Internship report M2
Non-singleton Elimination

Soudant Léo, supervised by **Pierre-Marie Pédrot**, Galinette team at LS2N

2025

General context

To efficiently formalise mathematics we need strong permissive theories. When attempting to mechanising proofs, however, strong computational properties, like typing decidability, are also expected. Studying the computational contents of sheaves could allow to emulate forcing directly in type theory, but also to bring a computational content to the axiom which the theory forces. Thierry coquand published a few papers on the computational aspect of forcing, Vincent Rahli studied effectful, though undecidable, type theory based on sheaves. Martin Baillon used a similar theory to show a continuity result on the cantor space [1].

Research problem

An element of a topological sheaf may be defined over an open whenever it is defined on a cover of that open, so long as they are compatible over the intersections. We hope to be able to do something similar in a type theory, where open set correspond to proof irrelevant propositions. This requires special attention to the compatibility condition : when a disjunction covers a proposition, it could be possible to eliminate from this disjunction to create a term in any type whenever compatible marginal terms are found.

Your contribution

We have proved using Rocq that a system T with a part of the wanted features normalises. I also have a model in Rocq where types are interpreted as sheaves, though some other variant probably existed before.

Arguments supporting its validity

The type theory devellopped by Martin Baillon, which we aim to generalise, proved continuity of all functional $(\mathbf{N} \to \mathbf{B}) \to \mathbf{B}$. We hope to find similar results at term.

Summary and future works

We will refine the type theory further as well as study other instances similar to the one Martin Baillon studied. We should also make a fork of the logrel Rocq project, and adapt it to show the good properties of any such theory.

Contents

Outline

In section 1 we explain what type theory is and detail the basic type theory we will study. Section 2 present the construction we would like to make appear in type theory, and why we think it is possible and useful. Section 3 present work done on models of type theory, though the important part is precisely the model of sheaf, since its theory is what we ar etrying to build syntactically. Section 4 exposes two sheaf theories, one which comes from previous work and is an important inspiration, and the other which is a prototype of a generalisation. The last section, section 5, uses the recipe of logical relation to make a proof of concept using a weaker theory, which normalises.

1 Type theory

Type theory is a manner of formalising mathematics with a very different design from set theory, and quite a bit closer in structure to a purely logical system like natural deduction. They are also more adapted to mechanising proof than ZFC.

Our meta-theory should be a type theory, with $\mathfrak{N}$ denoting its natural numbers, $\mathfrak{B}$ its boolean type with constants $\mathfrak{tt}$ and $\mathfrak{ff}$ and Type its type of types.

1.1 MLTT

We present a variant of MLTT (Martin-Löf Type Theory), a very standard type theory

It is also the theory that we will seek to extend.

We give ourself a poset of universe levels ℓ, with strict upper bounds to any finite subsets.

We add a type $\mathbf{N}$ of natural numbers and an empty type $\bot$ as positive types, in stead of the heavy syntax of a general inductive type. They both implement large elimination, which means, in particular, that their induction principle can be used to construct function outputting types, like $\lambda An, A^n$, in presence of a binary product.

With terms :

$$M, N ::= x|\lambda x.M|MN|0|S|\mathbf{N}_{\text{rec}}|\bot_{\text{rec}}|\mathbf{N}|\bot|\Pi x : A.B|\square_\ell$$

Contexts :

$$\Gamma ::= \Gamma, x : A|\cdot$$

And conversion rules :

$$\textsc{Wf-Empty}\frac{}{\cdot \vdash \mathsf{well\text{-}formed}} \qquad \textsc{Wf-Ext}\frac{\Gamma \vdash A \equiv A \qquad \Gamma \vdash \mathsf{well\text{-}formed}}{\Gamma, x : A \vdash \mathsf{well\text{-}formed}}$$

$$\textsc{Int-Typ}\frac{\Gamma \vdash \mathsf{well\text{-}formed}}{\Gamma \vdash \mathbf{N} \equiv \mathbf{N} : \square_\ell} \qquad \textsc{Emp-Typ}\frac{\Gamma \vdash \mathsf{well\text{-}formed}}{\Gamma \vdash \bot \equiv \bot : \square_\ell}$$

$$\textsc{Fun-Typ}\frac{\Gamma \vdash A \equiv A' : \square_\ell \qquad \Gamma, x : A \vdash B \equiv B' : \square_{\ell'} \qquad \ell \leq \ell'' \qquad \ell' \leq \ell''}{\Gamma \vdash \Pi x : A, B \equiv \Pi x : A', B' : \square_{\ell''}} \qquad \textsc{Typ-Typ}\frac{\Gamma \vdash \mathsf{well\text{-}formed} \qquad \ell < \ell'}{\Gamma \vdash \square_\ell \equiv \square_\ell : \square_{\ell'}}$$

$$\textsc{Fun-Intro}\frac{\Gamma, x : A \vdash M \equiv M' : B \qquad \Gamma \vdash A \equiv A : \square_\ell}{\Gamma \vdash \lambda x.M \equiv \lambda x.M' : \Pi x : A, B} \qquad \textsc{Fun-Elim}\frac{\Gamma \vdash M \equiv M' : \Pi x : A, B \qquad \Gamma \vdash N \equiv N' : A}{\Gamma \vdash MN \equiv M'N' : B(N/x)}$$

$$\textsc{Axiom}\frac{\Gamma \vdash \mathsf{well\text{-}formed} \qquad x : A \in \Gamma}{\Gamma \vdash x \equiv x : A} \qquad \textsc{Beta}\frac{\Gamma, x : A \vdash M \equiv M' : B \qquad \Gamma \vdash N \equiv N' : A}{\Gamma \vdash (\lambda x.M)N \equiv M'(N'/x) : B(N/x)}$$

$$\textsc{Int-Zero}\frac{\Gamma \vdash \mathsf{well\text{-}formed}}{\Gamma \vdash 0 \equiv 0 : \mathbf{N}} \qquad \textsc{Int-Succ}\frac{\Gamma \vdash \mathsf{well\text{-}formed}}{\Gamma \vdash S \equiv S : \mathbf{N} \to \mathbf{N}}$$

$$\textsc{Int-Rec}\frac{\Gamma \vdash \mathsf{well\text{-}formed}}{\Gamma \vdash \mathbf{N}_{\text{rec}} \equiv \mathbf{N}_{\text{rec}} : \Pi A : \mathbf{N} \to \square_\ell, A0 \to (\Pi n : \mathbf{N}, An \to A(Sn)) \to \Pi n : \mathbf{N}, An}$$

$$\textsc{Int-Rec-Zero}\frac{\Gamma \vdash A \equiv A : \mathbf{N} \to \square_\ell \qquad \Gamma \vdash N_0 \equiv N_0' \equiv A0 \qquad \Gamma \vdash N_S \equiv N_S : \Pi n : \mathbf{N}, An \to A(Sn)}{\Gamma \vdash \mathbf{N}_{\text{rec}} A N_0 N_S 0 \equiv N_0' : A0}$$

$$\textsc{Int-Rec-Succ}\frac{\begin{array}{c}\Gamma \vdash A \equiv A' : \mathbf{N} \to \square_\ell \qquad \Gamma \vdash N_0 \equiv N_0' \equiv A0 \\ \Gamma \vdash N_S \equiv N_S' : \Pi n : \mathbf{N}, An \to A(Sn) \qquad \Gamma \vdash N \equiv N' : \mathbf{N}\end{array}}{\Gamma \vdash \mathbf{N}_{\mathrm{rec}} A N_0 N_S (SN) \equiv N_S' N' (\mathbf{N}_{\mathrm{rec}} A' N_0' N_S' N') : A(SN)}$$

$$\textsc{Emp-Rec}\frac{\Gamma \vdash \mathsf{well\text{-}formed}}{\Gamma \vdash \bot_{\mathrm{rec}} \equiv \bot_{\mathrm{rec}} : \Pi A : \bot \to \square_\ell, \Pi e : \bot, Ae}$$

$$\textsc{Sym}\frac{\Gamma \vdash M \equiv M' : A}{\Gamma \vdash M' \equiv M : A} \qquad \textsc{Trans}\frac{\Gamma \vdash M \equiv M' : A \qquad \Gamma \vdash M' \equiv M'' : A}{\Gamma \vdash M \equiv M'' : A}$$

$$\textsc{Conv}\frac{\Gamma \vdash M \equiv M' : A \qquad \Gamma \vdash A \equiv A' : \square_\ell}{\Gamma M \equiv M' : A'}$$

But it is useful to consider the extension with a type of booleans $\mathbf{B}$, with new terms : $M, N ::= \ldots | \mathbf{B} | \mathbf{B}_{\mathrm{rec}} | \mathsf{tt} | \mathsf{ff}$
And conversion rules

$$\textsc{Bool-True}\frac{\Gamma \vdash \mathsf{well\text{-}formed}}{\Gamma \vdash \mathsf{tt} \equiv \mathsf{tt} : \mathbf{B}} \qquad \textsc{Bool-False}\frac{\Gamma \vdash \mathsf{well\text{-}formed}}{\Gamma \vdash \mathsf{ff} \equiv \mathsf{ff} : \mathbf{B}}$$

$$\textsc{Bool-Rec}\frac{\Gamma \vdash \mathsf{well\text{-}formed}}{\Gamma \vdash \mathbf{B}_{\mathrm{rec}} \equiv \mathbf{B}_{\mathrm{rec}} : \Pi A : \mathbf{B} \to \square_\ell, A\,\mathsf{tt} \to A\,\mathsf{ff} \to \Pi b : \mathbf{B}, Ab}$$

$$\textsc{Bool-Rec-True}\frac{\begin{array}{c}\Gamma \vdash A \equiv A : \mathbf{B} \to \square_\ell \\ \Gamma \vdash M_{\mathsf{tt}} \equiv M_{\mathsf{tt}}' \equiv A\mathsf{tt} \qquad \Gamma \vdash M_{\mathsf{ff}} \equiv M_{\mathsf{ff}} : A\,\mathsf{ff}\end{array}}{\Gamma \vdash \mathbf{B}_{\mathrm{rec}} A M_{\mathsf{tt}} M_{\mathsf{ff}}\,\mathsf{tt} \equiv M_{\mathsf{tt}}' : A\,\mathsf{tt}}$$

$$\textsc{Bool-Rec-False}\frac{\begin{array}{c}\Gamma \vdash A \equiv A : \mathbf{B} \to \square_\ell \\ \Gamma \vdash M_{\mathsf{tt}} \equiv M_{\mathsf{tt}} \equiv A\,\mathsf{tt} \qquad \Gamma \vdash M_{\mathsf{ff}} \equiv M_{\mathsf{ff}}' : A\,\mathsf{ff}\end{array}}{\Gamma \vdash \mathbf{B}_{\mathrm{rec}} A M_{\mathsf{tt}} M_{\mathsf{ff}}\,\mathsf{ff} \equiv M_{\mathsf{ff}}' : A\,\mathsf{ff}}$$

1.2 System T

To identify and solve problems in a simpler environment, we studied a modified System T before, based on the following variant.

With types :

$$A ::= A \to A | \mathbf{N} | \bot$$

Terms :

$$M, N ::= x | \lambda x.M | MN | 0 | S | \mathbf{N}_{\mathrm{rec}} | \bot_{\mathrm{rec}}$$

Contexts :

$$\Gamma ::= \Gamma, x : A | \cdot$$

And conversion rules :

$$\textsc{Fun-Intro}\frac{\Gamma, x : A \vdash M \equiv M' : B}{\Gamma \vdash \lambda x.M \equiv \lambda x.M' : A \to B} \qquad \textsc{Fun-Elim}\frac{\Gamma \vdash M \equiv M' : A \to B \qquad \Gamma \vdash N \equiv N' : A}{\Gamma \vdash MN \equiv M'N' : B}$$

$$\textsc{Axiom}\frac{x : A \in \Gamma}{\Gamma \vdash x \equiv x : A} \qquad \textsc{Beta}\frac{\Gamma, x : A \vdash M \equiv M' : B \qquad \Gamma \vdash N \equiv N' : A}{\Gamma \vdash (\lambda x.M)N \equiv M' : B}$$

$$\textsc{Int-Zero}\frac{\Gamma \vdash \mathsf{well\text{-}formed}}{\Gamma \vdash 0 \equiv 0 : \mathbf{N}} \qquad \textsc{Int-Succ}\frac{\Gamma \vdash \mathsf{well\text{-}formed}}{\Gamma \vdash S \equiv S : \mathbf{N} \to \mathbf{N}}$$

$$\textsc{Int-Rec}\frac{\Gamma \vdash \mathsf{well\text{-}formed}}{\Gamma \vdash \mathbf{N}_{\mathrm{rec}} \equiv \mathbf{N}_{\mathrm{rec}} : A \to (\mathbf{N} \to A \to A) \to \mathbf{N} \to A}$$

$$\textsc{Int-Rec-Zero}\frac{\begin{array}{c}\Gamma \vdash N_0 \equiv N_0' \equiv A \\ \Gamma \vdash N_S \equiv N_S : \mathbf{N} \to A \to A\end{array}}{\Gamma \vdash \mathbf{N}_{\mathrm{rec}} N_0 N_S 0 \equiv N_0' : A}$$

$$\textsc{Int-Rec-Succ}\frac{\begin{array}{c}\Gamma \vdash N_0 \equiv N_0' \equiv A \\ \Gamma \vdash N_S \equiv N_S' : \mathbf{N} \to A \to A \quad \Gamma \vdash N \equiv N' : \mathbf{N}\end{array}}{\Gamma \vdash \mathbf{N}_{\text{rec}} N_0 N_S (SN) \equiv N_S' N' (\mathbf{N}_{\text{rec}} N_0' N_S' N') : A}$$

$$\textsc{Emp-Rec}\frac{}{\Gamma \vdash \bot_{\text{rec}} \equiv \bot_{\text{rec}} : \bot \to A}$$

$$\textsc{Sym}\frac{\Gamma \vdash M \equiv M' : A}{\Gamma \vdash M' \equiv M : A} \qquad \textsc{Trans}\frac{\Gamma \vdash M \equiv M' : A \quad \Gamma \vdash M' \equiv M'' : A}{\Gamma \vdash M \equiv M'' : A}$$

2 Sheaves

Since we refer to more historical notion of sheaves and topoï are a set theorist approach of type theory, the meta-theory and language of subsections 2.1 and 2.3 are more alike set theory. Since the definition are presented by way of example, this should not be an issue.

2.1 Sheaves on topological spaces

While the notion of sheaves on topological space is the most distant from my own, we find that they offer a clearer insight on what sheaves are going to be used than sheaves on a grothendieck topology, and as such present the first and not the second in detail. Treating proof irrelevant proposition as subset of a space is a notion we like to keep in mind.

We fix a topological space X, in this case presheaves on that space are presheaves on the category underlying its poset of open subsets. More precisely :

Definition 2.1 (Presheaf on a topology)**.** *A* presheaf P *is given by*

- *For every open* $U \in \mathcal{O}(X)$*, a set* $P(U)$ *of* sections
- *For every open* U*, section* $s \in P(U)$*, and open subset* $V \subseteq U$*, a* restriction $s|_V \in P(V)$
- *If* $s \in P(U)$*,* $s|_U = s$
- *If* $s \in P(U)$*,* $W \subseteq V \subseteq U$*,* $(s|_V)|_W = s|_W$

The $\lambda(s \in P(U)), s|_V : P(U) \to P(V)$ function gives the $P(f)$ necessary to construct the functor $P : \mathcal{O}(X)^{op} \to$ **Set** justifying the use of the same term presheaf here and in the categorical context.

While in general presheaves are unrelated to sheaves, we can introduce what sheaves look like in the topos (see definition 2.4) of presheaves.

Definition 2.2 (Sheaf on a topology)**.** *A* sheaf $\mathcal{F}$ *is a presheaf such that given any open* U *and family of open* $(U_i)_i$ *such that* $U = \bigcup_i U_i$*, there is a bijection between :*

- *Sections* s *of* U
- Compatible *families* $(s_i \in \mathcal{F}(U_i))_i$ *of sections where* $s_i|_{U_i \cap U_j} = s_j|_{U_i \cap U_j}$

where the forward direction is given by $s \mapsto (s|_{U_i})_i$*.*

Since the definitions only access the open subsets and not the points of the space, they can be extended to *locales*, that is, posets with finite meets and arbitrary joins satisfying the infinite distributive law. As far as the reader is concerned, this only means $\vee, \wedge$ and $\leq$ will be used instead $\cup, \cap$ and $\subseteq$. We will stick with calling their elements opens, however.

2.2 Kripke and Beth semantics

Kripke models are models of intuitionistic logic, and come from interpreting the sorts of the language as presheaves.

To construct a Kripke model, we consider a locale (or in general, a small category). We consider a language and associate a downward closed set of opens to each of its propositional variables.

We can then define $U \Vdash \phi$, meaning U forces ϕ, by induction on the structure of ϕ.

- $U \Vdash \phi \wedge \psi$ iff for all $V \leq U$, $V \Vdash \phi$ and $V \Vdash \psi$

- $U \Vdash \phi \Rightarrow \psi$ iff for all $V \leq U$, if $V \Vdash \phi$ then $V \Vdash \psi$
- $U \Vdash \forall x : P, \phi$ iff for all $V \leq U$ and $s \in P(V)$, $V \Vdash \phi(s/x)$

And all other case are similar. We can then prove that for all ϕ, $U \Vdash \phi$ iff for all $V \leq U$, $V \Vdash \phi$, making all cases but $\Rightarrow$ and $\forall$ degenerate. For example, $U \Vdash \phi \wedge \psi$ iff for all $V \leq U$, $V \Vdash \phi$ and $V \Vdash \psi$ iff $U \Vdash \phi$ and $V \Vdash \psi$. This recovers the usual definition.

We note that by using the poset of contexts, Kripke models done with small categories are (only classicaly) complete for intuitionistic logic.

Sheaves also yield a more general kind of model, Beth models, where sorts are interpreted as sheaves.

$U \Vdash \phi$ is now defined as follows :

- $U \Vdash \phi \wedge \psi$ iff for all $V \leq U$ there exists $(V_i)_i$ with $V = \bigvee_i V_i$, such that for all i, $V_i \Vdash \phi$ and $V_i \Vdash \psi$.
- $U \Vdash \phi \vee \psi$ iff for all $V \leq U$ there exists $(V_i)_i$ with $V = \bigvee_i V_i$, such that for all i, $V_i \Vdash \phi$ or $U_i \Vdash \psi$.
- $U \Vdash \exists x : \mathcal{F}, \phi$ iff for all $V \leq U$, there exists $(V_i)_i$ with $V = \bigvee_i V_i$, such that for all i, there exists $s_i \in \mathcal{F}(V_i)$, $V_i \Vdash \phi(s_i/x)$.
- $U \Vdash \phi \Rightarrow \psi$ iff for all $V \leq U$ there exists $(V_i)_i$ with $V = \bigvee_i V_i$, such that for all i, if $V_i \Vdash \phi$ then $U_i \Vdash \psi$.
- $U \Vdash \forall x : \mathcal{F}, \phi$ iff for all $V \leq U$, there exists $(V_i)_i$ with $V = \bigvee_i V_i$, such that for all i and $s_i \in \mathcal{F}(V_i)$, $V_i \Vdash \phi(s_i/x)$.

Now it can also be proved that $U \Vdash \phi$ iff there exists $(V_i)_i$ with $V = \bigvee_i V_i$, for all i $V_i \Vdash \phi$, and the above degenerates into :

- $U \Vdash \phi \wedge \psi$ iff $U \Vdash \phi$ and $U \Vdash \psi$.
- $U \Vdash \phi \vee \psi$ iff there exists $(U_i)_i$ with $U = \bigvee_i U_i$, such that for all i, $U_i \Vdash \phi$ or $U_i \Vdash \psi$.
- $U \Vdash \exists x : \mathcal{F}, \phi$ iff there exists $(U_i)_i$ with $U = \bigvee_i U_i$, such that for all i, there exists $s_i \in \mathcal{F}(U_i)$, $U_i \Vdash \phi(s_i/x)$.
- $U \Vdash \phi \Rightarrow \psi$ iff for all $V \leq U$, if $V \Vdash \phi$ then $V \Vdash \psi$.
- $U \Vdash \forall x : \mathcal{F}, \phi$ iff for all $V \leq U$ and $s \in \mathcal{F}(V)$, $V \Vdash \phi(s/x)$.

They are other variant of Beth semantics, relying on different notion of *cover* in small categories.

2.3 Topoï and sheaves in topoï

Proofs, results and details for this section can often be found in *Sheaves in geometry and logic: A first introduction to topos theory* by Saunders Maclane and Ieke Moerdijk [2].

Definition 2.3 (Subobject)**.** *In a category, a* subobject *of X is an equivalence class of monomorphism $m : A \rightarrowtail X$, where the equivalence comes from the preorder where $m : A \rightarrowtail X$ is smaller than $m' : A' \rightarrowtail X$ when there is a map $f : A \to A'$ with $m' \circ f = m$.*

We deduce a presheaf **Sub** where $\mathbf{Sub}(X)$ is the the set of subobjects of X, and $\mathbf{Sub}(f) : \mathbf{Sub}(Y) \to \mathbf{Sub}(X)$ sends $m : A \to Y$ to its pullback by $f : X \to Y$.

Definition 2.4 (Topos)**.** *A* topos *is a cartesian closed category with all finite limits and a suboject* classifier Ω *and an isomorphism $\mathbf{Sub}(X) \cong \mathbf{Hom}(X, \Omega)$ natural in X.*

We note that topoï also have finite colimits.

Topoï form a class of models of type theory. However, they are reasoned upon as though they were a type theory with *e.g.* equality reflection, propositional extensionnality, strict propositions and the axiom of unique choice. This properties form a very strong, but also undecidable, system. In particular they are no implementation of type theory with both unique choice and strict propositions. They also have no universe in general, which, on the other hand, make them very weak.

To give a few examples of topoï, **Set** is a topos, and given a topos $\mathcal{E}$, $\mathcal{E}/X$, the category of maps with codomain X and commuting triangles, as well as $\mathcal{E}^{\mathbf{C}^{op}}$, the category of contravariant functors from a small category $\mathbf{C}$ and natural transformation, are all topoï. In particular categories of presheaves are topoï, and their internal language

are given by Kripke models. The categories of sheaves over topological spaces also form topoï, and their internal language are given by Beth models.

The subobject classifier Ω is equipped with an internal meet-semilattice structure inherited from the meet-semilattice structure on each $\mathbf{Sub}(X)$, which is natural in X.

Definition 2.5 (Lawvere-Tierney topology)**.** *A* Lawvere-Tierney topology *is a left exact idempotent monad j on the internal meet-semilattice on Ω.*

- $id_\Omega \leq j$,
- $j \circ j \leq j$
- $j \circ \wedge = \wedge \circ j \times j$

From a topology j we extract a closure operator J_X of $\mathbf{Sub}(X)$ for any X.

Definition 2.6 (Dense subobject)**.** *A suboject U of X is dense if $J_X U = X$*

A topology can be lifted to a left exact idempotent monad on the entirety of the topos, the sheafification monad.

Definition 2.7 (j-Sheaf in topos)**.** *An object F is a j-*sheaf *in a topos if for any dense subobject U of any object X, the morphism $\mathbf{Hom}(X, F) \to \mathbf{Hom}(U, F)$ obtained by precomposition is an isomorphism.*

A j-Sheaf is up to isomorphism the result of sheafifying an object.

j-Sheaves form a topos.

In the topos of presheaf over X, elements of Ω are downward closed sets of open. The j operator send such set to the set of sub-open of their union. A subobject A of P is then dense if for every open U and every section s of $P(U)$, there is a family of opens and sections $s_i \in U_i$ which can be glued back into s, *i.e.* $U = \bigvee U_i$ and $s_i = s|_{U_i}$. Although, to recover the earlier sheaf definition, one can simply look at presheaves P where $P(U)$ has at most one element, since morphisms out of these presheaves directly yield compatible families.

2.4 Geometric formulas

In the litterature around sheaves and forcing, we find the following definitions.

Not-a-Definition 2.8. *A* geometric formula *is a formula built from $\exists, \wedge, \bigvee$, and atomic formulas.*

Not-a-Definition 2.9. *A* geometric implication *is a formula of shape $\forall \overrightarrow{x}, \phi(\overrightarrow{x}) \to \psi(\overrightarrow{x})$, where ϕ and ψ are geometric.*

Not-a-Definition 2.10. *A* geometric theory *is a set geometric implication.*

We rather use the following definition

Definition 2.11. *A* geometric formula *is a formula of the form*

$$\bigwedge_{i:I} \left(\forall \overrightarrow{x}, \bigwedge_{j:J_i} O_{i,j}(\overrightarrow{x}) \to \bigvee_{k:K_i} \exists \overrightarrow{y}, \bigwedge_{l:L_{i,k}} Q_{i,k,l}(\overrightarrow{x}, \overrightarrow{y}) \right)$$

Where $O_{i,j}$ and $Q_{i,k,l}$ are atomic formulas, and both every J_i and every $L_{i,k}$ are finite.

Every « geometric theory » can be written as a single geometric formula.

We'd like to note that they define covers and can be read as saying that some intersection of atoms can be covered by a family of intersection of atoms, and that they may be integrated in a deduction system as :

$$\textsc{Geo-}i \frac{Q_{i,1,1}(\overrightarrow{x}, \overrightarrow{t_1}) \dots Q_{i,1,m_{i,1}}(\overrightarrow{x}, \overrightarrow{t_1}) \vdash \mathcal{J} \ \cdots \ Q_{i,p_i,1}(\overrightarrow{x}, \overrightarrow{t_{p_i}}) \dots Q_{i,p_i,m_{i,p_i}}(\overrightarrow{x}, \overrightarrow{t_{p_i}}) \vdash \mathcal{J}}{O_{i,1}(\overrightarrow{x}), \dots O_{i,n_i}(\overrightarrow{x}) \vdash \mathcal{J}}$$

2.5 Sheaves in type theory

Consider a type theory with a notion of proof irrelevant propositions Prop, *e.g.* book-HoTT with mere propositions, or Rocq with SProp.

In this case, a Lawvere-Tierney topology may be similarily defined, as a monad:

- $\mathsf{J} : \mathsf{Prop} \to \mathsf{Prop}$
- $\eta : \Pi(P : \mathsf{Prop}).P \to \mathsf{J}\ P$
- $\mathsf{bind} : \forall(PQ : \mathsf{Prop}).\mathsf{J}\ P \to (P \to \mathsf{J}\ Q) \to \mathsf{J}\ Q$

Then a sheaf is just a type T with

- A map $\mathsf{ask}_T : \Pi(P : \mathsf{Prop}).\ \mathsf{J}\ P \to (P \to T) \to T$
- A coherence $\varepsilon_T : \Pi(P : \mathsf{Prop})\ (j : \mathsf{J}\ P)\ (x : T).\ \mathsf{ask}_T\ \ P\ j\ (\lambda p : P.x) = x$

Now, the sheafified of a type doesn't exists in general, if the theory admits quotient inductive types, it can then be defined as follow :

$$
\begin{array}{l}
\mathsf{Inductive}\ \mathcal{S}_\mathsf{J}\ T : \mathsf{Type} := \\
\mid \mathsf{ret} : T \to \mathcal{S}_\mathsf{J}\ T \\
\mid \mathsf{ask} : \Pi(P : \mathsf{Prop}), \mathsf{J}\ P \to (P \to \mathcal{S}_\mathsf{J}\ T) \to \mathcal{S}_\mathsf{J}\ T \\
\mid \varepsilon : \Pi(P : \mathsf{Prop})\ (j : J\ P)\ (x : \mathcal{S}_\mathsf{J}\ T).\mathsf{ask}\ \ P\ j\ (\lambda p : P.x) = x
\end{array}
$$

We note that by taking $I := \Sigma(P : \mathsf{Prop}).\mathsf{J}\ P$ and $O\ (P, j) : P$, a sheaf is then defined as

Definition 2.12. *A* sheaf *or* (I, O)-sheaf*, is given by*

- *A type* T
- *A map* $\mathsf{ask}_T : \Pi(i : I), (O\ i \to T) \to T$
- *A coherence map* $\varepsilon_T : \Pi(i : I)\ (x : T), \mathsf{ask}_T\ \ i\ (\lambda o : O\ i.x) = x$

(I, O)-Sheafification can be defined similarily. This definition is marginally simpler, and make sheaves appear as quotient dialogue trees, hence why we will henceforth consider (I, O)*-sheaves* instead of J*-sheaves*.

It is also possible to see geometric formulas as being of the form $\Pi i : I, Oi$, and as such be used as the basis for sheaves.

3 Models

3.1 Categories with family

A common manner to construct models of type theory is through categories with families.

They are constituted of :

- A category of context and substitution
- For every context a type of inner types, and for every context and inner type in that context, a type of terms, both accompanied by the action of the substitution upon those, and the corresponding rules.
- A way to append a type to a context, substitution weakening, lifting and extension.

They may be complemented by :

- Dependent function inner types, with their constructor, eliminator, and their substitution laws.
- A universe inner type, and its element function and their substitution laws.
- Some positive types, like a boolean type, its eliminator and constructors, and their substitution laws.
- Dependent pair inner types, eliminators, constructors, and laws.

A significant part of my internship was dedicated to contructing models of type theory in Rocq, using categories with families.

The most notable being :

1. A model using dialogue trees.
2. An exceptional model.
3. A model using presheaves.
4. A model using (I, O)-sheaves, which requires univalence, and quotient inductive types to model positive types.

None of them completely implement all the feature listed above, but only the model using presheaves has no dependent function type.

Rocq served as both the meta-theory and the target.

3.2 Dialogue trees

Given a $I : \mathsf{Type}$ and $O : I \to \mathsf{Type}$, an (I, O)-dialogue tree, is a type A together with a map $\Pi i : I, (O\ i \to A) \to A$. More important, maybe, are how free dialogue tree are defined :

$$\begin{array}{l}\mathsf{Inductive}\ \ \mathcal{D}_{I,O}\ T : \mathsf{Type} := \\ |\ \mathsf{ret} : T \to \mathcal{D}_{I,O}\ T \\ |\ \mathsf{ask} : \Pi(i : I), (O\ i \to \mathcal{D}_{I,O}\ T) \to \mathcal{D}_{I,O}\ T\end{array}$$

They encode a fairly large family of monads. They are quite similar to sheaves, however, unlike sheaves, a model interpreting types as dialogue trees do not require no univalence nor HITs, but cannot implement booleans (or in general, inductive types) as usual.

Instead, they implement them in the way of Baclofen TT, meaning that, using booleans as an example, the eliminator for types $\mathbf{B}_{\mathrm{ind}} : \Pi P : \square_\ell, P \to P \to \mathbf{B} \to P$ is separate from the eliminator for type families $\mathbf{B}_{\mathrm{rec}} : \Pi P : \mathbf{B} \to \square_\ell, P\,\mathsf{tt} \to P\,\mathsf{ff} \to \Pi b : \mathbf{B}, \theta_{\mathbf{B}} P b$. $\mathbf{B}_{\mathrm{ind}}$ behaves like the usual eliminator applied to a constant predicate. $\mathbf{B}_{\mathrm{rec}}$ involves $\theta_{\mathbf{B}} P$, the linearisation of P defined as $\theta_{\mathbf{B}} := \lambda P n, \mathbf{B}_{\mathrm{ind}}((\mathbf{B} \to \square_\ell) \to \square_\ell)(\lambda Q, Q\,\mathsf{tt})(\lambda Q, Q\,\mathsf{ff}) b P$, or, more clearly, $\theta_{\mathbf{B}} P\,\mathsf{tt} := P\,\mathsf{tt}$ and $\theta_{\mathbf{B}} P\,\mathsf{ff} := P\,\mathsf{ff}$.

While our model doesn't model integers, it should also be possible, with the eliminator for types as $\mathbf{N}_{\mathrm{ind}} : \Pi P : \square_\ell, P \to (\mathbf{N} \to P \to P) \to \mathbf{N} \to P$ and the eliminator for type families as $\mathbf{N}_{\mathrm{rec}} : \Pi P : \mathbf{N} \to \square_\ell, P0 \to (\Pi n : \mathbf{N}, Pn \to P(Sn)) \to \Pi n : \mathbf{N}, \theta_{\mathbf{N}} P n$. $\mathbf{N}_{\mathrm{ind}}$ also behaves like the usual eliminator applied to a constant predicate, and $\mathbf{N}_{\mathrm{rec}}$ also involves its linearisation of P named $\theta_{\mathbf{N}} P$, defined as $\theta_{\mathbf{N}} := \lambda P n, \mathbf{N}_{\mathrm{ind}}((\mathbf{N} \to \square_\ell) \to \square_\ell)(\lambda Q, Q0)(\lambda n \theta' Q, \theta'(Q \circ S)) n P$, or, more clearly, $\theta_{\mathbf{N}} P 0 := P0$ and $\theta_{\mathbf{N}} P(Sn) := \theta_{\mathbf{N}}(P \circ S) n$. In particular, they coincide on integers of the form $S(\cdots(S0)\cdots)$.

Other writers may swap ind and rec.

This models highlights how sheaves do not need to weaken the theory down to Baclofen TT to add effects

3.3 Exceptional

The exceptional model (for exception type E) interprets a type as a pair of a type and a map $E \to A$. It is inconsistent, however, together with parametricity, a consistent model can be recovered.

This models highlights the importance of computations, since an inconsitent type theory can still be used to state a few thing as long as it computes properly.

3.4 Presheaves

The most time consuming model, where neither universes nor dependant function types could be implemented. The main issue is the straightforward implementation is not strict, meaning equalities have to be proven, and transport along them must be handled.

This model highlights why studying sheaves separately from presheaves is interesting.

3.5 Sheaves

The most important one is of course the model using sheaves.

The model requires univalence and used rewrite rules to simulate HIT, which are absent in Rocq 9.0.0.

Types were interpreted as sheaves as described in definition 2.12. Product types are relatively easy as an aribtray product of sheaves is still a sheaf. Positives types, in my case boolean, where contructed as the QIT with the same constructor as the original (true and false), together with ask and ε. The resulting type is equivalent to the sheafification of booleans but does not compute exactly the same.

One of the interest of type theory is that it yield meaningful computations from proof, so we were careful about that.

4 ShTT

4.1 Martin Baillon's ShTT

My work was meant to generalise the work my supervisor past PhD student, Martin Baillon, did in his thesis [1]. He worked on a roughly similar MLTT extended with boolean and a generic function $\alpha : \mathbf{N} \to \mathbf{B}$, which is known only through finite approximation.

The sheaf structure is the one appearing in this geometric formula :

$$\left(\bigwedge_{n:\mathfrak{N}} \top \to \alpha\overline{n} = \mathsf{tt} \vee \alpha\overline{n} = \mathsf{ff}\right) \wedge \left(\bigwedge_{n:\mathfrak{N}} \alpha\overline{n} = \mathsf{tt} \wedge \alpha\overline{n} = \mathsf{ff} \to \bot\right)$$

Where $\overline{n}$ is the term $S(\cdots(S\ 0)\cdots)$, with n occurrences of S.

The term then extends those of MLTT with

$$M, N ::= \ldots | \alpha$$

With *forcing contexts*, with n and b an integer and boolean respectively

$$\mathcal{L} ::= \mathcal{L}, n \mapsto b | \cdot$$

We write $n \mapsto_{\mathcal{L}} b$ when $n \mapsto b$ appears in $\mathcal{L}$, and $n \not\mapsto_{\mathcal{L}}$ when neither $n \mapsto \mathsf{tt}$ nor $n \mapsto \mathsf{ff}$ do.

And conversion rules :

$$\mathrm{n}\frac{\mathcal{L}, \Gamma \vdash \mathcal{J}_0 \quad \cdots \quad \mathcal{L}, \Gamma \vdash \mathcal{J}_n}{\mathcal{L}, \Gamma \vdash \mathcal{J}}$$

whenever the following is a rule of MLTT (with booleans)

$$\mathrm{n}\frac{\Gamma \vdash \mathcal{J}_0 \quad \cdots \quad \Gamma \vdash \mathcal{J}_n}{\Gamma \vdash \mathcal{J}}$$

With $\mathcal{J}$ a judgment, (conversion or well formation) and n the name of the rule. Exception made of Wf-Empty, which becomes

$$\textsc{Wf-Empty}\frac{}{\cdot, \cdot \vdash \mathsf{well\text{-}formed}}$$

The new conversion rules are :

$$\textsc{Wf-Ext-Forc}\frac{\mathcal{L}, \Gamma \vdash \mathsf{well\text{-}formed} \quad n \not\mapsto_{\mathcal{L}}}{\mathcal{L}, n \mapsto b, \Gamma \vdash \mathsf{well\text{-}formed}}$$

$$\textsc{Gen}\frac{\mathcal{L}, \Gamma \vdash \mathsf{well\text{-}formed}}{\mathcal{L}, \Gamma \vdash \alpha \equiv \alpha : \mathbf{N} \to \mathbf{B}} \qquad \textsc{Ask}\frac{\mathcal{L}, \Gamma \vdash \mathsf{well\text{-}formed} \quad n \mapsto_{\mathcal{L}} b}{\mathcal{L}, \Gamma \vdash \alpha\overline{n} \equiv \overline{b}}$$

$$\textsc{Split}\frac{\mathcal{L}, n \mapsto \mathsf{tt}, \Gamma \vdash M \equiv M' : A \quad \mathcal{L}, n \mapsto \mathsf{ff}, \Gamma \vdash M \equiv M' : A \quad n \not\mapsto_{\mathcal{L}}}{\mathcal{L}, \Gamma \vdash M \equiv M' : A}$$

Amongst other thing, this theory can be used to show that from any term $\cdot \vdash M \equiv M : (\mathbf{N} \to \mathbf{B}) \to \mathbf{N}$ of MLTT, a term $\cdot \vdash \{M\} \equiv \{M\} : \mathsf{continuous}\ M$ of MLTT proving its continuity can be constructed.

4.2 ShTT

Being sheaves, the model of every type contained a map ask, we sought to create our type theory of sheaves by adding this map to the syntax under the name $\digamma$

We suppose we have a type Ω of *atoms*. We then set a type I with decidable equality, and a (morally pointwise finite) type family $A : I \to \mathsf{Type}$ standing for arities, and finally a family $O : \Pi i : I, A\ i \to \Omega$, writing $O_{i,\alpha}$ instead of $O\ i\ \alpha$ for space concerns.

This roughly correspond to the geometric formula : $\bigwedge_{i:I} \left(\top \to \bigvee_{\alpha:A\ i} O_{i,\alpha} \right)$

The terms extend those of MLTT as follows :

$$M, N ::= \ldots | \digamma_i (M_\alpha)_{\alpha:A\ i}$$

When instanciating with finite Ai, it would rather be $\digamma_i M_1 \cdots M_n$.

The forcing contexts $\mathcal{L}$ are now subset of Ω (lists accessed only through wether they include some elements), and never cause ill-formation.

The conversion rules also copy those from MLTT by adding a forcing context as in 4.1, without the WF-Empty exception.

The new conversion rules are as follows :

$$\textsc{Dig-}i\ \frac{\forall \alpha \alpha'. \mathcal{L}, O_{i,\alpha}, O_{i,\alpha'}, \Gamma \vdash M_\alpha \equiv M'_{\alpha'} : A}{\mathcal{L}, \Gamma \vdash \digamma_i (M_\alpha)_\alpha \equiv \digamma_i (M'_\alpha)_\alpha : A}$$

This rule is both the expected congruence rule for conversion and also a compatibility rule for typing that would be stated separately in a system with a pure typing judgement.

$$\textsc{Ask-}i\ \frac{\forall \alpha \alpha'. \mathcal{L}, O_{i,\alpha}, O_{i,\alpha'}, \Gamma \vdash M_\alpha \equiv M'_{\alpha'} : A}{\mathcal{L}, O_{i,\alpha}, \Gamma \vdash \digamma_i (M_{\alpha'})_{\alpha'} \equiv M'_\alpha : A}$$

$$\textsc{Dig-Ev}\ \frac{\mathcal{L}, \Gamma \vdash N \equiv N' : A \qquad \forall \alpha \alpha', \mathcal{L}, O_{i,\alpha}, O_{i,\alpha}, \Gamma \vdash M_\alpha \equiv M_{\alpha'} : \Pi x : A, B}{\mathcal{L}, \Gamma \vdash (\digamma_i (M_\alpha)_\alpha) N \equiv \digamma (M_\alpha N)_\alpha : B(N/x)}$$

$$\textsc{Int-Rec-Dig}\ \frac{\begin{array}{c}\mathcal{L}, \Gamma \vdash A \equiv A : \mathbf{N} \to \square_{\mathbf{s}} \qquad \mathcal{L}, \Gamma \vdash M_0 \equiv M'_0 : \mathbf{N} \qquad \mathcal{L}, \Gamma \vdash M_S \equiv M'_S : \Pi n : \mathbf{N}, An \to A(Sn) \\ \forall \alpha \alpha', \mathcal{L}, O_{i,\alpha}, O_{i,\alpha}, \Gamma \vdash M_\alpha \equiv M_{\alpha'} : \mathbf{N}\end{array}}{\mathcal{L}, \Gamma \vdash \mathbf{N}_{\mathrm{rec}} A M_0 M_S (\digamma_i (M_\alpha)_\alpha) \equiv \digamma_i (\mathbf{N}_{\mathrm{rec}} A M_0 M_S M_\alpha)_\alpha : A \digamma_i (M_\alpha)_\alpha}$$

$$\textsc{Emp-Rec-Dig}\ \frac{\mathcal{L}, \Gamma \vdash A \equiv A : \bot \to \square_{\mathbf{s}} \qquad \forall \alpha \alpha', \mathcal{L}, O_{i,\alpha}, O_{i,\alpha}, \Gamma \vdash M_\alpha \equiv M_{\alpha'} : \bot}{\mathcal{L}, \Gamma \vdash \bot_{\mathrm{rec}} A (\digamma_i (M_\alpha)_\alpha) \equiv \digamma_i (\bot_{\mathrm{rec}} A M_\alpha)_\alpha : A \digamma_i (M_\alpha)_\alpha}$$

This theory is still incomplete. For example, to instantiate it with 4.1, even adding the necessary generic function α, and setting $I := \mathbf{N}$, $Ai = \mathbf{B}$ and $O_{n,b} := n \mapsto b$ so that Dig-i may play the role of Split, we have no way to provide the compatibility proof $n \mapsto \mathsf{tt}, n \mapsto \mathsf{ff} \vdash M_{\mathsf{tt}} \equiv M_{\mathsf{ff}}$, which must be some form of ex-falso. Ask-i cannot be used in stead of Ask either.

5 Logical relations

A rather standard manner of studying the property of a type theory is through following a recipe CITE GIRARD ?

5.1 System T extension

To get a hang of $\digamma$ and logical relations I started by studying a extension of system with $\digamma$

Again the terms simply extend those of system T :

$$M, N ::= \ldots | \digamma_i (M_\alpha)_\alpha$$

And rules from system are appropriately completed with a forcing context.

$$\textsc{Dig-}i\ \frac{\forall \alpha \alpha'. \mathcal{L}, O_{i,\alpha}, O_{i,\alpha'}, \Gamma \vdash M_\alpha \equiv M'_{\alpha'} : A}{\mathcal{L}, \Gamma \vdash \digamma_i (M_\alpha)_\alpha \equiv \digamma_i (M'_\alpha)_\alpha : A}$$

$$\textsc{Ask-}i\ \frac{\forall \alpha\alpha'.\mathcal{L}, O_{i,\alpha}, O_{i,\alpha'}, \Gamma \vdash M_\alpha \equiv M'_{\alpha'} : A}{\mathcal{L}, O_{i,\alpha}, \Gamma \vdash \digamma_i(M_{\alpha'})_{\alpha'} \equiv M'_\alpha : A}$$

$$\textsc{Dig-Ev}\ \frac{\mathcal{L}, \Gamma \vdash N \equiv N' : A \quad \forall \alpha\alpha', \mathcal{L}, O_{i,\alpha}, O_{i,\alpha}, \Gamma \vdash M_\alpha \equiv M_{\alpha'} : A \to B}{\mathcal{L}, \Gamma \vdash (\digamma_i(M_\alpha)_\alpha)N \equiv \digamma(M_\alpha N)_\alpha : B}$$

$$\textsc{Int-Rec-Dig}\ \frac{\mathcal{L}, \Gamma \vdash M_0 \equiv M'_0 : \mathbf{N} \quad \mathcal{L}, \Gamma \vdash M_S \equiv M'_S : \mathbf{N} \to A \to A \quad \forall \alpha\alpha', \mathcal{L}, O_{i,\alpha}, O_{i,\alpha}, \Gamma \vdash M_\alpha \equiv M_{\alpha'} : \mathbf{N}}{\mathcal{L}, \Gamma \vdash \mathbf{N}_{\mathrm{rec}} M_0 M_S (\digamma_i(M_\alpha)_\alpha) \equiv \digamma_i(\mathbf{N}_{\mathrm{rec}} M_0 M_S M_\alpha)_\alpha : A}$$

$$\textsc{Emp-Rec-Dig}\ \frac{\forall \alpha\alpha', \mathcal{L}, O_{i,\alpha}, O_{i,\alpha}, \Gamma \vdash M_\alpha \equiv M_{\alpha'} : \bot}{\mathcal{L}, \Gamma \vdash \bot_{\mathrm{rec}}(\digamma_i(M_\alpha)_\alpha) \equiv \digamma_i(\bot_{\mathrm{rec}} M_\alpha)_\alpha : A}$$

5.2 Reducibility

We define our reduction relation as follows :

$$\frac{M \rightsquigarrow M'}{MN \rightsquigarrow M'N} \qquad \frac{}{(\lambda x.M)N \rightsquigarrow M(N/x)}$$

$$\frac{M \rightsquigarrow M'}{\mathbf{N}_{\mathrm{rec}} M_0 M_S M \rightsquigarrow \mathbf{N}_{\mathrm{rec}} M_0 M_S M'} \qquad \frac{}{\mathbf{N}_{\mathrm{rec}} M_0 M_S O \rightsquigarrow M_0} \qquad \frac{}{\mathbf{N}_{\mathrm{rec}} M_0 M_S (SM) \rightsquigarrow M_S M (\mathbf{N}_{\mathrm{rec}} M_0 M_S M)}$$

$$\frac{M \rightsquigarrow M'}{\bot_{\mathrm{rec}} M \rightsquigarrow \mathbf{N}_{\mathrm{rec}} M'}$$

And our neutrals as follows, through the predicate Ne :

$$\frac{}{\mathrm{Ne}\, x} \qquad \frac{\mathrm{Ne}\, n}{\mathrm{Ne}\, n M} \qquad \frac{\mathrm{Ne}\, n}{\mathrm{Ne}\, \mathbf{N}_{\mathrm{rec}} M_0 M_S n} \qquad \frac{\mathrm{Ne}\, n}{\mathrm{Ne}\, \bot_{\mathrm{rec}} n}$$

We can then define a reducibility predicate $\Vdash$ as follows, by induction on the type.

- For type $A \to B$, $\mathcal{L}, \Gamma \Vdash M \equiv M' : A \to B$, if for all $\mathcal{L}', \Gamma'$ such that $\mathcal{L} \subset \mathcal{L}'$, and Γ is a prefix of Γ', if $\mathcal{L}', \Gamma' \Vdash N \equiv N' : A$, then $\mathcal{L}', \Gamma' \Vdash MN \equiv M'N'$.
- For type $\mathbf{N}$, we define inductively :

$$\frac{M \rightsquigarrow^* 0 \quad M' \rightsquigarrow^* 0}{\mathcal{L}, \Gamma \Vdash M \equiv M' : \mathbf{N}} \qquad \frac{M \rightsquigarrow^* SN \quad M' \rightsquigarrow^* SN' \quad \mathcal{L}, \Gamma \Vdash N \equiv N' : \mathbf{N}}{\mathcal{L}, \Gamma \Vdash M \equiv M' : \mathbf{N}}$$

$$\frac{M \rightsquigarrow^* n \quad M' \rightsquigarrow^* n' \quad \mathrm{Ne}\, n \quad \mathrm{Ne}\, n'}{\mathcal{L}, \Gamma \Vdash M \equiv M' : \mathbf{N}}$$

$$\frac{M \rightsquigarrow^* \digamma_i(M_\alpha)_\alpha \quad M' \rightsquigarrow^* \digamma_i(M'_\alpha)_\alpha \quad \forall \alpha, \alpha' \Vdash M_\alpha \equiv M'_{\alpha'} : \mathbf{N}}{\mathcal{L}, \Gamma \Vdash M \equiv M' : \mathbf{N}}$$

$$\frac{M \rightsquigarrow^* \digamma_i(M_\alpha)_\alpha \quad M' \rightsquigarrow^* M'_\alpha \quad O_{i,\alpha} \in \mathcal{L} \quad \forall \alpha, \alpha', \mathcal{L}, \Gamma \Vdash M_\alpha \equiv M'_{\alpha'} : \mathbf{N}}{\mathcal{L}, \Gamma \Vdash M \equiv M' : \mathbf{N}}$$

$$\frac{M \rightsquigarrow^* M_\alpha \quad M' \rightsquigarrow^* \digamma_i(M'_\alpha)_\alpha \quad O_{i,\alpha} \in \mathcal{L} \quad \forall \alpha, \alpha', \mathcal{L}, \Gamma \Vdash M_\alpha \equiv M'_{\alpha'} : \mathbf{N}}{\mathcal{L}, \Gamma \Vdash M \equiv M' : \mathbf{N}}$$

- For type $\bot$, we use a similar definition as above excluding the case mentioning 0 and the case mentioning S

We can then show the usual results, all proven in Rocq :

Lemma 5.1 (reflection)**.** *If* $\mathrm{Ne}\, n$ *and* $\mathrm{Ne}\, n'$*, then* $\mathcal{L}, \Gamma \Vdash n \equiv n' : A$*.*

Lemma 5.2 (reification)**.** *If* $\mathcal{L}, \Gamma \Vdash M \equiv M' : A$*, then* M *and* M' *normalise.*

Theorem 5.3 (soundness)**.** *If* $\mathcal{L}, \Gamma \vdash M \equiv M' : A$*, then if* $\mathcal{L} \subset \mathcal{L}'$*,* $\mathcal{L}', \Gamma' \Vdash \sigma \equiv \sigma' : \Gamma$*,* $\mathcal{L}', \Gamma' \Vdash M\sigma \equiv M'\sigma' : A$

Where substitution are encoded as lists of terms and reducibility of substitution is reducibility of each of these terms. Transitivity (Trans) made up the bulk of the proof.

We can then deduce that our theory normalises.

6 Conclusion

This constitutes a very imcomplete work. I don't think attempting to prove normalisation of ShTT is the next step, same for any theories at the same level of generality. It might be more interesting to either look at continuity for a more general α than in Martin Baillon's work, or find a simple enough theory with non-empty intersections.

References

[1] Martin Baillon. "Continuity in Type Theory". Theses. Nantes Université, Dec. 2023. URL: `https://theses.hal.science/tel-04617881`.

[2] Saunders MacLane and Ieke Moerdijk. *Sheaves in geometry and logic: A first introduction to topos theory*. Springer Science & Business Media, 2012.