$ is modal too for any $y : \bigcirc T$. Using twice the dependent eliminator of $\bigcirc$, it now suffices to check that

$$\prod_{x,y:T} \eta_T x = \eta_T y.$$

As T is supposed to be an h-proposition, this is true. It suffices to state

Lemma 3.9. *For any modality,* Type_{-1} *is closed under the reflector* $\bigcirc$*,* i.e.

$$\prod_{P:\mathrm{HProp}} \mathrm{IsHProp}(\bigcirc P).$$

A simple induction on the truncation level, together with the left-exactness property allows to state

Lemma 3.10. *For any left-exact modality, all* Type_p *are closed under the reflector* $\bigcirc$*,* i.e.

$$\prod_{P:\mathrm{Type}_p} \text{Is-}p\text{-type}(\bigcirc P).$$

3.1 Examples of modalities

3.1.1 *The identity modality.* Let us begin with the most simple modality one can imagine: the one doing nothing. We can define it by letting $\bigcirc A \stackrel{def}{=} A$ for any type A, and $\eta_A \stackrel{def}{=}$ idmap. Obviously, the desired computation rules are satisfied, so that the identity modality is indeed a left-exact modality.

It might sound useless to consider such a modality, but it can be precious when looking for properties of modalities: if it does not hold for the identity modality, it cannot hold for an abstract one.

3.1.2 *Truncations.* The first class of non-trivial examples might be the *truncations* modalities, as described in [UFP13, Section 7.3].

3.1.3 *Double negation modality*

Proposition 3.11. *The double negation modality* $\bigcirc A \stackrel{def}{=} \neg\neg A$ *is a modality.*

Proof. We define the modality with

(1) We will define the predicate P later.
(2) $\bigcirc$ is defined by $\bigcirc A = \neg\neg A$
(3) We want a term η_A of type $A \to \neg\neg A$. The term

$$\eta_A \stackrel{def}{=} \lambda\, x : A,\, \lambda\, y : \neg A,\, y\, a$$

matches this requirement.
Now, we can define P to be exactly $\prod_{A:\mathrm{Type}} \mathrm{IsEquiv}\, \eta_A$.

(4) Let A, B : Type, and $\varphi : A \to \neg\neg B$. We want to extend it into $\psi : \neg\neg A \to \neg\neg B$. Let $a : \neg\neg A$ and $b : \neg B$. Then $a(\lambda\, x : A,\, \varphi\, x\, b) : \mathbf{0}$, as wanted. One can check that it forms an equivalence.
(5) Let A : Type and $B : A \to$ Type such that $P(A)$ and $\prod_{a:A} P(B\, a)$. There is a map

$$\sum_{x:A} B\, x \to A$$

thus by the previous point, we can extend it into

$$\kappa : \neg\neg \sum_{x:A} B\, x \to A.$$

It remains to check that for any $x : \neg\neg \sum_{x:A} B\, x$, $B(\kappa\, x)$.
But the previous map can be easily extended to the dependent case, and thus it suffices to show that for all $x : \sum_{x:A} B\, x$, $B(\kappa(\eta\, x)$. As $\kappa \circ \eta = \mathrm{idmap}$, the goal is solved by $\pi_2 x$.

□

Unfortunately, it follows that the only types which can be modal are h-propositions, as they are equivalent to their double negation which is always an h-proposition. Thus, the type of modal types consists only of h-propositions, which is not satisfactory. The main purpose of this paper is to extend this modality into less destructive one.

3.2 Toward a new type theory

We suppose here that $\bigcirc$ is a left-exact modality such that $\mathrm{Type}^{\bigcirc}$ is modal. This is for example the case when the modality is *accessible* (see [BGL+17] for definition and proof). We call a modality $\bigcirc$ *consistent* if $\bigcirc\mathbf{0}$ is empty.

PROPOSITION 3.12. *The modal universe* $\mathrm{Type}^{\bigcirc}$ *is non-trivial (non contractible) if the type* $\bigcirc\mathbf{0}$ *is empty.*

PROOF. By condition (iv) of Definition 3.1, $\bigcirc\mathbf{0}$ is an initial object of $\mathrm{Type}^{\bigcirc}$, and thus corresponds to false for modal mere proposition. As $\bigcirc\mathbf{1} = \mathbf{1}$, $\mathrm{Type}^{\bigcirc}$ is non-trivial when $\bigcirc\mathbf{0} \neq \mathbf{1}$, that is when there is no proof of $\bigcirc\mathbf{0}$. □

In topos theory, Lawvere-Tierney topologies give rise to subtoposes $\mathrm{Sh}_j\mathcal{E} \hookrightarrow \mathcal{E}$; actually, every subtopos $\mathcal{F} \hookrightarrow \mathcal{E}$ comes from a Lawvere-Tierney topology [MM92, Corollary VII.4.7]. In the same way, left-exact modalities should induce sub-type theories, and we should be able to exhibit a translation from this sub-type theory into the ground type theory, as in [JLP+16].

3.3 Truncated modalities

As for colimits, we define a truncated version of modalities, in order to use it in section 5. Basically, a truncated modality is the same as a modality, but restricted to Type_n.

Definition 3.13 (TRUNCATED MODALITY). Let $n \geq -1$ be a truncation index. A left exact modality at level n is the data of

(1) A predicate $P : \mathrm{Type}_n \to \mathrm{HProp}$

(2) For every n-truncated type A, a n-truncated type $\bigcirc A$ such that $P(\bigcirc A)$
(3) For every n-truncated type A, a map $\eta_A : A \to \bigcirc A$

such that

(4) For every n-truncated types A and B, if $P(B)$ then

$$\begin{cases} (\bigcirc A \to B) & \to & (A \to B) \\ f & \mapsto & f \circ \eta_A \end{cases}$$

is an equivalence.
(5) for any $A : \mathrm{Type}_n$ and $B : A \to \mathrm{Type}_n$ such that $P(A)$ and $\prod_{x:A} P(Bx)$, then $P\left(\sum_{x:A} B(x)\right)$
(6) for any $A : \mathrm{Type}_n$ and $x, y : A$, if $\bigcirc A$ is contractible, then $\bigcirc(x = y)$ is contractible.

Properties of truncated left-exact modalities described in 3.4 are still true when restricted to n-truncated types, except the one that does not make sense: $\mathrm{Type}_n^{\bigcirc}$ cannot be modal, as it is not even a n-truncated type.

3.4 Formalization

Let us discuss here the formalization of the theory of modalities. General modalities are formalized in the Coq/HoTT library [BGL+17], thanks to the work of Mike Shulman [Shu]. The formalization might seem straightforward, but the universe levels (at least, their automatic handling by Coq) become a major issue. Hence, we have to explicitly give the universe levels and their constraints in a large part of the library. For example, the reflector $\bigcirc$ of a modality is defined in [BGL+17] as

$$\bigcirc : \mathrm{Type}^i \to \mathrm{Type}^i;$$

it maps any universe to itself.

In section 5, we will need a slightly more general definition of modality. The actual definitions stay the same, but the universes constraints we consider change. The reflector $\bigcirc$ will now have type

$$\bigcirc : \mathrm{Type}^i \to \mathrm{Type}^k, \quad i \leqslant k;$$

it maps any universe to a possibly higher one. Other components of the modality will have types

$$\begin{array}{lcll} P & : & \mathrm{Type}^i \to \mathrm{HProp}^k, & i \leqslant k \\ \eta & : & \prod_{A:\mathrm{Type}^i} \bigcirc A : \mathrm{Type}^k, & i \leqslant k \\ - & : & \prod_{A:\mathrm{Type}^i} \prod_{B:\mathrm{Type}^j} \prod_{h:P(B)} \mathrm{IsEquiv}(- \circ \eta_A), & i, j \leqslant k \end{array}$$

Fortunately, this change is small enough as to preserve usual properties of modalities. Of course, the examples of modalities mapping any universe to itself are still examples of generalized modalities, simply ones which do not make use of the possibility of inhabiting a higher universe.

We would like to have the same generalization for truncated modalities. However, in this case many new universe levels appear, mostly because in $\mathrm{Type}_n = \sum_{T:\mathrm{Type}} \text{Is-}n\text{-type}\, T$, and Is-$n$-type each comes with its own universe. As a result, handling "by hand" so many universes together with their constraints quickly gets out of control. One idea to fix this issue could be to use *resizing rules* [Voe], allowing h-propositions to live in the smallest universe. We could then get rid of the universes generated by Is-n-type, and treat the truncated modality exactly as any other generalized modality.

In our formalization, we decided to work with the `type-in-type` Coq option, to avoid any issue with universes.

4. SHEAVES IN TOPOSES

In this section, we will work in an arbitrary topos rather than in type theory. The next section will present a generalization of the results presented here.

Let us fix for the whole section a topos $\mathcal{E}$, with subobject classifier Ω. A *Lawvere-Tierney topology* on $\mathcal{E}$ is a way to modify slightly truth values of $\mathcal{E}$. It allows one to speak about *locally true* things instead of *true* things.

Definition 4.1 (Lawvere-Tierney topology [MM92]). A Lawvere-Tierney topology is an endomorphism $j : \Omega \to \Omega$ preserving $\top$ ($j\ \top = \top$), idempotent ($j \circ j = j$) and commuting with products ($j \circ \wedge = \wedge \circ (j, j)$).

A classical example of Lawvere-Tierney topology is given by double negation. Other examples are given by Grothendieck topologies, in the sense of the following:

Theorem 4.2 ([MM92, Theorem V.1.2]). *Every Grothendieck topology J on a small category $\mathbf{C}$ determines a Lawvere-Tierney topology j on the presheaf topos $\mathbf{Sets}^{\mathbf{C}^{\mathbf{op}}}$.*

Any Lawvere-Tierney topology j on $\mathcal{E}$ induces a closure operator $A \mapsto \overline{A}$ on subobjects. If we see a subobject A of E as a characteristic function χ_A, the closure $\overline{A}$ corresponds to the subobject of E whose characteristic function is

$$\chi_{\overline{A}} = j \circ \chi_A.$$

A subobject A of E is said to be dense when $\overline{A} = E$.

Next, we are interested in objects of $\mathcal{E}$ for which it is impossible to make a distinction between objects and their dense subobjects, *i.e.* for which "true" and "locally true" coincide. Such objects are called *sheaves*, and are defined by

Definition 4.3 (Sheaves[MM92, Section V.2]). On object F of $\mathcal{E}$ is a sheaf (or j-sheaf) if for every dense monomorphism $m : A \hookrightarrow E$ in $\mathcal{E}$, the canonical map $\mathrm{Hom}_{\mathcal{E}}(E, F) \to \mathrm{Hom}_{\mathcal{E}}(A, F)$ is an isomorphism.

One can show that $\mathrm{Sh}_{\mathcal{E}}$, the full sub-category of $\mathcal{E}$ given by sheaves, is again a topos, with classifying object

$$\Omega_j = \{P \in \Omega \mid jP = P\}.$$

Lawvere-Tierney sheafification is a way to build a left adjoint $\mathbf{a}_j$ to the inclusion $\mathcal{E} \hookrightarrow \mathrm{Sh}_{\mathcal{E}}$, exhibiting $\mathrm{Sh}_{\mathcal{E}}$ as a reflective subcategory of $\mathcal{E}$. In particular, this implies that logical principles valid in $\mathcal{E}$ are still valid in $\mathrm{Sh}_{\mathcal{E}}$.

For any object E of $\mathcal{E}$, $\mathbf{a}_j(E)$ is defined as in the following diagram

$$
\begin{tikzcd}
E \arrow[rr, "\{\cdot\}_E"] \arrow[d, "\theta_E"'] & & \Omega^E \arrow[d, "j^E"] \\
E' \arrow[rr, hook] \arrow[dr, "\text{closure}"'] & & \Omega_j^E \\
& \mathbf{a}_j(E) \arrow[ur] &
\end{tikzcd}
$$

The proof that $\mathbf{a}_j$ defines a left adjoint to the inclusion can be found in [MM92].

One classical example of use of sheafification is the construction, from any topos, of a boolean topos negating the continuum hypothesis. More precisely:

Theorem 4.4 (Negation of CH [MM92, Theorem VI.2.1]). *There exists a Boolean topos satisfying the axiom of choice, in which the continuum hypothesis fails.*

The proof actually follows almost exactly the famous construction by Paul Cohen of a model of ZFC negating the continuum hypothesis [Coh66]. Together with the model of constructible sets $\mathfrak{L}$ by Kurt Gödel [Gö40], this proves that CH is independent of ZFC, solving Hilbert's first problem.

5. SHEAVES IN HOMOTOPY TYPE THEORY

The idea of this section is to consider sheafification in toposes as only the first step towards sheafification in type theory. We remark that axioms for a Lawvere-Tierney topology on the subobject classifier Ω of a topos are very close to those of a modality on Ω. We will use this idea extensively, applying it to every subobject classifier Type_n as described in section 2. The subobject classifier Ω of a topos is seen as the object of *truth values* of the topos, which corresponds to the type HProp in our setting; the topos itself is considered proof irrelevant, corresponding to our HSet. Sheafification in toposes, when translated to the setting of homotopy type theory, is thus a way to build from a left-exact modality on HProp, a left-exact modality on HSet. Our hope in this section is to iterate this construction, extending a left-exact modality on HSet to one on Type_1, and so on.

The first thing we can note is that such a construction will not allow to reach every type: it is known that there exist types with no finite truncation level [UFP13, Example 8.8.6]. Worse, some types are not even the limit of their successive truncations, even in an hypercomplete setting [MV99]. This suggests that defining a sheafification functor for all truncated types won't give (at least easily) a sheafification functor on the whole of Type. Another issue which must be pointed out is the complexity of proofs. Whereas in a topos-theoretic setting everything is proof-irrelevant, this is no longer the case in a higher setting, forcing us to reason about witnesses for results which were previously true on the nose. This will oblige us to write long and technical proofs of coherence, and occasionally to completely modify some lemmas, such as Proposition [MM92, Theorem IV.7.8], stating that epimorphisms are coequalizers of their kernel pairs.

The main idea is thus to follow as closely as possible the topos-theoretic construction, and change it as few times as possible to make it work in our higher setting.

Note that, as the principles we want to add are inherited directly from the HProp level, the extension to all truncated types is automatic. The choice of the left-exact modality on HProp is thus crucial. For the rest of the section, we fix one, denoted $\bigcirc_{-1}$. The reader can think of the double negation $\bigcirc_{\neg\neg}$ defined in 3.1.3. We will define, by induction on the truncation level, left-exact modalities on all Type_n, as in the following theorem.

Theorem 5.1. *The sequence defined by induction by*

$$\begin{array}{l} \bigcirc : \forall\ (n : nat),\ \mathrm{Type}_n \to \mathrm{Type}_n \\ \bigcirc_{-1}\ (T) \quad \textit{given} \\ \bigcirc_{n+1}(T) \overset{def}{=} \displaystyle\sum_{u:T\to\mathrm{Type}_n^{\bigcirc}} \bigcirc_{-1} \left\| \sum_{a:T} u = (\lambda t,\ \ \bigcirc_n\ (a = t)) \right\| \end{array}$$

defines a sequence of left-exact modalities, coherent with each others in the sense that the following diagram commutes for any $P : \mathrm{Type}_n$, where $\hat{P}$ is P seen as an inhabitant of Type_{n+1}.

$$\begin{tikzcd} P \arrow[r, "\sim"] \arrow[d, "\eta_n"'] & \widehat{P} \arrow[d, "\eta_{n+1}"] \\ \bigcirc_n P \arrow[r, "\sim"] & \bigcirc_{n+1}\widehat{P} \end{tikzcd}$$

In what follows, formalized results are indicated by the name of the result in the library in `THIS SPECIAL FONT`.

5.1 Sheaf theory

Let n be a truncation index greater that -1, and $\bigcirc_n$ be the left-exact modality given by our induction hypothesis. As in the topos-theoretic setting, we will define what it means for a type to be a n-sheaf (or just “sheaf”, if the context is clear), and consider the reflective subuniverses of these sheaves; the reflector will exactly be the sheafification functor. The main issue in order to arrive at the “right” definition is the choice of the subobject classifier in which dense subobjects will be chosen: two choices appear, HProp and Type_n; we will actually use both. This principle guiding our choice is that the type of all n-sheaves should be a $(n+1)$-sheaf.

From the modality $\bigcirc_n$, one can build a *closure operator.*

Definition 5.2 (`CLOTURE`,`CLOSED`,`EnJ`). Let E be a type.

—The *closure* of a subobject of E with n-truncated homotopy fibers (or n-subobject of E, for short), classified by $\chi : E \to \mathrm{Type}_n$, is the subobject of E classified by $\bigcirc_n \circ \chi$.

—An n-subobject of E classified by χ is said to be *closed in* E if it is equal to its closure, *i.e.* if $\chi = \bigcirc_n \circ \chi$.

—An n-subobject of E classified by χ is said to be *dense in* E if its closure is E, *i.e.* if $\bigcirc_n \circ \chi = \lambda e, \mathbf{1}$

Topos-theoretic sheaves are characterized by a property of existence and uniqueness, which will be translated, as usual, into a proof that a certain function is an equivalence.

Definition 5.3 (Restriction (E_to_χmono_map, E_to_χ_map)). Let E, F : Type and $\chi : E \to \text{Type}$. We define the *restriction map* Φ_E^χ as

$$\Phi_E^\chi : \begin{array}{ccc} E \to F & \longrightarrow & \sum_{e:E} \chi e \to F \\ f & \longmapsto & f \circ \pi_1 \end{array}.$$

Here, we need to distinguish between dense (-1)-subobjects, which will be used in the definition of sheaves, and dense n-subobjects, which will be used in the definition of separated types.

Definition 5.4 (Separated Type (separated)). A type F in Type_{n+1} is *separated* if for any type E, and all dense n-subobject of E classified by χ, Φ_E^χ is an embedding.

From the point of view of topos theory, this means that given a map $\sum_{e:E} \chi\, e \to F$, if there is an extension $\tilde{f} : E \to F$, then it is unique, as in

$$\begin{tikzcd} \sum_{e:E} \chi e \arrow[r, "f"] \arrow[d, "\pi_1"'] & F \\ E \arrow[ur, dashed, "!"'] & \end{tikzcd}$$

Definition 5.5 (Sheaf (Snsheaf_struct)). A type F of Type_{n+1} is a $(n+1)$-*sheaf* if it is separated, and for any type E and all dense (-1)-subobject of E classified by χ, Φ_E^χ is an equivalence.

In this case, this means that given a map $f : \sum_{e:E} \chi\, e \to F$, one can extend it uniquely to $\tilde{f} : E \to F$, as in

$$\begin{tikzcd} \sum_{e:E} \chi e \arrow[r, "f"] \arrow[d, "\pi_1"'] & F \\ E \arrow[ur, dashed, "\exists!"'] & \end{tikzcd}$$

Note that these definitions are almost the same as those given in in [MM92]. The main difference is that separated is defined for n-subobjects, while sheaf only for (-1)-subobjects. It might seem bizarre to make such a distinction, but the following proposition gives a better understanding of the situation.

Proposition 5.6 (nj_paths_separated). *A type F is* Type_{n+1} *is separated if, and only if all its path types are n-modal.*

Proof. Let F : Type$_{n+1}$ a separated type, and $a, b : F$. We want to find an inverse to $\eta_{a=b}$. We consider the following diagram

$$\begin{tikzcd}
\sum_{(a,b):F\times F} a = b \arrow[r, "\mathrm{fst}\circ\pi_1"] \arrow[d, "\iota"'] & F \\
\sum_{(a,b):F\times F} \bigcirc(a = b) \arrow[ur, shift left, "\mathrm{snd}\circ\pi_1"] \arrow[ur, shift right, "\mathrm{fst}\circ\pi_1"'] &
\end{tikzcd}$$

Both $\mathrm{fst} \circ \pi_1$ and $\mathrm{snd} \circ \pi_1$ make the diagram commute, hence they are equal:

$$\prod_{a,b:f} \bigcirc(a = b) \to a = b.$$

One can check that this defines an inverse to $\eta_{a=b}$.

Conversely, let E : Type, $\chi : E \to \mathrm{Type}_n$ and $f, g : E \to F$ such that $p : f \circ \pi_1 = g \circ \pi_1$. Using functional extensionality, we want to show that $f\,x = g\,x$ for any $x : E$. As $\sum_{e:E} \chi\, e$ is a dense n-subobject of E, $\bigcirc_n(\chi\, x)$ is inhabited. By hypothesis, $f\,x = g\,x$ is modal, thus by induction principle of $\bigcirc$, we can suppose that we can inhabit $\chi\, x$ with a term w. We can then apply the equality p to the dependent pair $(x; w)$ to have $f\,x = g\,x$, as required. □

A $(n + 1)$-sheaf is hence just a type satisfying the usual property of sheaves (*i.e.* existence of uniqueness of arrow extension from dense (-1)-subobjects), with the condition that all its path types are n-sheaves. This is a way to force the compatibility of the modalities we are defining.

One can check that the property IsSeparated (resp. IsSheaf) is HProp: given a X : Type$_{n+1}$, there is at most one way for it to be separated (resp. a sheaf). In particular, when one needs to prove equality between two sheaves, it suffices to show the equality between the underlying types.

As mentioned earlier, these definitions allow us to prove the fundamental property that the type of all n-sheaves is itself a $(n + 1)$-sheaf .

Proposition 5.7 (`nType_j_Type_is_SnType_j_Type`). *Type$_n^{\bigcirc}$ is a $(n+1)$-sheaf.*

Proof. We have two things to prove here: separation, and the sheaf property.

—Let E : Type and $\chi : E \to$ Type, dense in E. Let $\phi_1, \phi_2 : E \to \mathrm{Type}_n^{\bigcirc}$, such that $\phi_1 \circ \pi_1 = \phi_2 \circ \pi_1$ and let $x : E$. We show $\phi_1(x) = \phi_2(x)$ using univalence.
As χ is dense, we have a term $m_x : \bigcirc_n(\chi\, x)$. But as $\phi_2(x)$ is modal, we can obtain a term $h_x : \chi\, x$. As ϕ_1 and ϕ_2 are equal on $\sum_{e:E} \chi\, e$, we have an arrow $\phi_1(x) \to \phi_2(x)$. The same method leads to an arrow $\phi_2(x) \to \phi_1(x)$, and one can prove that they are each other inverse.

—Now, we prove that Type$_n^{\bigcirc}$ is a sheaf. Let E : Type and $\chi : E \to$ HProp, dense in E. Let $f : \sum_{e:E} \chi\, e \to \mathrm{Type}_n^{\bigcirc}$. We want to extend f into a map $E \to \mathrm{Type}_n^{\bigcirc}$. We define g as $g(e) = \bigcirc_n(\mathrm{fib}_\phi(e))$, where

$$\phi : \sum_{b:\sum_{e:E} \chi\, e} (f\, b) \to E$$

defined by $\phi(x) = (x_1)_1$. Using the following lemma, one can prove that the map $f \mapsto g$ defines an inverse of Φ_E^{χ}.

Lemma 5.8 (nj_fibers_compose). *Let A, B, C : Type_n, $f : A \to B$ and $g : B \to C$. Then if all fibers of f and g are n-truncated, then*

$$\prod_{c:C} (\bigcirc_n(\mathrm{fib}_{g\circ f}(c))) \simeq \bigcirc_n \left(\sum_{w:\mathrm{fib}_g(c)} \bigcirc_n(\mathrm{fib}_f(w_1)) \right).$$

Proof. This is just a modal counterpart of the property characterizing the fibers of composition of functions. □

Another fundamental property on sheaves we will need is that the type of (dependent) functions is a sheaf as soon as its codomain is a sheaf.

Proposition 5.9 (dep_prod_SnType_j_Type). *If A : Type_{n+1} and $B : A \to \mathrm{Type}_{n+1}$ such that for any $a : A$, $(B\ a)$ is a sheaf, then $\prod_{a:A} B\,a$ is a sheaf.*

Proof. Again, when proving equivalences, we will only define the maps. The proofs of section and retraction are technical, not really interesting, and present in the formalization.

—*Separation:* Let E : Type and $\chi : E \to \mathrm{Type}_n$ dense in E. Let $\phi_1, \phi_2 : E \to \prod_{a:A} B\,a$ equal on $\sum_{e:E} \chi\,e$ *i.e.* such that $\phi_1 \circ \pi_1 = \phi_2 \circ \pi_1$. Then for any $a : A$, $(\lambda x : E,\ \phi_1(x, a))$ and $(\lambda x : E,\ \phi_2(x, a))$ coincide on $\sum_{e:E}(\chi\,e)$, and as $B\,a$ is separated, they coincide also on all E.

—*Sheaf:* Let E : Type, $\chi : E \to$ HProp dense in E and $f : \sum_{e:E} \chi\,e \to \prod_{a:A} B\,a$. Let $a : A$; the map $(\lambda x,\ f(x, a))$ is valued in the sheaf $B\,a$, so it can be extended to all E, allowing f to be extended to all E.

□

5.2 Sheafification

The sheafification process will be defined in two steps. First one will build, from any T : Type_{n+1}, a separated object $\square_{n+1} T$: Type_{n+1}; one can then show that $\square_{n+1}$ defines a modality on Type_{n+1}. Second, one builds, from any separated type T : Type_{n+1}, a sheaf $\bigcirc_{n+1}(T)$; one can show that $\bigcirc_{n+1}$ is indeed the left-exact modality we are searching.

Let n be a fixed truncation index, and $\bigcirc_n$ a left-exact modality on Type_n, compatible with $\bigcirc_{-1}$ as in

Condition 5.10. For any mere proposition P (where $\widehat{P}$ is P seen as a Type_n), $\bigcirc_n \widehat{P} = \bigcirc_{-1} P$ and the following coherence diagram commutes

$$\begin{tikzcd} P \arrow[r, "\sim"] \arrow[d, "\eta_{-1}"'] & \widehat{P} \arrow[d, "\eta_n"] \\ \bigcirc_{-1} P \arrow[r, "\sim"] & \bigcirc_n \widehat{P} \end{tikzcd}$$

5.2.1 *From types to separated types.* Let $T : \mathrm{Type}_{n+1}$. We define $\square_{n+1} T$ as the image of $\bigcirc_n^T \circ \{\cdot\}_T$, as in

$$\begin{tikzcd} T \arrow[r, "\{\cdot\}_T"] \arrow[d, "\mu_T"'] & (\mathrm{Type}_n)^T \arrow[d, "\bigcirc_n^T"] \\ \square_{n+1} T \arrow[r] & \left(\mathrm{Type}_n^{\bigcirc}\right)^T \end{tikzcd} ,$$

where $\{\cdot\}_T$ is the singleton map $\lambda(t : T),\ \lambda(t' : T),\ t = t'$. $\square_{n+1} T$ can be given explicitly by

$$\begin{aligned} \square_{n+1} T &\stackrel{def}{=} \mathrm{Im}(\lambda\ t : T,\ \lambda\ t',\ \ \bigcirc_n (t = t')) \\ &\stackrel{def}{=} \sum_{u : T \to \mathrm{Type}_n^{\bigcirc}} \left\| \sum_{a:A} (\lambda t,\ \ \bigcirc_n (a = t)) = u \right\| . \end{aligned}$$

This corresponds to the free separated object used in the topos-theoretic construction, but using $\mathrm{Type}_n^{\bigcirc}$ instead of the j-subobject classifier Ω_j.

Proposition 5.11 (separated_Type_is_separated). *For any $T : \mathrm{Type}_{n+1}$, $\square_{n+1} T$ is separated.*

Proof. We use the following lemma:

Lemma 5.12 (separated_mono_is_separated). *A $(n+1)$-truncated type T with an embedding $f : T \to U$ into a separated $(n+1)$-truncated type U is itself separated.*

Proof. Let $E : \mathrm{Type}$ and $\chi : E \to \mathrm{Type}_n$ dense in E. Let $\phi_1, \phi_2 : \sum_{e:E} \chi\, e \to T$ such that $\phi_1 \circ \pi_1 \sim \phi_2 \circ \pi_1$. Postcomposing by f yields an homotopy $f \circ \phi_1 \circ \pi_1 \sim f \circ \phi_2 \circ \pi_1$. As $f \circ \phi_1, f \circ \phi_2 : \sum_{e:E} \chi\, e \to U$, and U is separated, we can deduce $f \circ \phi_1 \sim f \circ \phi_2$. As f is an embedding, $\phi_1 \sim \phi_2$. □

As $\square_{n+1} T$ embeds in $\left(\mathrm{Type}_n^{\bigcirc}\right)^T$, we only have to show that the latter is separated. But this is the case because $\mathrm{Type}_n^{\bigcirc}$ is a sheaf (by Proposition 5.7) and a function type is a sheaf as soon as its codomain is a sheaf (by Proposition 5.9).

We will now show that $\square_{n+1}$ defines a modality, with unit map μ. The left-exactness of $\bigcirc_{n+1}$ will come from the second part of the process. The first thing to show is that $\square_{n+1} T$ is universal among separated types below T. In the topos-theoretic sheafification, it comes easily from the fact that epimorphims are coequalizers of their kernel pairs. As it is not true anymore in our setting, we will use its generalization, the proposition 2.7. Here is a sketch of the proof: as μ_T is a surjection (it is defined by the surjection-embedding factorization), $\square_{n+1} T$ is the colimit of its iterated kernel pair. Hence, for any type Q defining a cocone on $\mathrm{KP}(\mu_T)$, there is a unique arrow $\square_{n+1} T \to Q$. What remains to show is that any separated type Q defines a cocone on $\mathrm{KP}(\mu_T)$; we will actually show that any separated type Q defines a cocone on $\| \mathrm{KP}(\mu_T) \|_{n+1}$, which is enough. We do it by defining another diagram $\mathring{T}$, equivalent to $\| \mathrm{KP}(\mu_T) \|_{n+1}$, for which it is easy to define a cocone into any separated type Q.

This comes from the following construction which connects $\square_{n+1} T$ to the colimit of the iterated kernel pair of μ_T.

Definition 5.13 (`OTid`). Let X : Type. Let $\mathring{T}_X$ be the higher inductive type generated by

—$\mathring{t}: \ \|X\|_{n+1} \to \mathring{T}_X$
—$\mathring{\alpha}: \ \forall a\, b : \|X\|_{n+1}, \ \bigcirc(a = b) \to \mathring{t}(a) = \mathring{t}(b)$
—$\mathring{\alpha}_1: \ \forall a : \|X\|_{n+1}, \ \mathring{\alpha}(a, a, \eta_{a=a}1) = 1$

We view $\mathring{T}$ as the coequalizer of

$$\sum_{a,b:\|X\|_{n+1}} \bigcirc(a = b) \overset{\pi_1}{\underset{\pi_2}{\rightrightarrows}} \|X\|_{n+1}$$

preserving $\eta_{a=a}1$.

We consider the diagram $\mathring{T}$:

$$\|X\|_{n+1} \longrightarrow \|\mathring{T}_X\|_{n+1} \longrightarrow \|\mathring{T}_{\mathring{T}_X}\|_{n+1} \longrightarrow \cdots$$

The main result we want about $\mathring{T}$ is the following:

Lemma 5.14 (`separation_colimit_OTtelescope`). *Let* T : Type_{n+1}. *Then* $\Box_{n+1} T$ *is the* $(n+1)$*-colimit of the diagram* $\mathring{T}$*.*

The key point of the proof is that diagrams $\mathring{T}$ and $\|\operatorname{KP}(\mu_T)\|_{n+1}$ are equivalent. We will need the following lemma:

Lemma 5.15 (`OT_Omono_sep`). *Let* $A, S : \mathrm{Type}_{n+1}$*,* S *separated, and* $f : A \to S$*. Then if*

$$\forall a, b : A, \ f(a) = f(b) \simeq \bigcirc(a = b), \tag{1}$$

then

$$\forall a, b : \|\operatorname{KP}_f\|_{n+1}, \ |\tilde{f}|_{n+1}(a) = |\tilde{f}|_{n+1}(b) \simeq \bigcirc(a = b).$$

Proof (Sketch). By induction on truncation, we need to show that

$$\forall a, b : \operatorname{KP}_f, \ \tilde{f}(|a|_{n+1}) = \tilde{f}(|b|_{n+1}) \simeq \bigcirc(|a|_{n+1} = |b|_{n+1}).$$

We use the encode-decode [UFP13, Section 8.9] method to characterize $\tilde{f}(|a|_{n+1}) = x$, and the result follows. We refer to the formalization for details. □

This lemma allows one to prove that, in the iterated kernel pair diagram of f

$$\begin{tikzcd}
X \arrow[r] \arrow[drrr, "f"'] & \operatorname{KP}(f) \arrow[r] \arrow[drr, "f_1"] & \operatorname{KP}(f_1) \arrow[r] \arrow[d, "f_2"'] & \operatorname{KP}(f_2) \arrow[r] \arrow[dl, "f_3"] & \cdots \\
 & & S & &
\end{tikzcd}$$

if f satisfies (1), then each $|f_i|_{n+1}$ does.

Remark 5.16. It is clear that if A and B are equivalent types, and for all $a, b : A, \ f(a) = f(b) \simeq \bigcirc(a = b)$, then

$$\operatorname{Coeq}_1\left(\sum_{a,b:A} fa = fb \overset{\pi_1}{\underset{\pi_2}{\rightrightarrows}} A\right) \simeq \operatorname{Coeq}_1\left(\sum_{a,b:B} \bigcirc(a = b) \overset{\pi_1}{\underset{\pi_2}{\rightrightarrows}} B\right)$$

PROOF OF LEMMA 5.14. As said, it suffices to show that $\|C(\mu_T)\|_{n+1} = \mathring{T}$.

$$\begin{tikzcd}
\| \mathrm{KP}^0(\mu_T)\|_{n+1} \arrow[r] \arrow[d, "\wr"] & \| \mathrm{KP}^1(\mu_T)\|_{n+1} \arrow[r] \arrow[d, "\wr"] & \| \mathrm{KP}^2(\mu_T)\|_{n+1} \arrow[r] \arrow[d, "\wr"] & \cdots \\
\mathring{T}_0 \arrow[r] & \mathring{T}_1 \arrow[r] & \mathring{T}_2 \arrow[r] & \cdots
\end{tikzcd}$$

The first equivalence is trivial, so let us begin with the second. What we need to show is

$$\| \mathrm{KP}(\mu_T)\|_{n+1} \simeq \|\mathring{T}_T\|_{n+1},$$

i.e.

$$\mathrm{Coeq}_1\left(\sum_{a,b:T} \mu_T a = \mu_T b \overset{\pi_1}{\underset{\pi_2}{\rightrightarrows}} T\right) \simeq \mathrm{Coeq}_1\left(\sum_{a,b:T} \bigcirc(a = b) \overset{\pi_1}{\underset{\pi_2}{\rightrightarrows}} T\right).$$

By the previous remark, it suffices to show that μ_T satisfies condition (1), *i.e.* $\prod_{a,b:T} \bigcirc_n(a = b) = (\mu_T a = \mu_T b)$. By univalence, we want arrows in both directions, forming an equivalence.

—Suppose $p : (\mu_T a = \mu_T b)$. Then projecting p along first components yields $q : \prod_{t:T} \bigcirc_n(a = t) = \bigcirc_n(b = t)$. Taking for example $t = b$, we deduce $\bigcirc_n(a = b) = \bigcirc_n(b = b)$, and the latter is inhabited by $\eta_{b=b}1$.

—Suppose now $p : \bigcirc_n(a = b)$. Let ι be the first projection from $\square_{n+1} T \to (T \to \mathrm{Type}_n^{\bigcirc})$. ι is an embedding, thus it suffices to prove $\iota(\mu_T a) = \iota(\mu_T b)$, *i.e.* $\prod_{t:T} \bigcirc_n(a = t) = \bigcirc_n(b = t)$. The latter remains true by univalence.

As the fact that these two form an equivalence is rather technical, we refer to the formalization for an explicit proof.

Let us now show the other equivalences by induction. Suppose that, for a given $i : \mathbb{N}$, $\| \mathrm{KP}^i(\mu_T)\|_{n+1} \simeq \mathring{T}_i$. We want to prove $\| \mathrm{KP}^{i+1}(\mu_T)\|_{n+1} \simeq \mathring{T}_{i+1}$, *i.e.*

$$\begin{aligned}
&\left\|\mathrm{Coeq}_1\left(\sum_{a,b:\mathrm{KP}^i(\mu_T)} f_i a = f_i b \overset{\pi_1}{\underset{\pi_2}{\rightrightarrows}} \mathrm{KP}^i(\mu_T)\right)\right\|_{n+1} \\
\simeq &\left\|\mathrm{Coeq}_1\left(\sum_{a,b:\|\mathring{T}_i\|_{n+1}} \bigcirc(a = b) \overset{\pi_1}{\underset{\pi_2}{\rightrightarrows}} \|\mathring{T}_i\|_{n+1}\right)\right\|_{n+1}
\end{aligned}$$

where f_i is the map $\mathrm{KP}^i(\mu_T) \to \square_{n+1} T$. But lemma 5.15 just asserted that f_i satisfies (1), hence the previous yields the result.

Next one would need to show that, modulo these equivalences, the arrows of the two diagrams are equal. We leave that to the reader, who can refer to the formalization if needed. $\square$

Now, let Q be any separated Type_{n+1}, and $f : X \to Q$. Then the following diagram commutes

$$\begin{tikzcd} \|X\|_{n+1} \arrow[r] \arrow[dr] & \|\mathring{T}_X\|_{n+1} \arrow[r] \arrow[d] & \|\mathring{T}_{\mathring{T}_X}\|_{n+1} \arrow[r] \arrow[dl] & \cdots \\ & Q & & \end{tikzcd}$$

But we know (lemma 5.14) that $\Box_{n+1} T$ is the $(n+1)$-colimit of the diagram $\mathring{T}$, thus there is an universal arrow $\Box_{n+1} T \to Q$. This is enough to state the following proposition.

Proposition 5.17 (separation_reflective_subuniverse). *$(\Box_{n+1}, \mu)$ defines a reflective subuniverse on* Type_{n+1}.

To show that $\Box_{n+1}$ is a modality, it remains to show that separation is a property stable under sigma-types. Let $A : \mathrm{Type}_{n+1}$ be a separated type and $B : A \to \mathrm{Type}_{n+1}$ be a family of separated types. We want to show that $\sum_{x:A} B\,x$ is separated. Let E be a type, and $\chi : E \to \mathrm{Type}_n$ a dense subobject of E.

Let f, g be two maps from $\sum_{e:E} \chi\, e$ to $\sum_{x:A} B\,x$, equal when precomposed with π_1.

$$\begin{tikzcd} \sum_{e:E} \chi\, e \arrow[r, shift left, "f\circ\pi_1"] \arrow[r, shift right, "g\circ\pi_1"'] \arrow[d, "\text{dense}"'] & \sum_{x:A} B\,x \\ E \arrow[ur, shift left, "f"] \arrow[ur, shift right, "g"'] & \end{tikzcd}$$

We can restrict the previous diagram to

$$\begin{tikzcd} \sum_{e:E} \chi\, e \arrow[r, shift left, "\pi_1\circ f\circ\pi_1"] \arrow[r, shift right, "\pi_1\circ g\circ\pi_1"'] \arrow[d, "\text{dense}"'] & A \\ E \arrow[ur, shift left, "\pi_1\circ f"] \arrow[ur, shift right, "\pi_1\circ g"'] & \end{tikzcd}$$

and as A is separated, $\pi_1 \circ f = \pi_1 \circ g$. For the second components, let $x : E$. Notice that $\sum_{y:E} x = y$ has a dense n-subobject, $\sum_{y:\sum_{e:E} \chi\, e} x = y_1$:

$$\begin{tikzcd} \sum_{y:\sum_{e:E} \chi\, e} x = y_1 \arrow[r, shift left, "\pi_2\circ f\circ\pi_1\circ\pi_1"] \arrow[r, shift right, "\pi_2\circ g\circ\pi_1\circ\pi_1"'] \arrow[d, "\text{dense}"'] & B\,x \\ \sum_{y:E} x = y \arrow[ur, shift left, "\pi_2\circ f\circ\pi_1"] \arrow[ur, shift right, "\pi_2\circ g\circ\pi_1"'] & \end{tikzcd}$$

Using the separation property of $B\,x$, one can show that second components, correctly transported along the first component's equality, are themselves equal. The complete proof can be found in the formalization. This proves the following proposition

Proposition 5.18 (`separated_modality`). *$(\Box_{n+1}, \mu)$ defines a truncated modality on* Type_{n+1}.

As this modality forms just one step in our construction, we do not need to show that it is left exact. This we will do only for the sheafification modality.

5.2.2 *From Separated Type to Sheaf.* For any T in Type_{n+1}, $\bigcirc_{n+1}T$ is defined as the closure of $\Box_{n+1}T$, seen as a subobject of $T \to \mathrm{Type}_n^{\bigcirc}$. $\bigcirc_{n+1}T$ can be given explicitly by

$$\bigcirc_{n+1}T \overset{def}{=} \sum_{u:T\to\mathrm{Type}_n^{\bigcirc}} \bigcirc_{-1} \left\| \sum_{a:T} (\lambda t,\ \ \bigcirc_n (a = t)) = u \right\|.$$

To prove that $\bigcirc_{n+1}T$ is a sheaf for any $T : \mathrm{Type}_{n+1}$, we use the following lemma.

Lemma 5.19 (`closed_to_sheaf`). *Any closed (-1)-subobject of a sheaf is a sheaf.*

Proof. Let U be a sheaf, and $\kappa : U \to \mathrm{HProp}$ be a closed (-1)-subobject. Let $E : \mathrm{Type}$ and $\chi : E \to \mathrm{HProp}$ dense in E. Let $\phi : \sum_{e:E} \chi\, e \to \sum_{u:U} \kappa\, u$. As $\pi_1 \circ \phi$ is a map $\sum_{e:E} \chi\, e \to U$ and U is a sheaf, it can be extended into $\psi : E \to U$. As κ is closed, it suffices now to prove $\prod_{e:E} \bigcirc_n(\kappa\,(\psi\, e))$ to obtain a map $E \to \sum_{u:U} \kappa\, u$.

Let $e : E$. As χ is dense, we have a term $w : \bigcirc_n(\chi\, e)$, and by $\bigcirc_n$-induction, a term $\widetilde{w} : \chi\, e$. Then, by the retraction property, $\psi(e) = \phi(e, \widetilde{w})$, and by $\pi_2 \circ \phi$, we have hence our term of type $\kappa(\psi\, e)$. □

As $T \to \mathrm{Type}_n^{\bigcirc}$ is a sheaf, and $\bigcirc_{n+1}T$ is closed in $T \to \mathrm{Type}_n^{\bigcirc}$, $\bigcirc_{n+1}T$ is a sheaf. We now prove that it forms a reflective subuniverse.

Proposition 5.20 (`sheafification_subu`). *$(\bigcirc_{n+1}, \nu)$ defines a reflective subuniverse.*

Proof. Let $T, Q : \mathrm{Type}_{n+1}$ such that Q is a sheaf. Let $f : T \to Q$. Because Q is a sheaf, it is in particular separated; thus we can extend f to $\Box_{n+1} f : \Box_{n+1} T \to Q$.

But as $\bigcirc_{n+1}T$ is the closure of $\Box_{n+1} T$, $\Box_{n+1} T$ is dense into $\bigcirc_{n+1}T$, so the sheaf property of Q allows to extend $\Box_{n+1} f$ to $\bigcirc_{n+1}f : \bigcirc_{n+1}T \to Q$.

As all these steps are universal, so is the composition. □

The next step is the closure under dependent sums, specifically:

Proposition 5.21 (`sheafification_modality`). *$(\bigcirc_{n+1}, \nu)$ defines a modality.*

Proof. The proof uses the same ideas as in subsection 5.2.1. Let $A : \mathrm{Type}_{n+1}$ a sheaf and $B : A \to \mathrm{Type}_{n+1}$ a sheaf family. By proposition 5.18, we already know that $\sum_{a:A} B\, a$ is separated. Let E be a type, and $\chi : E \to \mathrm{HProp}$ a dense subobject. Let $f : \sum_{e:E} \chi\, e \to \sum_{x:A} B\, x$; we want to extend it into a map $E \to \sum_{x:A} B\, x$.

$$\begin{tikzcd}
\sum_{e:E} \chi\, e \arrow[r, "f"] \arrow[d] & \sum_{x:A} B\, x \\
E \arrow[ur, dotted] &
\end{tikzcd}$$

As A is a sheaf, and $\pi_1 \circ f : \sum_{e:E} \chi\, e \to A$, we can recover a map $g_1 : E \to A$. We then want to show $\prod_{e:E} B(g_1\, e)$. Let $e : E$. As χ is dense, we have a term $w : \bigcirc_n(\chi\, e)$, and as $B(g_1\, e)$ is a sheaf, we can recover a term $\widetilde{w} : \chi\, e$. Then $g_1(e) = f(e, \widetilde{w})$, and $\pi_2 \circ f$ gives the result. □

It remains to show that $\bigcirc_{n+1}$ is left exact and is compatible with $\bigcirc_{-1}$. To do that, we need to extend the notion of compatibility and show that, in fact, every modality $\bigcirc_{n+1}$ is compatible with $\bigcirc_n$ on lower homotopy types.

Proposition 5.22. *If* $T : \mathrm{Type}_n$*, then* $\bigcirc_{n+1}\widehat{T} = \bigcirc_n T$*, where* $\widehat{T}$ *is* T *seen as a* Type_{n+1}.

Proof. We prove it by induction on n:

—For $n = -1$: Let $T : \mathrm{HProp}$. Then

$$\begin{aligned}\bigcirc_0 \widehat{T} &\stackrel{def}{=} \sum_{u:T\to\mathrm{Type}_n^{\bigcirc}} \bigcirc_{-1} \left\| \sum_{a:T} (\lambda t,\ \ \bigcirc_{-1}(a=t)) = u \right\|_{-1} \\ &= \sum_{u:T\to\mathrm{Type}_n^{\bigcirc}} \bigcirc_{-1} \left(\sum_{a:T} (\lambda t,\ \ \bigcirc_{-1}(a=t)) = u \right)\end{aligned}$$

because the type inside the truncation is already in HProp. Now, let define $\phi : \bigcirc_{-1} T \to \bigcirc_0 T$ by

$$\phi t = (\lambda t',\, \mathbf{1}; \kappa)$$

where κ is defined by $\bigcirc_{-1}$-induction on t. Indeed, as T is an HProp, $(a = t) \simeq \mathbf{1}$. Let $\psi : \bigcirc_0 T \to \bigcirc_{-1} T$ by obtaining the witness $a : T$ (which is possible because we are trying to inhabit a modal proposition), and letting $\psi(u; x) = \eta_T a$. These two maps form an equivalence (the section and retraction are trivial because the equivalence is between mere propositions).

—Suppose now that $\bigcirc_{n+1}$ is compatible with all $\bigcirc_k$ on lower homotopy types. Let $\bigcirc_{n+2}$ be as above, and let $T : \mathrm{Type}_{n+1}$. Then, as $\bigcirc_{n+1}$ is compatible with $\bigcirc_n$, and $(a = t)$ is in Type_n,

$$\bigcirc_{n+2}\widehat{T} = \sum_{u:T\to\mathrm{Type}_{n+1}^{\bigcirc}} \bigcirc_{-1} \left\| \sum_{a:T} (\lambda t,\ \ \bigcirc_n (a=t)) = u \right\|_{-1}.$$

It remains to prove that for every (u, x) inhabiting the Σ-type above, u is in $T \to \mathrm{Type}_n^{\bigcirc}$, *i.e.* that for every $t : T$, Is-n-type$(u\, t)$. But for any truncation index p, the type Is-p-type X : HProp is a sheaf as soon as X is, so we can get rid of $\bigcirc_{-1}$ and of the truncation, which tells us that for every $t : T$, $u\, t = \bigcirc_n(a = t) : \mathrm{Type}_n$.

□

This proves in particular that $\bigcirc_{n+1}$ is compatible with $\bigcirc_{-1}$ in the sense of condition 5.10.

The last step is the left-exactness of $\bigcirc_{n+1}$. Let T be in Type_{n+1} such that $\bigcirc_{n+1} T$ is contractible. Thanks to the just shown compatibility between $\bigcirc_{n+1}$ and $\bigcirc_n$ for Type_n, left-exactness means that for any $x, y : T$, $\bigcirc_n(x = y)$ is contractible.

Using a proof by univalence as we have done for proving $\bigcirc_n(a = b) \simeq (\mu_T(a) = \mu_T(b))$ in Proposition 5.14, we can show that:

PROPOSITION 5.23 (`good_sheafification_unit_paths_are_nj_paths`). *For all $a, b : T$, $\bigcirc_n(a = b) \simeq (\nu_T a = \nu_T b)$.*

As $\bigcirc_{n+1}T$ is contractible, path spaces of $\bigcirc_{n+1}T$ are contractible, in particular $(\nu_T a = \nu_T b)$, which proves left exactness.

5.3 Summary

Starting from any left-exact modality $\bigcirc_{-1}$ on HProp, we have defined for any truncation level n, a new left-exact modality $\bigcirc_n$ on Type_n, which corresponds to $\bigcirc_{-1}$ when restricted to HProp.

When $\bigcirc_{-1}$ is consistent (in the sense of section 3.2), $\bigcirc_n \mathbf{0} = \bigcirc_{-1}\mathbf{0}$ is also not inhabited, hence $\bigcirc_n$ is consistent. In particular, the modality induced by the double negation modality on HProp is consistent.

In topos theory, the topos of Lawvere-Tierney sheaves for the double negation topology is a boolean topos. In homotopy type theory, this result can be expressed as:

PROPOSITION 5.24. *Taking $(\bigcirc_{\neg\neg})_n$, the modality obtained by sheafification of the double negation modality, the following holds*

$$\prod_{P:\mathrm{HProp}} \bigcirc_{\neg\neg}(P + \neg P).$$

PROOF. Let $P : \mathrm{HProp}$, and pose $Q \overset{def}{=} P + \neg P$. Then, as P and $\neg P$ are disjoint h-propositions, $P + \neg P$ is itself a h-proposition [BGL+17, `ishprop_sum`]. Thus, $\bigcirc_{\neg\neg} Q \simeq \neg\neg Q$, and the latter is inhabited by the usual

$$\lambda\,(x : \neg Q),\ x(\mathrm{inr}(\lambda\, y : P,\ x(\mathrm{inl}\, y))).$$

□

5.4 Extension to Type

In the previous section, we have defined a (countably) infinite family of modalities $\mathrm{Type}_i \to \mathrm{Type}_i$. One can extend them to whole Type by composing with truncation:

LEMMA 5.25. *Let $\bigcirc_i : \mathrm{Type}_i \to \mathrm{Type}_i$ be a modality. Then $\bigcirc \overset{def}{=} \bigcirc_i \circ \|\cdot\|_i : \mathrm{Type} \to \mathrm{Type}$ is a modality in the sense of section 3*

PROOF. It is straightforward to check each property of a modality. □

If $\bigcirc_{-1}$ is the double negation modality on HProp and $i = -1$, $\bigcirc$ is exactly the double negation modality on Type described in 3.1.3. Choosing $i \geqslant 0$ is a refinement of this double negation modality on Type: it will collapse every type to a Type_i, instead of an HProp.

Obviously, as truncation modalities are not left-exact [UFP13, Exercise 7.11], $\bigcirc$ isn't either. But in the following sense, when restricted to i-truncated types, it is:

LEMMA 5.26. *Let $A : \mathrm{Type}_i$. Then if $\bigcirc(A)$ is contractible, for any $x, y : A$, $\bigcirc(x = y)$ is contractible.*

PROOF. For i-truncated types, $\bigcirc = \bigcirc_i$, and $\bigcirc_i$ is left-exact. $\square$

The compatibility between the modalities $\bigcirc_n$ and between the modalities $\|\cdot\|_n$ allows us to take the truncation index as high as desired. Taking it as a non-fixed parameter allows one to work in a universe where the new principle (*e.g.* mere excluded middle) is true for any explicit truncated type. Indeed, i can be chosen dynamically along a proof, and thus be increased as much as needed, without changing results for lower truncated types.

Furthermore, univalence remains true in this new type theory in the following sense:

PROPOSITION 5.27. *Let n be a given truncation index, and $\bigcirc$ the modality associated to n as defined in lemma 5.25. Then, for any type $A, B : \mathrm{Type}_n^{\bigcirc}$, if φ is the canonical arrow*

$$A = B \to A \simeq B,$$

then $\mathrm{IsEquiv}(\varphi)$ *is modal.*

PROOF. The first thing to notice is that, if X and Y are modal, and $f : X \to Y$, then the mere proposition $\mathrm{IsEquiv}\, f$ is also modal. Therefore, it suffices to show that both $A = B$ and $A \simeq B$ are modal. By proposition 3.4, $A = B$ is modal. Moreover, $(A \simeq B) \simeq \sum_{f:A\to B} \mathrm{IsEquiv}\, f$. Therefore, as A and B are modal, $A \simeq B$ is too.

Hence, $\mathrm{IsEquiv}\, \varphi$ is modal. $\square$

6. FORMALIZATION

A Coq formalization of the sheafification process based on the Coq/HoTT library [BGL+17] is available at `https://github.com/KevinQuirin/sheafification`.

After reviewing the content and some statistics about the formalization in Section 6.1, we present the limitations of our formalization in Section 6.2, in particular the issues relative to universe polymorphism.

6.1 Content of the formalization

We provide a more detailed insight of the structure of our formalization:

—Colimits and iterated kernel pairs are formalized in `Limit`, `T.v`, `OT.vv` `OT_Tf.v`, `T_telescope.v`, `Tf_Omono_sep.v`.

—Reflective subuniverses and modalities are formalized in `reflective_subuniverse.v`, `modalities.v`.

—The definition of the dense topology as a left exact modality on HProp is given in `sheaf_base_case.v`.

—Section 5.1 is formalized in `sheaf_def_and_thm.v`.

—Section 5.2 is formalized in `sheaf_induction.v`.

Overall, the project contains 8000 lines. This, however, could be reduced a bit by improving the way Coq tries to rewrite and apply lemmas automatically. The `coqwc` tool counts 1600 lines of specifications (definitions, lemmas, theorems, propositions) and 5500 lines of proof script. This constitutes a significant amount of work but the part dedicated to sheaves and sheafification is only 2200 lines of proof script, which seems quite reasonable and encouraging. Moreover it suggests that homotopy type theory provides a convenient tool in which to formalize some parts of the theory of higher toposes.

6.2 Limitations of the formalization

In the formalization, we we forced to use the `type-in-type` option to handle the universe issues we faced. However, a lot of the code compiles without this flag (but still needs universe polymorphism).

Universes are used in type theory to ensure consistency by checking that definitions are well-stratified according to a certain hierarchy. Universe polymorphism [ST14] supports generic definitions over universes, reusable at different levels. Although the presence of universe polymorphism is mandatory for our formalization, its implementation is still too rigid to allow a complete formalization of our work for the following reasons.

While Coq handles cumulativity on Type natively, this is not the case for the Σ-type Type_n, which requires propositional resizing. This issue could be solved by adding an axiom of cumulativity for Type_n with an explicit management of universes. However, as it would not have any computational content, such a solution would needlessly complicate the proofs. Indeed, the axiom would then appear everywhere cumulativity is needed and one would need explicit annotations for universe levels throughout the formalization.

One issue with universe polymorphism lies in the management of recursive definitions. Indeed, the following recursive definition of sheafification

$$
\begin{aligned}
&\bigcirc : \forall\ (n : nat),\ \mathrm{Type}_n \to \mathrm{Type}_n \\
&\bigcirc_{-1}\ (T) \overset{def}{=} \neg\neg T \\
&\bigcirc_{n+1}(T) \overset{def}{=} \sum_{u:T\to\mathrm{Type}_n^{\bigcirc}} \bigcirc_{-1} \left\| \sum_{a:T} u = (\lambda t,\ \ \bigcirc_n\ (a = t)) \right\|
\end{aligned}
$$

is not allowed. This is because Coq forces the universe of the first Type_n occurring in the definition to be the same for every n, whereas the universe of the first Type_{n+1} occurring in $\bigcirc_{n+1}$ should be at least one level higher as the one of Type_n occurring in $\bigcirc_n$ because of the use of Σ-type over $T \to \mathrm{Type}_n^{\bigcirc}$ and equality on the return type of $\bigcirc_n$. Thus, the induction step presented in this paper has been formalized, but the complete recursive sheafification can not be defined for the moment. Note that the same increase in the universe levels occurs in the Rezk completion of categories [AKS15]. In the definition of the completion, one uses the Yoneda embedding and representable functors, which is similar in spirit to our use of characteristic functions.

This restriction in our formalization may be solved by generalizing the management of universe polymorphism for recursive definitions or by the use of more general "resizing axioms" which are still under discussion in the community.

7. CONCLUSION AND FUTURE WORKS

In this paper, we have demonstrated a way to extend Lawvere-Tierney sheafification to truncated types in homotopy type theory.

Above and beyond this individual result, our work is part of a larger program which aims to illustrate that homotopy type theory is a promising candidate for the formalization of mathematics inside a proof assistant.

In future work, we would like to improve this construction in three ways. (i) The extension to whole Type in lemma 5.25 is not totally satisfactory, as every type is collapsed to a truncated one. But some types in homotopy type theory are not truncated [UFP13, Example 8.8.6]. (ii) We would like to have more examples of left-exact modalities on HProp, in order to have sheaves for different properties than excluded middle. (iii) In topos theory, Lawvere-Tierney subsumes Grothendieck [MM92, Section V.4] in the sense that any Grothendieck topology gives rise to a Lawvere-Tierney topology with the same notion of sheaves. Higher Lawvere-Tierney sheaves are presented here, and higher Grothendieck sheaves have been defined in [Lur09]. It should be interesting to check if the subsumption remains true in higher topos theory.

Moreover, we highly suspect that modalities (at least, left-exact accessible modalities) induces new type theories, as Grothendieck sheaves exhibits some $(\infty, 1)$-subtoposes. It would be nice to give a better sense to sub-type theories and to instantiate them with sheafification, giving, for example, a model of homotopy type theory with computational mere excluded middle.

ACKNOWLEDGMENTS

We would like to thank all maintainers of the Coq/HoTT library, who have provided a very solid base for our formalization. We also would like to thank Bas Spitters for helpful discussions on the subject. This work has been funded by the CoqHoTT ERC Grant 637339.

References

[AKL15] Jeremy Avigad, Krzysztof Kapulkin, and Peter LeFanu Lumsdaine. Homotopy limits in type theory. *Mathematical Structures in Computer Science*, January 2015.

[AKS15] Benedikt Ahrens, Krzysztof Kapulkin, and Michael Shulman. Univalent Categories and the Rezk completion. *Mathematical Structures in Computer Science*, January 2015.

[BBC98] Stefano Berardi, Marc Bezem, and Thierry Coquand. On the computational content of the axiom of choice. *Journal of Symbolic Logic*, pages 600–622, 1998.

[BGL+17] Andrej Bauer, Jason Gross, Peter LeFanu Lumsdaine, Mike Shulman, Matthieu Sozeau, and Bas Spitters. The HoTT Library: A formalization of homotopy type theory in Coq. In *Certified Programs and Proofs (CPP), 6th ACM SIGPLAN Conference on*, 2017.

[Bou16] Simon Boulier. Colimits in HoTT. Blog post, `http://homotopytypetheory.org/2016/01/08/colimits-in-hott/`, 2016.

[BU02] Gilles Barthe and Tarmo Uustalu. CPS translating inductive and coinductive types. *ACM SIGPLAN Notices*, 37(3):131–142, 2002.

[CDT15] The Coq Development Team. *The Coq proof assistant reference manual*. 2015. Version 8.5.

[Coh66] Paul Cohen. *Set theory and the continuum hypothesis.* WA Benjamin New York, 1966.

[Göd38] Kurt Gödel. The consistency of the axiom of choice and of the generalized continuum-hypothesis. *Proceedings of the National Academy of Sciences*, 24(12):556–557, 1938.

[Gö40] Kurt Gödel. *The Consistency of the Axiom of Choice and of the Generalized Continuum Hypothesis with the Axioms of Set Theory.* Princeton University Press, 1940.

[Her12] Hugo Herbelin. A constructive proof of dependent choice, compatible with classical logic. In *Proceedings of the 2012 27th Annual IEEE/ACM Symposium on Logic in Computer Science*, pages 365–374. IEEE Computer Society, 2012.

[JLP+16] Guilhem Jaber, Gabriel Lewertowski, Pierre-Marie Pédrot, Matthieu Sozeau, and Nicolas Tabareau. The Definitional Side of the Forcing. In *LICS*, New York, United States, May 2016.

[JTS12] Guilhem Jaber, Nicolas Tabareau, and Matthieu Sozeau. Extending type theory with forcing. In *Logic in Computer Science (LICS), 27th Annual IEEE Symposium on*, pages 395–404. IEEE, 2012.

[KL12] Krzysztof Kapulkin and Lumsdaine Peter LeFanu. The simplicial model of univalent foundations. *arXiv preprint arXiv:1211.2851*, 2012.

[Kri03] Jean-Louis Krivine. Dependent choice,'quote'and the clock. *Theoretical Computer Science*, 308(1):259–276, 2003.

[LS13] Peter LeFanu Lumsdaine and Michael Shulman. Higher inductive types. *preparation*, 2013.

[Lur09] Jacob Lurie. *Higher topos theory.* Annals of mathematics studies. Princeton University Press, Princeton, N.J., Oxford, 2009.

[MM92] Saunders MacLane and Ieke Moerdijk. *Sheaves in Geometry and Logic.* Springer-Verlag, 1992.

[MV99] Fabien Morel and Vladimir Voevodsky. A^1-homotopy theory of schemes. *Publications Mathématiques de l'IHÉS*, 90:45–143, 1999.

[Par93] Michel Parigot. Classical proofs as programs. In *Computational logic and proof theory*, pages 263–276. Springer, 1993.

[RS15] Egbert Rijke and Bas Spitters. Sets in homotopy type theory. *Mathematical Structures in Computer Science*, 25(5):1172–1202, 2015.

[Shu] Mike Shulman. Module for Modalities. Blog post, `http://homotopytypetheory.org/2015/07/05/modules-for-modalities/`.

[Shu15a] Michael Shulman. Brouwer's fixed-point theorem in real-cohesive homotopy type theory. *arXiv preprint arXiv:1509.07584*, 2015.

[Shu15b] Michael Shulman. Univalence for inverse diagrams and homotopy canonicity. *arXiv preprint arXiv:1203.3253*, 2015.

[SS14] Urs Schreiber and Michael Shulman. Quantum gauge field theory in cohesive homotopy type theory. In Ross Duncan and Prakash Panangaden, editors, Proceedings 9th Workshop on *Quantum Physics and*

Logic, Brussels, Belgium, 10-12 October 2012, volume 158 of *Electronic Proceedings in Theoretical Computer Science*, pages 109–126, 2014.

[ST14] Matthieu Sozeau and Nicolas Tabareau. Universe polymorphism in Coq. In *Interactive Theorem Proving*, 2014.

[Tie72a] Myles Tierney. *Sheaf theory and the continuum hypothesis*. Springer, 1972.

[Tie72b] Myles Tierney. Sheaf theory and the continuum hypothesis. *LNM 274*, pages 13–42, 1972.

[UFP13] The Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations of Mathematics*. `http://homotopytypetheory.org/book`, IAS, 2013.

[Voe] Vladimir Voevodsky. Resising Rules - their use and semantic justification. `www.math.ias.edu/~vladimir/Site3/Univalent_Foundations_files/2011_Bergen.pdf`.