Formalization of the Internal Language of a Topos

Johannes Folttmann

Geboren am 7. April 2003 in Soest

18. August 2025

Bachelorarbeit Mathematik

Betreuer: Prof. Dr. Floris van Doorn

Zweitgutachter: Prof. Dr. Philipp Hieronymi

Mathematisches Institut

Mathematisch-Naturwissenschaftliche Fakultät der

Rheinischen Friedrich-Wilhelms-Universität Bonn

Abstract

This thesis is about the formalization of the internal language of a topos in the interactive theorem prover Lean. In particular, the goal is to construct the logical connectives on predicates of this language as morphism in this topos. In this context we must give care to the fact, that this language is in general not classical, but intuitionistic.

The conjunction and (Heyting-) implication can be defined purely out of the axioms of a topos. The disjunctions additionally require the fact that a topos has all finite colimits. To construct the proposition "false" and through it the (pseudo-) negation we require the fundamental theorem of topos theory. It states that all slice categories over objects of a topos are themselves topoi. Thus, a significant part of this formalization's goal is to prove both these statements.

With this formalization we build upon an already existing formalization by C. Conneen. It follows S. Mac Lane' and I. Moerdijk's argumentative path in [MM92], which we therefore also do. The proof, that a topos has all finite colimits, as well as the constructions of the conjunction, disjunction and (Heyting-) implication morphisms were completed. Additionally, we come across a nice way of equipping the type `Type u` of Lean with the stucture of a topos. When formalizing the fundamental theorem of topos theory we see that the basic definitions, which were inherited from Conneen's project, are not optimally chosen; neither for this proof in particular, nor for working on topos theory in Lean in general. Furthermore we argue, that Mac Lane's and Moerdijk's definition of a topos and approach to the proof of the fundamental theorem do not lend themselves very well to the formalization in Lean.

At `https://github.com/CharredLee/topos`, you can access Conneen's complete code.
At `https://github.com/johannesfoltt/topos`, you can access the author's complete code.

Zusammenfassung

Diese Arbeit widmet sich der Formalisierung der inneren Sprache eines Topos' mittels des Beweisassistenten Lean. Ihr Ziel ist insbesondere, die prädikatenlogischen Verknüpfungen dieser Sprache als Morphismen in diesem Topos zu konstruieren. Hierbei muss darauf geachtet werden, dass die Sprache sich im allgemeinen nicht klassisch, sondern intuitionistisch ist.

Die Konjunktion und (Heyting-) Implikation können rein mithilfe der grundlegenden Axiomen eines Topos' konstruiert werden. Die Disjunktion benötigt zusätzlich dazu noch den Fakt, dass ein Topos alle endlichen Kolim-

iten haben. Um die falsche Aussage und durch sie die (Pseudo-) Negation zu definieren benötigen wir den Fundamentalsatz der Topostheorie. Dieser besagt, dass alle Slice-Kategorien über Objekten eines Topos' wiederum Topoi sind. Es ist also auch ein wichtiges Ziel dieser Formalisierung, diese beiden Aussagen zu beweisen.

Mit unserer Formalisierung bauen wir auf einem bereits existierenden Projekt von C. Conneen auf. Dieses Projekt verfolgt den argumentativen Pfad von S. Mac Lane und I. Moerdijk in [MM92], was wir somit auch tun. Der Beweis, dass ein Topos alle endlichen Kolimiten hat, wurde wie auch die Konstruktionen der Morphismen für Konjunktion, Disjunktion und (Heyting-) Implikation wurden fertig gestellt. Zudem stoßen wir auf einen interresanten Weg, den Lean-Typ `Type u` mit der Struktur eines Topos zu versehen.

Bei der Formalisierung des Hauptsatzes der Topostheorie stellt sich heraus, dass die grundlegenden Definitionen, welche aus Conneens Projekt übernommen wurden, nicht optimal gewählt sind; sowohl für diesen Beweis im Spezifischen, als auch für das Arbeiten mit Lean an Topostheorie im Allgemeinen. Darüberhinaus argumentieren wir, dass sich Mac Lanes und Moerdijks Definition eines Topos' und deren Herangehensweise an den Beweis des Fundamentalsatzes nicht gut für die Formalisierung in Lean eignen.

Unter `https://github.com/CharredLee/topos` kann Conneens vollständiger Code gefunden werden.
Unter `https://github.com/johannesfoltt/topos` kann der Code des Autors gefunden werden.

Contents

1 Introduction

A topos (*plural:* topoi) $\mathcal{E}$ is a category with certain nice qualities, which make it behave like the category **SET** of sets.

These properties give rise to a formal language, which we call the internal language, whose terms and formulas are just the morphisms in $\mathcal{E}$. It is higher-order and intuitionistic, meaning that the law of excluded middle (LEM) and the axiom of choice (AC) generally do not hold.

We can use this language to describe and define mathematical objects like group objects, lattice objects, and in certain cases even the real numbers in $\mathcal{E}$ using their familiar definitions found in set theory. This is achieved by using a process analogous to set comprehension.

Among other things, this is nice because it gives us insight into which theorems in "regular" mathematics depend on (LEM) or (AC). For example, in [MM92], the authors construct a topos in which every function on the real numbers (as an object in the topos) is continuous.

The goal of this thesis is to formalize the propositional part (i.e., the connectives $\wedge$, $\vee$, $\Rightarrow$, and $\neg$) of this language in the interactive theorem prover Lean.

1.1 The Lean theorem prover and mathlib

Lean is a programming language and interactive theorem prover based on the formal system of dependent type theory. It was developed by Leonardo Moura and others at Microsoft Research and Carnegie Mellon University.

Within Lean, mathematical objects can be defined, and theorems can be stated and proven. This works by reformulating a theorem's hypotheses and conclusion using what are called tactics. Often, the success and ease of this depend on the right choice of definition for a given mathematical notion.

While this process can be quite tedious and time-consuming, it also comes with several benefits. For one, meticulously going over every single detail in a proof step by step can deepen one's understanding of it and give new insights into the subject.

Another benefit is mathlib, which is an extensive library of mathematical definitions and proofs. It is an ongoing collaborative effort to which hundreds of mathematicians have already contributed. There are countless definitions and theorems already formalized in mathlib, spanning almost all branches of mathematics, from probability theory to category theory. The latter was particularly useful to the formalization accompanying this work.

Lean and mathlib also improve collaboration between mathematicians. They make it easy to agree on common definitions and notation. Furthermore, they make it easy to divide up work amongst multiple collaborators.

This has recently again been demonstrated by a formalization of Carleson's theorem on the convergence of Fourier series by Becker, van Doorn et al. [Bec+25a]. This project is notable because it did not only formalize a known theorem using well-established methods; it also used new methods, which were published alongside the Lean formalization [Bec+25b]. This demonstrates how Lean is already of use in ongoing research.

Another example of the involvement of Lean in groundbreaking original research is the Liquid Tensor Experiment by Commelin, Topaz et al. [Com22]. It was based on a challenge posed by Scholze to the Lean community. In it, he asked the community to formally check a result from his and Clausen's recent work on condensed mathematics so that they could have a high degree of confidence in its validity. This shows how Lean is already being used as a "standard for mathematical correctness."

Other notable projects include the ongoing formalization of Fermat's Last Theorem by Buzzard et al., the formalization of the independence of the Continuum Hypothesis by van Doorn and Han, and the formalization of the proof of the polynomial Freiman-Rusza conjecture by Tao et al., [BT25], [HD20], [Tao25].

1.2 Topoi, their internal language, and their formalization

Topos theory was independently developed by mathematicians from different fields, primarily F.W. Lawvere in categorical logic and A. Grothendieck in algebraic geometry, and ever since then, they have been of use in a variety of mathematical areas.

For example, topoi can be used as models for higher-order intuitionistic set theory. In 1975, Diaconescu used them as such to prove that in intuitionistic logic, the axiom AC already implies LEM [Dia75].

Topos theory also has some applications in current research. For example, Mahadevan has recently shown that "the category of LLMs [(large language models)] forms a topos" [Mah25].

In this work, we view topoi from the perspective of mathematical logic. To develop the basics of topos theory, we closely follow Mac Lane and Moerdijk's approach as it is laid out in their work [MM92]. In particular, we need to prove the following two facts:

Theorem 1.1 ([MM92, Cor. IV.5.4]). *A topos $\mathcal{E}$ is finitely cocomplete, i.e., it has all finite colimits.*

Theorem 1.2 ([MM92, Thm. IV.7.1]). *For any object X in a topos $\mathcal{E}$, the slice category $\mathcal{E}/X$, consisting of all morphisms into X, is again a topos.*

The second theorem is one part of what is commonly called the fundamental theorem of topos theory. It is necessary for us because it implies that pullbacks preserve colimits.

Arguably the most important notion in topos theory is that of a subobject classifier. It can be thought of as an "object Ω of truth-values" [MM92, p.

32]. This means that morphism into Ω can be viewed as predicates on their domain objects.

Using this insight, we then define the logical connectives for some topos $\mathcal{E}$ as morphisms $\wedge_\Omega, \vee_\Omega, \Rightarrow_\Omega$: $\Omega \times \Omega \to \Omega$ and $\neg_\Omega : \Omega \to \Omega$, where Ω is the subobject classifier of $\mathcal{E}$. For this we use the constructions found in [Kos11, pp. 75f.].

For predicates $\phi, \psi : X \to \Omega$, we can then define new predicates $\phi \Box \psi$ and $\neg\phi$ as the compositions

$$X \xrightarrow{\langle \phi, \psi \rangle} \Omega \times \Omega \xrightarrow{\Box_X} \Omega,$$

where $\Box \in \{\wedge, \vee, \Rightarrow\}$ and $X \xrightarrow{\phi} \Omega \xrightarrow{\neg_\Omega} \Omega$.

We can confirm the correctness of these definitions by checking that the subobjects classified by the aforementioned predicates correspond to the meet, join, and Heyting implication on the lattice of subobjects. For this, we closely follow [Gol14, Chapter 7].

Theorems 1.1 and 1.2 are quite difficult to prove and require the construction of a large amount of theory about topoi. Therefore, the greatest part of the time and effort used for the formalization was spent on formulating and proving them.

This was exacerbated by a realization, which came quite late in the formalization effort, namely during the formalization of the fundamental theorem: The basic definitions used in [Con25] for the definition of a topos (and thus all subsequent definitions and theorems) had to be adjusted and refactored.

Therefore, the author only managed to formalize the definitions of the connectives $\wedge, \vee, \Rightarrow$, and $\neg$ but did not manage to prove any facts about the latter three. However, the conjunction $\wedge$ and some of its properties were used in the proof of the fundamental theorem.

1.3 Thesis structure

We now give a short description of the structure of this thesis.

In Chapter 2, we present background knowledge about some basic aspects of category theory, namely adjunctions, pullbacks, and subobjects. We also give a short description of lattices and Heyting algebras. This section is based on [MM92].

In Chapter 3, we examine the category **SET** and isolate the properties that make it a nice context for mathematical reasoning. We formulate these properties for arbitrary categories, which results in the definitions of subobject classifiers, power objects, and exponential objects. Then, we use them to give our definition of a topos and show some basic properties, which arise from these definitions. This chapter is based on [MM92, pp. I.6, IV.1, IV.2]

In Chapter 4, we first define the notion of direct images in a topos. We then use this notion to prove Theorem 1.1. Lastly, we define the notion of image factorizations in categories and use the finite cocompleteness of a topos to show that it has such factorizations. This section is based on [MM92, pp. IV.3–IV.6]

In Chapter 5, we construct the internal connectives as morphisms on the subobject classifier. Then, we prove that these morphisms "work as intended," meaning that they behave well in relation to the partial order relation on the subobjects of some object X in $\mathcal{E}$. We will use the previously defined connectives to define analogues to the usual set-theoretic operations on power objects. This section is based on [Kos11, p. 75f.] and [Gol14, Chapter 7]

In Chapter 6, we prove Theorem 1.2. For this, we first give a short overview of what a slice category is. Then we construct subobject classifiers and power objects in the slice categories of $\mathcal{E}$, proving Theorem 1.2. Afterward, we explain how this theorem implies the notion that pullbacks preserve colimits and prove some basic consequences of this fact. This chapter is based on [MM92, p. IV.7]

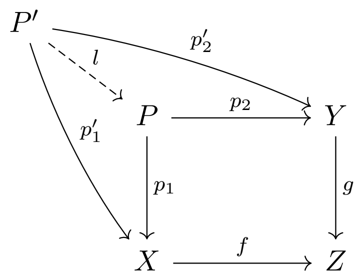
In Chapter 7, we give a short overview of the formalization accompanying this thesis. Firstly, we give some background information about Lean. Then we give an overview of the formalization results. Afterwards, we present an interesting implementation detail about the topos `Type u`. Finally we discuss a design choice faced during the formalization.

2 Some Preliminaries

In this chapter, we will introduce two important concepts from category theory and the notions of lattices and Heyting algebras. These concepts frequently appear in the context of topos theory. We assume that the reader has some background knowledge in category theory. For this, we refer to [MM92, Categorical Preliminaries]. It is of note that we use a rather unusual notation for the composition of morphisms in a category. For two morphisms $f : X \to Y$ and $g : Y \to Z$, we let $f \gg g$ denote their composition. This is done because compositions are represented in this way in Lean, and so the reader has an easier time comparing this thesis with the accompanying code.

2.1 Pullbacks

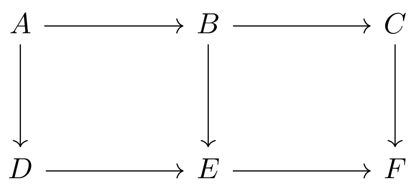
One categorical concept, which is central to the theory of topoi is the notion of pullbacks. In a category $\mathcal{C}$, we call a commutative square consisting of the morphisms p_1, p_2, f, and g, as seen below, a **pullback** if for all other p_1' and p_2' making the outer edges commute, there exists a unique lifting map l such that the entire diagram commutes.



We say that a category **has all pullbacks** if for all morphisms $f : X \to Z$, $g : Y \to Z$, there exists a pair $p_1 : P \to X$, $p_2 : P \to Y$ making the above square a pullback.

In that case, let $X \times_Z Y$, f^*g, and g^*f denote an arbitrary choice of such a P, p_1, and p_2. We call f^*g the **pullback of f along g**. It is a well-known fact that pullbacks of monomorphism along some other morphism are again pullbacks.

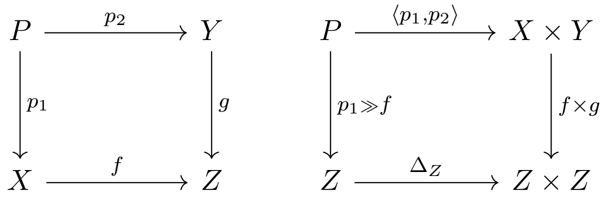
For a commuting diagram of the form



in our category $\mathcal{C}$, we know that if both inner squares are pullbacks, then the outer square is also a pullback. Furthermore, if the outer square and the right square are pullbacks, then the left square is also a pullack. We call these two facts the **pullback pasting lemma** or just **pullback pasting**.

A useful fact when working with pullbacks is the following:

Lemma 2.1. *If $\mathcal{C}$ has binary products and the left square in the following diagram is a pullback, then the right square is a pullback as well.*



2.2 Subobjects

When working in an arbitrary category $\mathcal{C}$, we do not know what the objects of $\mathcal{C}$ are. This means that we have no notion resembling the "elements of" some object X in $\mathcal{C}$.

However, we would still like to define an analogue to the notion of subsets in set theory. We call two monomorphisms $m : S \rightarrowtail X$ and $m' : S' \rightarrowtail X$ equivalent if $m = i \gg m'$ for some isomorphism $i : S \to S'$. This obviously forms an equivalence relation on the class of monomorphisms into X.

Definition 2.2 (Subobjects)**.** A **subobject** of X is an equivalence class of monomorphisms into X under the abovementioned equivalence relation.
For some monomorphism $m : S \rightarrowtail X$, let $[m]$ denote the subobject represented by m.
Let $\mathrm{Sub}_{\mathcal{C}}(X)$ denote the set of all subobjects of X.

The idea behind this definition is that in **SET**, we can identify a subset $U \subset M$ with its inclusion $\iota_U : U \to M$. In this light, a subobject can be thought of as "all injections, which map onto the same subset."

Just like the subsets of some set, we can equip $\mathrm{Sub}_{\mathcal{C}}(X)$ with a partial order. For $[m], [m'] \in \mathrm{Sub}_{\mathcal{C}}(X)$, we say that $[m]$ "is under" $[m']$ (or just $[m] \leq [m']$) if there exists a morphism k such that $k \gg m' = m$ holds.

Now assume that $\mathcal{C}$ has all pullbacks. Using the uniqueness of pullbacks up to isomorphism and the fact that pullbacks of monics are again monic, for any morphism $f : X \to Y$, we get a well-defined mapping

$$\mathrm{Sub}_{\mathcal{C}}(f) : \mathrm{Sub}_{\mathcal{C}}(Y) \to \mathrm{Sub}_{\mathcal{C}}(X);\ [m] \mapsto [f^*m]$$

Now if $k \gg m' = m$ holds for some k, m', m like above, then by pasting pullbacks we see that $f^*k \gg f^*m'$ forms the pullback of m along f (up to unique isomorphism, of course). Thus $[f^*m] \leq [f^*m']$ holds, which means that $\mathrm{Sub}_{\mathcal{C}}(f)$ is an order-preserving function.

With this, we have just shown that $\mathrm{Sub}_{\mathcal{C}}(-) : \mathcal{C}^{\mathbf{op}} \Rightarrow \mathbf{POSET}$ is a (contravariant) functor. Forgetting the partial order gives us a functor $\mathrm{Sub}_{\mathcal{C}}(-) : \mathcal{C}^{\mathbf{op}} \Rightarrow \mathbf{SET}$.

2.3 Lattices and Heyting Algebras

Considering that the main goal of this work is to define connectives for the internal language of a topos, which is intuitionistic, it is useful to know how intuitionistic propositional logic is usually modeled.

For this, we use the notion of Heyting algebras. In essence, they are weaker forms of Boolean algebras, which model classical propositional logic. Both of these are special cases of structures called lattices.

Definition 2.3. A **lattice** is a bounded partially ordered set $(L, \leq, 0_L, 1_L)$, where any two elements $a, b \in L$ have a greatest lower bound $a \wedge_L b$ (also called their **meet**) and a least upper bound $a \vee_L b$ (also called their **join**). This means that for all $x \in L$ with $x \leq a, b$ we have $x \leq a \wedge_L b$ and for all $y \in L$ with $a, b \leq y$ we have $a \vee_L b \leq y$.
A lattice is called a **Heyting algebra** if for all $a, b \in L$ there additionally exists an element $a \Rightarrow_L b$, called their **Heyting implication**, such that for all $x \in L$:

$$(x \wedge_L a) \leq b, \text{ if, and only if } x \leq (a \Rightarrow_L b)$$

Remark 2.4. If $a \leq b$ holds for two elements a, b of a lattice L, then b is their greatest lower bound. On the other hand, if b is their greatest lower bound, then $a \leq b$ holds. Thus we know that $a \leq b$ holds if and only if $a = a \wedge_L b$ does.

These definitions imply the uniqueness of meets, joins, and Heyting implications. Thus they form functions $\wedge_L, \vee_L, \Rightarrow_L\colon L \times L \to L$. Furthermore, for every element $a \in L$, we call $a \Rightarrow_L 0$ its **pseudocomplement** $\neg_L a$. A Heyting algebra is a Boolean algebra if and only if $\neg_L \neg_L a = a$ is true for all $a \in L$.

3 The Definition of a Topos

In this chapter we examine the category **SET** and filter out four properties, which make it a good setting to do mathematics in. These properties are the existence of a subobject classifier, the existence of pullbacks, the existence of a power object, and its Cartesian closedness. This then gives us the following definition of a topos:

Definition 3.1. A topos $\mathcal{E}$ is a category which has

- all pullbacks;
- a subobject classifier;
- power objects;
- exponential objects.

After that, we present some basic consequences of these definitions, most notably the fact that with the other two axioms, the existence of exponential objects and the existence of power objects mutually imply each other. However, we work with this definition because it is also the one we use in the accompanying Lean formalization.

3.1 Subobject Classifiers

Set comprehension is a very useful tool when doing mathematics on the basis of set theory. We now want to generalize this notion to a categorical concept.

For this, it is of note that any predicate $\phi(x)$ on some set M can be interpreted as a function $M \to \{0, 1\}$, sending an element $m \in M$ to 1 (or "true"), if and only if $\phi(m)$ holds.

Now let $f : M' \to M$ be an arbitrary function. If $\phi(f(m'))$ holds for all $m' \in M'$, then

$$f(M') \subset U := \{x \in M \mid \phi(x)\}.$$

In other words, if $f \gg \phi$ factors through the inclusion $\iota_{\{1\}} : \{1\} \rightarrowtail \{0, 1\}$, then f also factors through the inclusion $\iota_U : U \rightarrowtail M$. But this just means that the square

$$\begin{array}{ccc} U & \longrightarrow & \{1\} \\ {\scriptstyle \iota_U}\big\downarrow & & \big\downarrow{\scriptstyle \iota_{\{1\}}} \\ M & \xrightarrow{\ \phi\ } & \{0,1\} \end{array}$$

is a pullback in the category **SET**.

So from a categorical perspective, set comprehension is nothing more than taking the pullback of $\iota_{\{1\}}$ along some predicate ϕ. This is why we will require the existence of pullbacks in the definition of a topos.

Now the behavior that $\{0, 1\}$ exhibits above can be generalized into an arbitrary category as follows:

Definition 3.2. Let $\mathcal{C}$ be a category with a terminal object $\top$.
A **subobject classifier** in $\mathcal{C}$ consists of an object Ω and a monomorphism **true** $: \top \rightarrowtail \Omega$, such that for every monomorphism $m : S \rightarrowtail X$ in $\mathcal{C}$ there exists a unique morphism $\chi_m : X \to \Omega$, called the **characteristic morphism**, making the following square a pullback:

$$\begin{array}{ccc} S & \longrightarrow & \top \\ {\scriptstyle m}\big\downarrow & & \big\downarrow{\scriptstyle \mathbf{true}} \\ X & \xrightarrow{\chi_m} & \Omega \end{array}$$

We call morphisms $\phi : X \to \Omega$ **predicates** on X.

Returning to our example in **SET**, we can see that the object $\{0, 1\}$ forms a subobject classifier together with the inclusion $\iota_{\{1\}} : \{1\} \to \{0, 1\}$. For a subset $U \subset M$, its characteristic morphism is just the usual characteristic function $\mathbb{1}_U : M \to \{0, 1\}$ with $\mathbb{1}_U(m) = 1$, if and only if $m \in U$.

As the name suggests, subobject classifiers behave well with subobjects, in that two morphisms represent the same subobject if and only if they have the same characteristic morphism. This follows from the lifting property of pullbacks and the uniqueness of the characteristic morphism. When we say that a predicate $\phi : X \to \Omega$ characterizes or classifies a subobject $[s] \in \mathrm{Sub}_{\mathcal{C}}(X)$, we mean that $\phi = \chi_s$ holds.

3.2 Power Objects

In **SET**, we can also consider predicates over all the subsets of some set, which lets us interpret higher-order logic. This is due to the existence of power sets. We now want to find a categorical description of how these power sets behave.

For this, let us examine some set M and its power set $\mathcal{P}M$. Let N be an arbitrary set and $R \subset M \times N$ a binary relation with characteristic function $r : M \times N \to \{0, 1\}$.

Now keeping any $n \in N$ fixed gives us a function $r(-, n) : M \to \{0, 1\}$, which characterizes the subset

$$\{x \in M \mid r(x, n) = 1\} \in \mathcal{P}M. \tag{3.1}$$

Hence we can represent r (and thus also R) with a unique function $\widehat{r} : N \to \mathcal{P}M$, called its transpose. Later we see that this transpose is just a special case of a more general phenomenon named currying.

We can recover the original characteristic function r from the transpose $\widehat{r}$ using the usual membership relation $(- \in -) \subset M \times \mathcal{P}M$. Now for a pair $(m, n) \in M \times N$, we know that

$$m \in \widehat{r}(n) = \{x \in M \mid r(x, n) = 1\}, \text{ if and only if } \ r(m, n) = 1.$$

We can express this property in purely categorical terms:

Definition 3.3 (Power object)**.** Let $\mathcal{C}$ be a category with binary products and a subobject classifier Ω, and let X be an object of $\mathcal{C}$.
A **power object** of X consists of an object $\mathcal{P}X$ and a morphism $\in_X : X \times \mathcal{P}X \to \Omega$, such that for every morphism $f : X \times Y \to \Omega$ there exists a unique $\widehat{f} : Y \to \mathcal{P}X$ making the following diagram commute:

$$\begin{array}{ccc} X \times Y & & \\ {\scriptstyle \mathrm{id}_X \times \hat{f}} \downarrow & \searrow^{f} & \\ X \times \mathcal{P}X & \xrightarrow{\in_X} & \Omega \end{array}$$

We call $\widehat{f} : Y \to \mathcal{P}X$ the **transpose** of f and $\in_X$ the **membership predicate** on X.
For any morphism $g : Y \to \mathcal{P}X$, we define its **inverse transpose** $\widetilde{g}$ to be the composition

$$X \times Y \xrightarrow{\mathrm{id}_X \times g} X \times \mathcal{P}Y \xrightarrow{\in_X} \Omega$$

Using the uniqueness of the transpose, it is easy to see that the operations of taking the transpose and taking the inverse transpose are indeed inverses of each other.

Using the transpose, we can define the following map, which can be thought of as pointing out the subset characterized by some predicate ϕ.

Definition 3.4 (Name)**.** Let $\mathcal{C}$ be a category with binary products and a subobject classifier Ω, and let X be an object of $\mathcal{C}$ with a power object $\mathcal{P}X$. Now given a morphism $\phi : X \to \Omega$, we define its **name** $\ulcorner \phi \urcorner : \top \to \mathcal{P}X$ to be the transpose of the map

$$X \times \top \xrightarrow{\pi_1} X \xrightarrow{\phi} \Omega$$

For a monomorphism $s : S \rightarrowtail X$, we define its name $\ulcorner s \urcorner$ to be the name $\ulcorner \chi_s \urcorner$ of its characteristic morphism.

Now every function $f : M \to N$ on sets induces a function $f^{-1} : \mathcal{P}N \to \mathcal{P}M$, mapping a subset of N to its preimage under f. If the category $\mathcal{C}$ has power objects, we can define an analogous notion.

Definition 3.5 (Inverse image)**.** For any morphism $f : X \to Y$ in $\mathcal{C}$, we define its **inverse image** $\mathcal{P}f : \mathcal{P}Y \to \mathcal{P}X$ as the transpose of the morphism

$$X \times \mathcal{P}Y \xrightarrow{f \times \mathrm{id}_{\mathcal{P}Y}} Y \times \mathcal{P}Y \xrightarrow{\in_Y} \Omega.$$

To see why these morphisms can be thought of as preimage morphisms, consider the following square, which commutes by the definition of the transpose:

$$\begin{array}{ccc} X \times \mathcal{P}Y & \xrightarrow{f \times \mathrm{id}_{\mathcal{P}Y}} & Y \times \mathcal{P}Y \\ {\scriptstyle \mathrm{id}_X \times \mathcal{P}f}\downarrow & & \downarrow{\scriptstyle \in_Y} \\ X \times \mathcal{P}X & \xrightarrow[\in_X]{} & \Omega \end{array} \tag{3.2}$$

In **SET**, this diagram can be interpreted as the statement

$$f(x) \in S, \text{ if and only if } x \in \mathcal{P}f(S) \text{ for all } x \in X \text{ and } S \subset Y,$$

for all $x \in X$ and $S \subset Y$. This is just the definition of the familiar inverse image on subsets.

Furthermore, this inverse image is functorial, in the sense that the assingments $X \mapsto \mathcal{P}X$ and $(f : X \to Y) \mapsto (\mathcal{P}f : \mathcal{P}Y \to \mathcal{P}X)$ form a contravariant functor $\mathcal{P}- : \mathcal{E}^{\mathbf{op}} \Rightarrow \mathcal{E}$. To prove this, we simply need to check that

$$\mathcal{P}\mathrm{id}_x = \mathrm{id}_{\mathcal{P}X} \quad \text{and} \quad \mathcal{P}(f \gg g) = (\mathcal{P}g) \gg (\mathcal{P}f)$$

for all objects X and morphisms $f : X \to Y$ and $g : Y \to Z$. Both cases immediately follow from the definitions of power objects and transposes.

3.3 Exponential Objects

Similarly to subsets, we also want to define predicates on some sets of functions. In **SET** this is possible, because for a pair of sets M, N we can define the set N^M of functions $M \to N$.

Now let us examine a function $f : L \to N^M$ into this set. For every $l \in L$ and $m \in M$, we first get a map $f(l) : M \to N$ by applying f to l, which we can then apply to m to get an element $f(l)(m) \in N$. In effect, we have just described a map $\mathbf{uc}(f) : M \times L \to N$.

This also works the other way around: For $g : M \times L \to N$, we can define the function $\mathbf{c}(g) : L \to N^M$, mapping an l to the function $g(l, -) : M \to N$. We call the assignments

$$\hom_{\mathbf{SET}}(M \times L, N) \underset{\mathbf{uc}}{\overset{\mathbf{c}}{\rightleftarrows}} \hom_{\mathbf{SET}}(L, N^M)$$

currying and uncurrying (named after the mathematician and computer scientist Haskell Curry).

Remark 3.6. Currying is also used in Lean and other functional programming languages to define product types.

Now the bijection above is natural in L and N, meaning that $(-)^M$ is a right adjoint functor of the product functor $M \times -$. Therefore, we can generalize the concept of N^M into category theory as follows:

Definition 3.7. Assume that $\mathcal{C}$ has all binary products, and let X, Y be objects of $\mathcal{C}$.
If the functor $X \times -$ has a right adjoint $(-)^X$, we call X **exponentiable** and Y^X an **exponential object**.
We call $\mathcal{C}$ **Cartesian closed** if it has a terminal object and all of its objects are exponentiable.
Furthermore, let **c** and **uc** be the assignments such that

$$\hom_{\mathcal{C}}(X \times Y, Z) \underset{\mathbf{uc}}{\overset{\mathbf{c}}{\rightleftarrows}} \hom_{\mathcal{C}}(Y, Z^X) \tag{3.3}$$

forms the natural transformation corresponding to the adjunction $(X \times -) \dashv (-)^X$.

Remark 3.8. We require a Cartesian closed category (CCC) $\mathcal{C}$ to have a terminal object, because this automatically yields the monoidal category $(\mathcal{C}, \top, \times)$. For an arbitrary monoidal category $(\mathcal{D}, 1, \otimes)$, monoidal closedness is defined as the existence of an adjunction $(X \otimes -) \dashv [-, X]$ for every X in $\mathcal{D}$. Thus, Cartesian closedness is just a special instance of monoidal closedness. This is how Cartesian closedness is defined in Mathlib.

3.4 Basic consequences of the definition

Now we go over some basic but very useful constructions in and statements about topoi. For this, let $\mathcal{E}$ be an arbitrary topos.

Recall that in definition 3.2, we assume the existence of a terminal object. Thus the definition of a topos also implicitly assumes the existence of a terminal object. Any category with all pullbacks and a terminal object already has all finite limits. Thus $\mathcal{E}$ has all finite limits.

We now define some simple but useful morphisms in a topos.

Definition 3.9. Let X be an object of $\mathcal{E}$.
We define its **equality predicate** $\delta_X : X \times X \to \Omega$ as the characteristic morphism of the usual diagonal $\Delta_X = \langle \mathrm{id}_X, \mathrm{id}_X \rangle : X \rightarrowtail X \times X$.
We define its **singleton morphism** $\{\cdot\}_X : X \to \mathcal{P}X$ as the transpose of δ_X.
We define the monomorphism u_X to be the pullback $\in_X{}^*\mathbf{true}$.
We define $\mathbf{true}_X : X \to \Omega$ to be the composition $!_X \gg \mathbf{true}$, where $!_X$ is the unique morphism into the terminal object.

Although we do not know what the objects of $\mathcal{E}$ are, the existence of the membership predicate hints at their "set-like" behavior. Therefore, it is often useful to think of objects of $\mathcal{E}$ as containing elements.

In the above definition, δ_X can be interpreted as the equality predicate stating $x = y$ for "elements" x, y of X.

The singleton morphism $\{\cdot\}_X$ is analogous to the function, which sends x to the subobject of X, which contains only x. In **SET**, this is obviously injective; analogously, it is also easy to show that this is a monomorphism in $\mathcal{E}$. [MM92, cf. p. 166, Lem. 1] The resulting characteristic morphism $\sigma_X := \chi_X : \mathcal{P}X \to \Omega$ can be interpreted as the predicate "S is a singleton".

The monomorphism u_X represents the membership relation as a subobject of $X \times \mathcal{P}X$. The predicate $\mathbf{true}_X$ can be thought of as the predicate sending any element of X to "True". It characterizes the monomorphism id_X, so X interpreted as a subobject of itself.

Another aspect, which makes a topos behave like the **SET** is exhibited by the following proposition:

Proposition 3.10. *In $\mathcal{E}$, all monomorphisms are regular, i.e., are the equalizer monomorphism for some parallel pair of morphisms. Furthermore $\mathcal{E}$ is balanced, meaning any monomorphism that is also an epimorphism is an isomorphism.*

Proof. Definition 3.2 implies that a monomorphism $m : S \rightarrowtail X$ in $\mathcal{E}$ is the equalizer of χ_m and $\mathbf{true}_X$.
Furthermore, if m is also an epimorphism, then $m \gg \mathbf{true}_X = m \gg \chi_m$ implies $\mathbf{true}_X = \chi_m$. Now equalizers of equal morphisms are isomorphisms. □

Next, provide equivalent definitions of a topos. Firstly, consider the following theorem. It essentially states that power objects are just a special case of exponential objects.

Theorem 3.11. *In a category $\mathcal{C}$ with a subobject classifier Ω, an exponentiable object X already has a power object, given by $\mathcal{P}X = \Omega^X$ and $\in_X = \boldsymbol{uc}(id_{\Omega^X})$.*

Proof. Let X be an exponentiable object of $\mathcal{C}$ and $f : X \times Y \to \Omega$ a morphism. We want to use $\mathbf{c}(h)$ as its transpose. By the naturality of (3.3), we have:

$$(\mathrm{id}_X \times \mathbf{c}(h)) \gg \mathbf{uc}(\mathrm{id}_{\Omega^X}) = \mathbf{uc}(\mathbf{c}(h) \gg \mathrm{id}_{\Omega^X}) = h$$

Furthermore, if $h = (\mathrm{id}_X \times k) \gg \mathbf{uc}(\mathrm{id}_{\Omega^X})$ for some $k : Y \to \Omega^X$, then also by the naturality of (3.3) we get:

$$h = \mathbf{uc}(k \gg \mathrm{id}_{\Omega^X}) = \mathbf{uc}(k)$$

Therefore $\mathbf{c}(h) = k$ holds, proving the uniqueness of the transpose. □

This implies that we could also have defined a topos as a Cartesian closed category with equalizers and a subobject classifier (Note, that the existence of all equalizers, binary products, and a terminal object already implies the existence of all finite limits).

We can also go the other way around:

Theorem 3.12 ([MM92, p. 167, Thm. 1]). *A finitely complete category $\mathcal{C}$ with a subobject classifier Ω and power objects is Cartesian closed.*

We will do not prove this theorem here. However, the core argument is to define the subobjects of $\mathcal{P}(X \times Y)$ acting like the subset of all graphs of functions $X \to Y$.

Thus, we can also define a topos as a category with subobject classifiers, power objects, and all pullbacks.

4 Finite Colimits in a Topos

Originally, the definition of topoi also included the existence of finite colimits (this property is also known as finite cocompleteness). However, it quickly became apparent that this definition was redundant. [LMW75, p. 120] In this chapter, we show that the current topos axioms already imply the finite cocompleteness of a topos. For this, we first define the direct image of a monomorphism. Lastly, we use the existence of finite colimits to show that in a topos all morphisms can be written as a unique (up to unique isomorphism) monomorphism, precomposed with an epimorphism.

4.1 The direct image morphism

In this section, let $k : A \rightarrowtail B$ be a monomorphism in a topos $\mathcal{E}$.

We want to define the direct image map $\exists_k : \mathcal{P}A \to \mathcal{P}B$, which in **SET** corresponds to the usual map $S \mapsto k(S)$ on subsets $S \subseteq A$. In categorical language this means that $\exists_k$ should make the following diagram commute:

$$\begin{array}{ccc} A \times \mathcal{P}A & & \\ {\scriptstyle k\times\exists_k}\downarrow & \searrow^{\in_A} & \\ B \times \mathcal{P}B & \xrightarrow{\in_B} & \Omega \end{array}$$

Writing $k \times \exists_k$ as $(k \times 1) \gg (1 \times \exists_k)$, we can expand this diagram to the following:

$$\begin{array}{ccc} U_A & \longrightarrow & \top \\ {\scriptstyle u_A}\downarrow & & \downarrow{\scriptstyle \mathbf{true}} \\ A \times \mathcal{P}A & \xrightarrow{\in_A} & \Omega \\ {\scriptstyle k\times \mathrm{id}_{\mathcal{P}A}}\downarrow & & \| \\ B \times \mathcal{P}A & \overset{\mathbf{e}_k}{\dashrightarrow} & \Omega \\ {\scriptstyle \mathrm{id}_B\times\exists_k}\downarrow & & \| \\ B \times \mathcal{P}B & \xrightarrow{\in_B} & \Omega \end{array} \tag{4.1}$$

The upper square is the defining pullback of u_A, and $\mathbf{e}_k$ is the characteristic map of $u_A \gg (k \times \mathrm{id}_{\mathcal{P}A})$. Note that this map is only a monomorphism, because k is a monomorphism. This shows why we require that assumption for this definition. Now the lower square is the defining diagram for the transpose of $\mathbf{e}_k$, and therefore we can define $\exists_k$ as such:

Definition 4.1. The **direct image** $\exists_k$ of a monomorphism $k : A \rightarrowtail B$ is defined as the transpose of the map $\mathbf{e}_k := \chi_{u_A \gg (k \times \mathrm{id}_{\mathcal{P}A})}$.

To further convince ourselves that this definition is the one we want, we prove the following lemma.

Lemma 4.2 ([MM92, Prop 3.1])**.** *Monomorphisms $m : A \rightarrowtail B$ and $k : B \rightarrowtail C$ satisfy the equality*

$$\ulcorner m \urcorner \gg \exists_k = \ulcorner m \gg k \urcorner.$$

Theorem 4.3 ([MM92, Prop IV.3.2])**.** *Consider the following diagrams:*

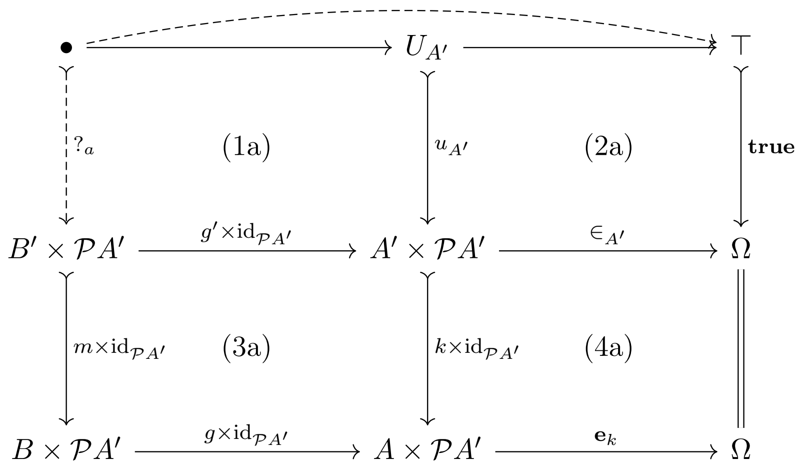
$$\begin{array}{ccc} B' & \xrightarrow{g'} & A' \\ {\scriptstyle m}\big\downarrow & (i) & \big\downarrow{\scriptstyle k} \\ B & \xrightarrow{g} & A \end{array} \qquad\qquad \begin{array}{ccc} \mathcal{P}A' & \xrightarrow{\mathcal{P}g'} & \mathcal{P}B' \\ {\scriptstyle \exists_k}\big\downarrow & (ii) & \big\downarrow{\scriptstyle \exists_m} \\ \mathcal{P}A & \xrightarrow{\mathcal{P}g} & \mathcal{P}B \end{array}$$

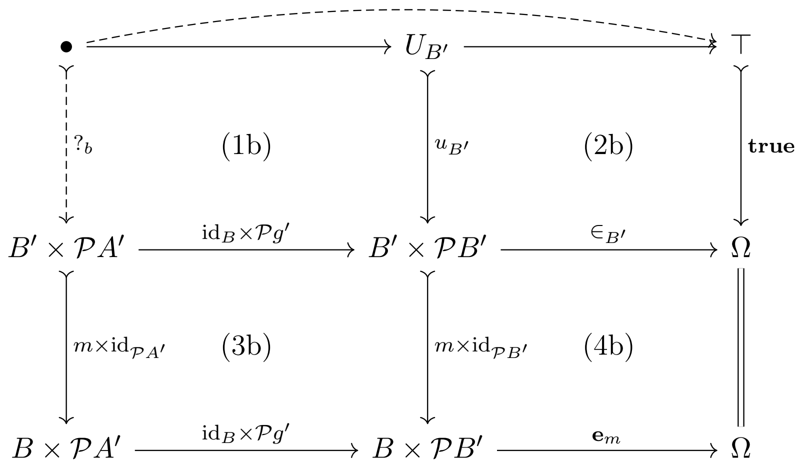
If k is monic and (i) is a pullback (and thus m is also monic), then (ii) commutes.

Proof. We prove the equality of $Pg' \gg \exists_m$ and $\exists_k \gg Pg$ by showing that their inverse transposes are equal. It is easy to confirm that:

$$\widetilde{\exists_k \gg \mathcal{P}g} = (g \times \mathrm{id}_{\mathcal{P}A'}) \gg \mathbf{e}_k$$
$$\widetilde{\mathcal{P}g' \gg \exists_m} = (\mathrm{id}_B \times \mathcal{P}g') \gg \mathbf{e}_m$$

Now we want to use the uniqueness of the characteristic morphism. To that end, consider the following two diagrams:





Here, let $?_a$ and $?_b$ be the pullback of **true** along $(g' \times \mathrm{id}_{\mathcal{P}A'}) \gg\in_{A'}$ and $(\mathrm{id}_B \times Pg') \gg\in_{B'}$ respectively. Thus the (dashed) upper halves of both diagrams are pullbacks by definition.

By definition, 3.9 (2a) and (2b) are pullbacks. Thus, by pullback pasting and the uniqueness of the morphism into the terminal object, we can find suitable morphisms into $U_{A'}$ and $U_{B'}$, such that (1a) and (1b) are also pullbacks.

The square (3b) is a pullback by assumption (and the fact that products of pullbacks are again pullbacks), and it is easy to see that (3b) is also a pullback. Thus, by pullback pasting, the combined squares (1a)+(3a) and (1b)+(3b) are pullbacks. The combined squares (2a)+(4a) (resp. (2b)+(4b)) are also pullbacks by the definition of $\exists_k$ (resp. $\exists_m$). Therefore, again by the pasting law, both outer squares are pullbacks. Therefore, $?_a \gg (m \times \mathrm{id}_{\mathcal{P}A'})$

and $?_b \gg (m \times \mathrm{id}_{\mathcal{P}A'})$ represent the subobjects classified by $(g \times \mathrm{id}_{\mathcal{P}A'}) \gg \mathbf{e}_k$ and $(\mathrm{id}_B \times \mathcal{P}g') \gg \mathbf{e}_m$ respectively.

Now by 3.2 of $\mathcal{P}$ we know that

$$(g' \times \mathrm{id}_{\mathcal{P}A'}) \gg\in_{A'} = (\mathrm{id}_B \times Pg') \gg\in_{B'} .$$

Thus we can choose $?_a$ and $?_b$ to be the same, and so $(g \times \mathrm{id}_{\mathcal{P}A'}) \gg \mathbf{e}_k$ and $(\mathrm{id}_B \times Pg') \gg \mathbf{e}_m$ classify the same subobject, as was to be shown. □

Remark 4.4. While this proof looks exactly like the proof of [MM92, Prop 3.2], there is a slight nuance. In [MM92], the morphisms $?_a$ and $?_b$ were defined as the pullbacks of $u_{A'}$ and $u_{B'}$ along $g' \times \mathrm{id}_{\mathcal{P}A'}$ and $\mathrm{id}_B \times \mathcal{P}g'$ respectively. Our method has the advantage that because of the equality of the compositions through the middle of the diagram, we can define $?_a$ and $?_b$ to be equal. This is especially useful in the Lean formalization.

We will use the following corollary without proof.

Corollary 4.5 ([MM92, Cor. IV.3.3])**.** *For a monomorphism $k : B' \rightarrowtail B$ in $\mathcal{C}$, we know that $\exists_k \gg \mathcal{P}k = id_{\mathcal{P}B'}$.*

4.2 A topos has all Finite Colimits

The basic idea behind the proof of theorem 1.1 is to show that the functor $\mathcal{P}-$ creates limits.

A functor creates limits if any diagram whose images under the functor has a limit cone also has a limit cone.

Now, $\mathcal{E}$ has all limits of the shape $\mathcal{J}^{\mathbf{op}}$, where $\mathcal{J}$ is a finite indexing category. If $\mathcal{P}- : \mathcal{E}^{\mathbf{op}} \Rightarrow \mathcal{E}$ creates limits, this means $\mathcal{E}^{\mathbf{op}}$ has all limits of shape $\mathcal{J}^{\mathbf{op}}$. But that just means that $\mathcal{E}$ has all colimits of shape $\mathcal{J}$.

The following lemma gives four conditions that thogether imply the creation of limits for $\mathcal{P}-$:

Proposition 4.6. *The functor $\mathcal{P}- : \mathcal{E}^{\mathbf{op}} \Rightarrow \mathcal{E}$ creates limits if*

(i) $\mathcal{E}^{\mathbf{op}}$ has coequalizers of all reflexive pairs;

(ii) $\mathcal{P}-$ has a left adjoint;

(iii) $\mathcal{P}-$ reflects isomorphisms;

(iv) $\mathcal{P}-$ preserves coequalizers of reflexive pairs.

Remark 4.7. This proposition is essentially a combination of [MM92, Prop. IV.4.1], which states that monadic functors create limits, and [MM92, Cor. IV.4.3], which established the above conditions as conditions for a functor's monadicity. To fully understand these theorems, we would need to take a detour into the theory of monads on categories and their relation to adjuntions. Because the two aforementioned results are already formalized in Mathlib, we will not bother with going into detail about them.

Reflexive pairs in a category $\mathcal{C}$ are pairs of morphisms $f, g : X \to Y$, such that there exists an $s : Y \to X$ with $s \gg f = \mathrm{id}_Y = s \gg g$. Note that this immediately implies that f and g are epimorphisms.
A functor $F : \mathcal{C} \Rightarrow \mathcal{D}$ is said to reflect isomorphisms if every morphism i in $\mathcal{C}$ for which Fi is an isomorphism is also an isomorphism.
A functor $F : \mathcal{C} \Rightarrow \mathcal{D}$ preserves coequalizers if and only if the image of every coequalizer diagram in $\mathcal{C}$ is again a coequalizer diagram in $\mathcal{D}$.

The fact that $\mathcal{E}^{\mathbf{op}}$ has all coequalizers of reflexive pairs immediately follows from the fact that $\mathcal{E}$ has all finite limits, in particular all equalizers.

Now we go on to prove that $\mathcal{P}-$ fulfills condition (iv) in proposition 4.6. For this, we first need the following lemma.

Lemma 4.8. *Consider the following diagrams in an arbitrary category $\mathcal{C}$.*

$$A \underset{g}{\overset{f}{\rightrightarrows}} B \xrightarrow{\;p\;} P \qquad\qquad \begin{array}{ccc} A & \xrightarrow{f} & B \\ {\scriptstyle g}\downarrow & & \downarrow{\scriptstyle p} \\ B & \xrightarrow{p} & P \end{array}$$

If the left one describes a coequalizer of a reflexive pair, then the right square is a pushout.

Proof. Let $s : B \to A$ be the morphism reflecting f and g. Now for every pair of morphisms $h, h' : B \to Z$ with $f \gg h = g \gg h'$ we know that

$$h = s \gg f \gg h = s \gg g \gg h' = h'.$$

Thus, h coequalizes f and g. By the universal property of the coequalizer, there exists a unique morphism $d : P \to Z$, such that $h' = h = p \gg d$. But this is nothing more than the universal pullback property for the right square. ☐

Lemma 4.9. *The functor $\mathcal{P}- : \mathcal{E}^{\mathbf{op}} \Rightarrow \mathcal{E}$ preserves coequalizers of reflexive pair.*

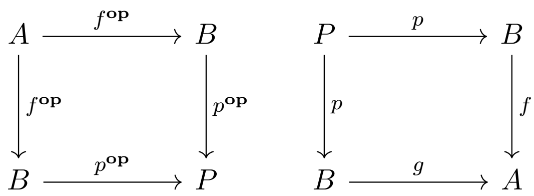
Proof. Consider a coequalizer of a reflexive pair in $\mathcal{E}^{\mathbf{op}}$:

$$A \underset{g^{\mathbf{op}}}{\overset{f^{\mathbf{op}}}{\rightrightarrows}} B \xrightarrow{\;p^{\mathbf{op}}\;} P$$

We need to show that

$$\mathcal{P}A \underset{\mathcal{P}g}{\overset{\mathcal{P}f}{\rightrightarrows}} \mathcal{P}B \xrightarrow{\;\mathcal{P}p\;} \mathcal{P}P \tag{4.2}$$

is a coequalizer. By lemma 4.8, the following left square is a pushout in $\mathcal{E}^{\mathbf{op}}$:



Now pushouts in the opposite category are simply pullbacks in the original category. Hence, the right square is a pullback in $\mathcal{E}$. Recall that $f^{\mathbf{op}}$ and $g^{\mathbf{op}}$ are both epimorphisms, and so f and g are monomorphisms. Now Theorem 4.3 and Corollary 4.5 give us the equations:

$$\exists_f \gg \mathcal{P}g = \mathcal{P}p \gg \exists_p, \quad \exists_p \gg \mathcal{P}p = \mathrm{id}_{\mathcal{P}P} \quad \text{and} \quad \exists_f \gg \mathcal{P}f = \mathrm{id}_B.$$

Additionally, the functoriality of $\mathcal{P}-$ implies

$$\mathcal{P}f \gg \mathcal{P}p = \mathcal{P}g \gg \mathcal{P}p.$$

These four properties turn (4.2) into what is commonly called a "split fork" [Mac71, cf. p. 145]. By well-known fact, split forks are coequalizers, which concludes our proof. □

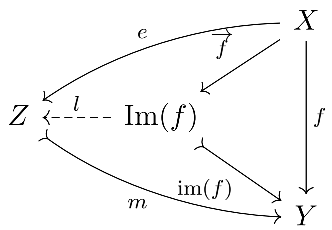
Remark 4.10. Because Mathlib already contains the notion of split forks and the fact that they are coequalizers, in the Lean formalization it also suffices to prove that (4.2) is one.

4.3 Image factorizations

Next, we show how every morphism in a topos factors into an epimorphism followed by a monomorphism.

Definition 4.11. Let $f : X \to Y$ be a morphism in $\mathcal{E}$.

An **image factorization** of f is a factorization $f = \overrightarrow{f} \gg \mathrm{im}(f)$, with $\mathrm{im}(f)$ a monomorphism, such that for all other factorizations $f = e \gg m$ where m is a monomorphism, there exists a unique morphism l making the following diagram commute.



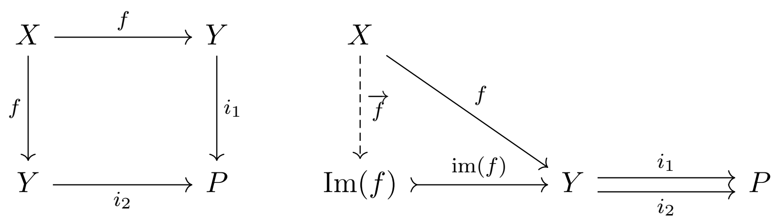
We call $\mathrm{im}(f)$ the image monomorphism or simply the **image** of f.

Remark 4.12. Because $\mathrm{im}(f)$ is a monomorphism, it immediately follows that l is a monomorphism as well.

We will now construct such a factorization for morphisms in $\mathcal{E}$.

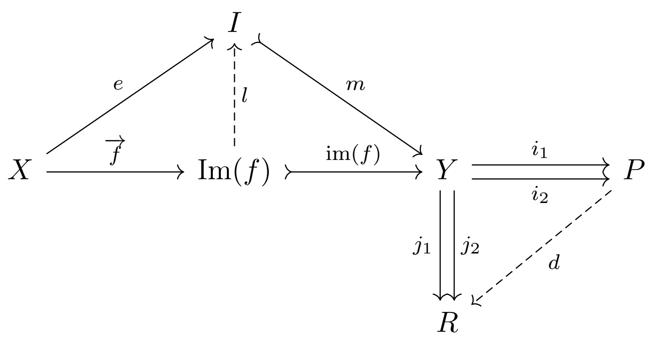
Theorem 4.13. *Every morphism $f : X \to Y$ in $\mathcal{E}$ has an image factorization.*

Proof. Let $i_1, i_2 : Y \to P$ be morphisms such that the left square below is a pushout. Such morphisms exist because we now know that a topos has all pushouts.



Now we define our image $\mathrm{im}(f)$ to be the equalizer monomorphism of i_1 and i_2 as seen above. Obviously f equalizes i_1 and i_2. Then by the universal property of equalizers, there exists a unique morphism $\overrightarrow{f}$ making the triangle commute.

Now we need to prove that this factorization is in fact an image factorization. Towards this end, let $e \gg m = f$ be an arbitrary factorization, with m a monomorphism. Because in a topos, all monomorphisms are regular (by Prom 3.10), there exist two morphisms j_1 and j_2 as seen below, such that m is their equalizer.



Therefore we know that $f \gg j_1 = f \gg j_2$, and so by the universal property of pushouts, there exists a d making the lower right triangle commute.

This now implies that $\mathrm{im}(f) \gg j_1 = \mathrm{im}(f) \gg j_2$. Now by the universal property of equalizers for m, there exists a unique morphism l making the upper right triangle commute.

Finally, the upper left triangle simply commutes because m is a monomorphism. $\square$

Although we have now proved that $\mathcal{E}$ has all images, we have not yet shown that $\overrightarrow{f}$ is an epimorphism. However, it is known any image factorization in a category with all equalizers consists of an epimorphism and a monomorphism. [Mit65, p.12] This result also exists in Mathlib and was used in the accompanying Lean formalization.

Now for any other factorization $e \gg m = f$, where $m : I \rightarrowtail X$ is a monomorphism and $e : I \twoheadrightarrow Y$ an epimorphism, we know that $\overrightarrow{f} \gg l = e$, with l as in definition 4.11. Therefore, l is an epimorphism. By remark 4.12, it is also an epimorphism. Thus, by the balancedness of $\mathcal{E}$, it is an isomorphism. Essentially we have just shown that any factorization consisting of an epimorphism and isomorphism is the image factorization (up to unique isomorphism).

Remark 4.14. Notice that in this chapter the only three properties of $\mathcal{E}$ we used, namely the existence of equalizers, the existence of pushouts, and the fact that all monomorphisms in $\mathcal{E}$ are regular (recall that in Proposition 3.10 this property already implies the balancedness of $\mathcal{E}$). We can therefore perform these constructions in any category with these three properties.

5 Constructing the Internal Connectives

In this chapter, let $\mathcal{E}$ be a topos and ϕ, $\psi : X \to \Omega$ be predicates on some object X of $\mathcal{E}$.

Now, we use almost all the structure which has so far been shown to exist in a topos to define the logical connectives $\wedge, \vee, \Rightarrow$ and $\neg$ as morphisms

$$\wedge_\Omega, \vee_\Omega, \Rightarrow_\Omega \colon \Omega \times \Omega \longrightarrow \Omega,$$

and $\neg_\Omega : \Omega \to \Omega$, closely following [Kos11, p. 75]. These morphisms can then be used to construct predicates of the form $\phi \Box \psi$ for $\Box \in \{\wedge, \vee, \Rightarrow\}$ and $\neg\phi$ as the compositions

$$X \xrightarrow{\langle \phi, \psi \rangle} \Omega \times \Omega \xrightarrow{\Box_\Omega} \Omega$$

$$X \xrightarrow{\phi} \Omega \xrightarrow{\neg_\Omega} \Omega.$$

We then show that we have defined the connectives correctly. By that, we mean that the subobjects classified by $\psi \Box \phi$ are equal to the corresponding connective of the subobjects ψ and ϕ in the partial order $\mathrm{Sub}_{\mathcal{E}}(X)$. We then demonstrate how we can use these connectives to define analogues to the usual set-theoretic operations on power objects $\mathcal{P}X$.

5.1 Defining the connectives

The idea behind the following definitions of $\wedge_\Omega$, $\vee_\Omega$, and $\Rightarrow_\Omega$ is fairly straightforward: We can think of "elements of" $\Omega \times \Omega$ as pairs (p, q) of truth values. For some connective $\Box$, we want to find a monomorphism that points out the relation on $\Omega \times \Omega$ containing precisely those pairs (p, q), such that $p \Box q$ is true with $\Box \in \{\wedge, \vee, \Rightarrow\}$. The desired morphism $\Omega \times \Omega \to \Omega$ will then just be the corresponding characteristic morphism.

However, we have to keep in mind that because we are working intuitionistically, our subobject classifier does not generally contain just the two values "true" and "false". Thus, to construct these subobject monomorphisms into $\Omega \times \Omega$, we cannot use the familiar Boolean truth tables, as might be our first guess.

Using the abovementioned intuition, the morphism $\wedge_\Omega$ is easiest to define. The predicate $\phi \wedge \psi$ should map elements of X to **true**, if and only if both ϕ and ψ map it there. Note that this is precisely the case if $\langle \phi, \psi \rangle$ maps the element onto $s_\wedge := (\mathbf{true}, \mathbf{true})$, which is pointed out by the monomorphism

$$\top \times \top \overset{\mathbf{true} \times \mathbf{true}}{\rightarrowtail} \Omega \times \Omega.$$

Thus, $\wedge_\Omega$ should simply be its characteristic map.

Definition 5.1. We define the connective $\wedge_\Omega : \Omega \times \Omega \to \Omega$ to be $\chi_{s_\wedge}$.

Defining $\vee_\Omega$ is slightly less trivial. We want the predicate $\phi \vee \psi$ to map an element x of X onto **true** if and only if one of the two predicates maps it there. If ϕ already maps x onto true, this means that $\langle \phi, \psi \rangle$ maps it onto any pair $(\mathbf{true}, q)$ of truth values. These pairs are pointed out by the monomorphism

$$\top \times \Omega \xrightarrow{\mathbf{true} \times \mathrm{id}_\Omega} \Omega \times \Omega.$$

Analogously, $\mathrm{id}_\Omega \times \mathbf{true}$ points out all pairs $(p, \mathbf{true})$.

Therefore, we would like $\vee_\Omega$ to classify the morphism

$$\top \times \Omega + \Omega \times \top \xrightarrow{[\mathbf{true} \times \mathrm{id}_\Omega, \mathrm{id}_\Omega \times \mathbf{true}]} \Omega \times \Omega.$$

However, in all nontrivial topoi, this is clearly not a monomorphism. Therefore we have to define $\vee_\Omega$ as the characteristic morphism of its image $\mathrm{im}([\mathbf{true} \times \mathrm{id}_\Omega, \mathrm{id}_\Omega \times \mathbf{true}]) =: s_\vee$.

Definition 5.2. We define the connective $\vee_\Omega : \Omega \times \Omega \to \Omega$ to be $\chi_{s_\vee}$.

The key to defining $\Rightarrow_\Omega$ is the following observation: For propositions P and Q

$$P \Rightarrow Q, \text{ if and only if } (P \wedge Q) = P.$$

Therefore, the equalizer

$$\leq \xrightarrow{s_\Rightarrow} \Omega \times \Omega \underset{\pi_1}{\overset{\wedge_\Omega}{\rightrightarrows}} \Omega$$

should point out the correct subobject. Note that equalizer morphisms are always monomorphisms, and so $s_\Rightarrow$ is also one.

Definition 5.3. We define the connective $\Rightarrow_\Omega : \Omega \times \Omega \to \Omega$ to be $\chi_{s_\Rightarrow}$.

For the next construction, we need to assume that the morphism $!_\bot : \bot \to \top$, where $\bot$ is the initial object in $\mathcal{E}$, is a monomorphism This follows from Theorem 1.2, which we show later.

Definition 5.4. Let **false** denote the characteristic morphism $\chi_{!_\bot} : \top \to \Omega$ of the unique map $!_{bot} : \bot \to \top$.
We define $\neg_\Omega : \Omega \to \Omega$ to be the characteristic map $\chi_{\mathbf{false}}$.

5.2 $\wedge_\Omega$ is correct

Next, we confirm that our definition of $\wedge_\Omega$ is correct in the sense described above. To that end, we need to prove that in the partially ordered set $\mathrm{Sub}_\mathcal{E}(X)$, the subobject classified by $\phi \wedge \psi$ is the meet of the subobjects classified by ϕ and ψ. For this, we use the following proposition from [MM92] :

Proposition 5.5 ([MM92, Prop. IV.6.3])**.** *For any object X in a topos $\mathcal{E}$, the partially ordered set $Sub_{\mathcal{E}}(X)$ is a lattice with the meet*

$$[s] \cap [t] := [s^*t \gg s]$$

and the join

$$[s] \cup [t] := [im([s,t])],$$

where $[s,t] : S + T \to X$ is the usual map induced by the universal property of the coproduct.

Now we can easily confirm that we made the right choice for $\wedge_\Omega$.

Theorem 5.6. *Let $s : S \hookrightarrow X$ and $t : T \hookrightarrow X$ be monomorphisms in a topos $\mathcal{C}$. Then:*

$$\chi_{[s]\cap[t]} = \chi_s \wedge \chi_t$$

Proof. By Proposition 5.5, we know that $[s] \cap [t] = [s^*t \gg s]$. Now by the uniqueness of the characteristic morphism, it suffices to prove that the outer square in the following diagram is a pullback.

$$\begin{array}{ccccccc}
S \times_X T & \xrightarrow{s^*t \times t^*s} & S \times T & \longrightarrow & \top \times \top & \longrightarrow & \top \\
\big\downarrow{\scriptstyle s^*t \gg s} & & \big\downarrow{\scriptstyle s \times t} & & \big\downarrow{\scriptstyle \mathbf{true} \times \mathbf{true}} & & \big\downarrow{\scriptstyle \mathbf{true}} \\
X & \xrightarrow{\Delta_X} & X \times X & \xrightarrow{\chi_s \times \chi_t} & \Omega \times \Omega & \xrightarrow{\wedge_\Omega} & \Omega
\end{array}$$

The right square is a pullback by the definition of the characteristic map. The middle square is the product of two pullback squares, so it is also a pullback. The left square is a pullback by Lemma 2.1. By twofold application of pullback pasting, the outer square is in fact a pullback. □

Remark 5.7. Because $\top \times \top \cong \top$, we could also have defined $\wedge_\Omega$ as the characteristic morphisms of $\langle \mathbf{true}, \mathbf{true} \rangle$, which is how it was done in [Kos11]. However, as is apparent in the proof above, the definition at hand is more useful. This is especially true when considering that in Lean, the composition with an extra isomorphism in the above diagram would bring with it an additional challenge.

The proof of the correctness of $\vee_\Omega$ requires the fact that pullbacks preserve colimits. Thus we skip it and refer to [Gol14, Theorem 7.1.3].

5.3 $\Rightarrow_\Omega$ and $\neg_\Omega$ are correct

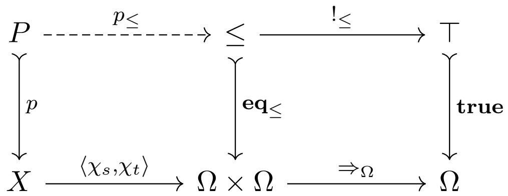
The proof of the correctness of $\Rightarrow_\Omega$ works differently from the previous one. Instead of outright constructing the subobject corresponding to the Heyting implication in $\mathrm{Sub}_{\mathcal{E}}(X)$ and then proving an equality, we prove that the subobject classified by $\chi_s \Rightarrow \chi_t$ satisfies the property of the Heyting implication from Definition 2.3.

Theorem 5.8. *Let $s : S \rightarrowtail X$, $t : T \rightarrowtail X$ be monomorphisms in $\mathcal{E}$. Then in $Sub_{\mathcal{E}}(X)$, there exists a Heyting implication $[s] \subseteq [t]$, classified by $\chi_s \Rightarrow \chi_t$.*

Before we start this theorem's proof, we need to state a small lemma. It follows immediately from the pullback pasting law and the uniqueness of the characteristic momorphism.

Lemma 5.9. *Let $s : S \rightarrowtail X$ be a monomorphism and $h : X' \to X$ a morphism in $\mathcal{E}$. Then $\chi_{h^*s} = h \gg \chi_s$.*

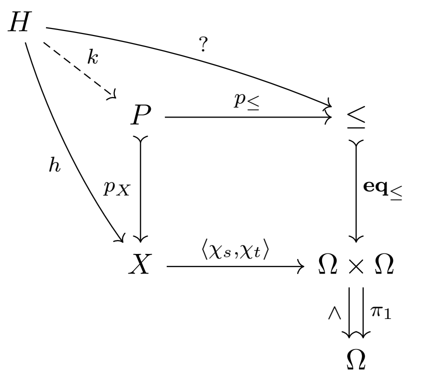
Proof of theorem 5.8. Let $p : P \to X$ be the pullback of **true** along $\langle \chi_s, \chi_t \rangle \gg \Rightarrow_\Omega$. In the diagram below, the right square is by definition a pullback, and so by pullback pasting, there exists a $p_\leq$ making the left square a pullback.



We define $[s] \subseteq [t] := [p]$. Now by definition 2.3 it suffices to prove that for all monomorphisms $h : H \hookrightarrow X$:

$$[h] \leq [p], \text{ if and only if } [s] \cap [h] \leq [t]$$

We know that $[h] \leq [p]$ just means that there exists some $k : H \to P$ such that $h = k \gg p$. Now if such a k exists, then the following diagram (with $? = k \gg p_\leq$) commutes, implying that $h \gg \langle \chi_s, \chi_t \rangle$ factors through $\mathbf{eq}_\leq$.



On the other hand, if $h \gg \langle \chi_s, \chi_t \rangle$ factors through $\mathbf{eq}_\leq$ with some ?, then by the universal property of a pullback, there exists a (unique) k such that the diagram commutes. Therefore such a k exists if and only if $h \gg \langle \chi_s, \chi_t \rangle$ factors through $\mathbf{eq}_\leq$. By the universal property of the equalizer, this is the case if and only if

$$h \gg (\chi_s \wedge \chi_t) = h \gg \langle \chi_s, \chi_t \rangle \gg \wedge_\Omega = h \gg \langle \chi_s, \chi_t \rangle \gg \pi_1 = h \gg \chi_s$$

Now by theorem 5.6, lemma 5.9, and the uniqueness of the characteristic morphism, this is equivalent to the equality $h^*(s^*t \gg s) = h^*s$. Using that $\cap$ is the meet operator in $\mathrm{Sub}_{\mathcal{E}}(X)$, this implies the equality

$$([s] \cap [h]) \cap [t] = [h] \cap ([s] \cap [t]) = [h^*(s^*t \gg s) \gg h]$$
$$= [h^*s \gg h] = [h] \cap [s] = [s] \cap [h]$$

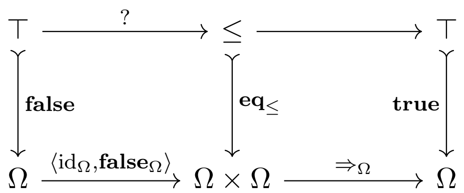
The opposite direction holds, because h is a monomorphism. Finally, by 2.4, this equality holds, if and only if $[s] \cap [h] \leq [t]$, which was to be shown. $\square$

Finally we prove that $\neg_\Omega$ is the correct morphism describing pseudonegation. Recall from Section 2.3 that the pseudocomplement of a subobject $[s]$ in $\mathrm{Sub}_{\mathcal{E}}(X)$ is just $[s] \subseteq [0_X]$. Thus, we want to show that $\neg\chi_s$ characterizes $[s] \subseteq [0_X]$. The following theorem is a more general form of this statement. The case in $\mathrm{Sub}_{\mathcal{E}}(X)$ follows as a corollary.

Theorem 5.10. *The following identity holds:*

$$\langle id_\Omega, \boldsymbol{false}_\Omega \rangle \gg \Rightarrow_\Omega = \neg_\Omega.$$

Proof. Consider the following diagram:



The right square is just a pullback by the definition of $\Rightarrow_\Omega$. By the uniqueness of the characteristic morphism of **false** and the pullback pasting lemma, it suffices to find a morphism ?, such that the left square is a pullback. From the universal property of the equalizer, we know that there exists a morphism ? making the left square commute if and only if the morphism

$$\mathbf{false} \gg \langle \mathrm{id}_\Omega, \mathbf{false}_\Omega \rangle = \langle \mathbf{false}, \mathbf{false} \rangle$$

equalizes $\wedge_\Omega$ and π_1. By theorem 5.6, we know that

$$\mathbf{false} \wedge_\Omega \mathbf{false} = \chi_{0_\top} \wedge \chi_{0_\top} = \chi_{[0_\top] \cap [0_\top]} = \chi_{0_\top} = \mathbf{false}$$

and so the desired ? exists. Towards proving the universal pullback property of the right square, let $z_1 : Z \to \leq$ and $z_2 : Z \to \Omega$ be morphisms with $z_1 \gg \mathbf{eq}_\leq = z_2 \gg \langle \mathrm{id}_\Omega, \mathbf{false}_\Omega \rangle$. Because $!_Z$ is unique, it suffices to prove that

$$z_1 = !_Z \gg ? \text{ and } z_2 = !_Z \gg \mathbf{false} = \mathbf{false}_Z.$$

Now because $z_2 \gg \langle \mathrm{id}_\Omega, \mathbf{false}_\Omega \rangle$ factors through $\mathbf{eq}_\leq$ and we know that

$$z_2 = z_2 \gg (\mathrm{id}_\Omega \wedge_\Omega (\mathbf{false}_\Omega)) = z_2 \gg \mathbf{false}_\Omega = \mathbf{false}_Z.$$

Finally, see that

$$\begin{aligned} z_1 \gg \mathbf{eq}_\leq &= z_2 \gg \langle \mathrm{id}_\Omega, \mathbf{false}_\Omega \rangle \\ &= !_Z \gg \mathbf{false} \gg \langle \mathrm{id}_\Omega, \mathbf{false}_\Omega \rangle = !_Z \gg ? \gg \mathbf{eq}_\leq \end{aligned}$$

and since $\mathbf{eq}_\leq$ is a monomorphism, $z_1 = !_? \gg ?$ and we are finished. □

Corollary 5.11. *For any monomorphism $s : S \rightarrowtail X$ in $\mathcal{E}$ we have*

$$\chi_{[s]\subseteq[0_X]} = \chi_s \Rightarrow \boldsymbol{\mathit{false}}_X = \neg\chi_s.$$

Proof. Observing that $\chi_s \Rightarrow \mathbf{false}_X$ is equal to the composition

$$X \xrightarrow{\quad \chi_s \quad} \Omega \xrightarrow{\langle \mathrm{id}_\Omega, \mathbf{false}_\Omega \rangle} \Omega \times \Omega \xrightarrow{\quad \Rightarrow_\Omega \quad} \Omega,$$

the statement immediately follows from the aforementioned theorem. □

5.4 Defining set theoretic operations on $\mathcal{P}X$

In set theory, the usual set-theoretic operations on subsets are defined using set comprehension and the usual logical connectives. In this way, the intersection $U \cap U'$ of two subsets $U, U' \subset M$ can be defined as

$$\{x \in M \mid x \in U \wedge x \in U'\}$$

Similarly, we can use the previously defined connective morphisms of the form $\Box_\Omega : \Omega \times \Omega \to \Omega$ to define analogous operations $\Box_X : \mathcal{P}X \times \mathcal{P}X \to \mathcal{P}X$ for some object X in a topos $\mathcal{E}$ as follows.

Definition 5.12. Let $\Box \in \{\wedge, \vee, \Rightarrow\}$. Now define $\Box_X : \mathcal{P}X \times \mathcal{P}X \to \mathcal{P}X$ as the transpose of the map

$$X \times (\mathcal{P}X \times \mathcal{P}X) \xrightarrow{\ \eta\ } (X \times \mathcal{P}X)^2 \xrightarrow{\in_X \times \in_X} \Omega^2 \xrightarrow{\ \Box_\Omega\ } \Omega, \tag{5.1}$$

where $\eta = \langle \mathrm{id}_X \times \pi_1, \mathrm{id}_X \times \pi_2 \rangle$.

Notation 5.13. For two morphisms $f, g : Y \to \mathcal{P}X$ and $\Box \in \{\wedge, \vee, \Rightarrow\}$, let $f \Box_X g$ denote the composition $\langle f, g \rangle \gg \Box_X$.

Now with the following lemma, we demonstrate that this definition makes sense. It essentially states that it does not matter whether we first apply the connective $\Box_\Omega$ to two predicates ϕ, ψ on some product $X \times Y$ and then take the resulting predicate's transpose or whether we first take their transposes and then connect them using the operation $\Box_X$.

Theorem 5.14. *For morphisms $\phi, \psi : X \times Y \to \Omega$ and $\square_\Omega \in \{\wedge_\Omega, \vee_\Omega, \Rightarrow_\Omega\}$ the following equation holds:*

$$\widehat{\phi \,\square_\Omega\, \psi} = \widehat{\phi}\,\square_X\,\widehat{\psi}$$

Proof. Consider the following diagram:

$$\begin{array}{ccccccc}
X\times Y & \xrightarrow{\Delta_{X\times Y}} & (X\times Y)^2 & \xrightarrow{f\times g} & \Omega^2 & \xrightarrow{\diamond} & \Omega \\
\downarrow{\scriptstyle \mathrm{id}_X\times\langle\widehat{f},\,\widehat{g}\rangle} & & \downarrow{\scriptstyle (\mathrm{id}_X\times\widehat{f})\times(\mathrm{id}_X\times\widehat{g})} & & \| & & \| \\
X\times(\mathcal{P}X\times\mathcal{P}X) & \xrightarrow{\eta} & (X\times\mathcal{P}X)^2 & \xrightarrow{\in_X\times\in_X} & \Omega^2 & \xrightarrow{\diamond} & \Omega \\
\downarrow{\scriptstyle \mathrm{id}_X\times\diamond_X} & & & & & & \| \\
X\times\mathcal{P}X & & & \xrightarrow{\in_X} & & & \Omega
\end{array}$$

The upper left square commutes for formal reasons. The upper right square trivially commutes. The upper middle square is simply the product of the transpose identities for f and g, and thus it also commutes. The lower square commutes by the definition of $\square_X$ as the transpose of the horizontal composition through the middle. Thus the outer square commutes, and so by the uniqueness of the transpose, the equality holds. □

By the definition of the name map as a transpose, we easily get the following corollary:

Corollary 5.15. *For two predicates $\phi, \psi : X \to \Omega$*

$$\ulcorner \phi \,\square_\Omega\, \psi \urcorner = \ulcorner\phi\urcorner\,\square_X\,\ulcorner\psi\urcorner$$

holds.

6 The Fundamental Theorem of Topos Theory

6.1 Slice categories and the change-of-base functor

In this section, let $\mathcal{C}$ be a category and X an object of $\mathcal{C}$.

Definition 6.1. The **slice category** $\mathcal{C}/X$ of $\mathcal{C}$ over X is the category whose objects are morphisms $A \xrightarrow{f} X$, $B \xrightarrow{g} X$ into X. Its morphisms $h : f \to g$ are commuting triangles, i.e., morphisms $h : A \to B$ in $\mathcal{C}$, such that $h \gg g = f$.

Now every morphism $j : X' \to X$ the assignment $f \mapsto j^*f$ for objects f of $\mathcal{C}/X$ defines a functor $j^* : \mathcal{C}/X \to \mathcal{C}/X'$ between slice categories, which is commonly called the **change-of-base** functor [MM92, cf. p. 59].

As already seen in chapter 5, we want to show that pullbacks preserve colimits. Now this just translates into the statement "the change-of-base-functor preserves colimits". Furthermore, it is a fact that "left adjoints preserve colimits" [MM92, p. 22]. Now by [MM92, p. 59, Thm.4], we can find a right adjoint to j^* if $\mathcal{C}/X$ is Cartesian closed.

6.2 Proof of the fundamental theorem

We now know that in order to prove that in a topos $\mathcal{E}$ all pullbacks preserve colimits, we have to prove that for any object X of $\mathcal{E}$ the slice category $\mathcal{E}/X$ is again a topos. By 3.12, it suffices to prove that $\mathcal{E}/X$ has all pullbacks, a subobject classifier, and all power objects.

Remark 6.2. e can easily see that $\mathrm{id}_X : X \to X$ is the terminal object of $\mathcal{E}/X$.

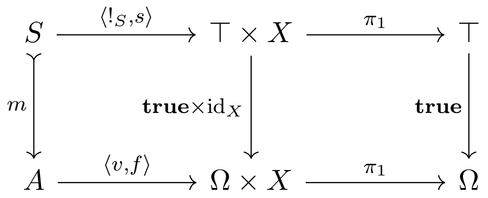
Additionally, the universal property of the product in $\mathcal{E}/X$ essentially turns into the universal property of the pullback in $\mathcal{E}$. We see that product $f \times g$ in $\mathcal{E}/X$ is given by the morphism $f^*g \gg f : A \times_X B \to X$. Because the topos $\mathcal{E}$ has all pullbacks, $\mathcal{E}/X$ has all products.

Furthermore, we can see that equalizers in $\mathcal{E}/X$ are just equalizers in $\mathcal{E}$.

Thus, $\mathcal{E}/X$ has all finite limits.

Theorem 6.3. *The element $\Omega_X := \Omega \times X \xrightarrow{\pi_2} X$ of $\mathcal{E}/X$ forms a subobject classifier together with the morphism $\boldsymbol{true}'_X$ in $\mathcal{E}/X$ given by $\langle \boldsymbol{true}_X, id_X\rangle : X \to \Omega \times X$.*

Proof. Let $f : A \to X$ and $s : S \to X$ be elements in $\mathcal{E}/X$ and $m : f \to g$ be a monomorphism. It is known that morphisms in $\mathcal{E}/X$ are monomorphisms if and only if their underlying morphism in $\mathcal{E}$ is one. We can also easily prove that a square in $\mathcal{E}/X$ is a pullback if and only if its underlying square is a pullback. From the triangle identity we know that all morphisms $f \to \Omega_X$ are of the shape $\langle v, f\rangle$ for some $v : A \to \Omega$. Thus it suffices to find a unique such v, turning the left square below into a pullback.



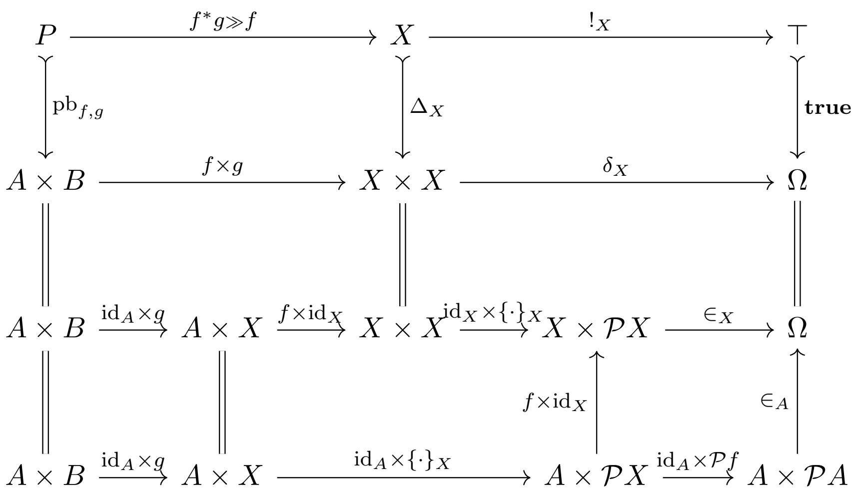
It is easy to see that χ_s makes this square a pullback. It is left to show that χ_s is unique with this property. Note that the right square is obviously a pullback. Thus, for any other v making the left square a pullback, by the pasting lemma the entire square becomes a pullback. Then the uniqueness of χ_m implies $\chi_m = v$, which was to be proven. □

Next we construct the power objects of some $A \xrightarrow{f} X$ in $\mathcal{E}/X$. To this end, we need the following lemma:

Lemma 6.4. *For two morphisms $f : A \to X$, $g : B \to X$, the morphism $pb_{f,g} := \langle f^*g, g^*f \rangle$ is a monomorphism and*

$$\chi_{pb_{f,g}} = (g \gg \widetilde{\{\cdot\}_B \gg \mathcal{P}f})$$

Proof. Consider the following diagram:



The upper left square is a pullback by Lemma 2.1. The upper right square is just a pullback by the definition of δ_X. Thus the combined upper square is also a pullback.

Being a pullback of a monomorphism, $\text{pb}_{f,g}$ is also a monomorphism. Furthermore, it is characterized by the morphism $(f \times g) \gg \delta_X$.

Now, the center left, bottom right, and bottom center squares obviously commute. The center right square commutes by the definition $\{\cdot\}_X$. The bottom left square is just (3.2). Thus

$$(f \times g) \gg \Delta_X = (\mathrm{id}_A \times (g \gg \{\cdot\}_X \gg \mathcal{P}f)) \gg \in_A,$$

which is just the inverse transpose given above. □

To that end, consider the following diagram:

$$\begin{array}{ccccc} \mathrm{Eq} & \xrightarrow{\ \mathrm{eq}\ } & \mathcal{P}A \times X & \overset{p}{\underset{t}{\rightrightarrows}} & \mathcal{P}A \\ & & \big\downarrow {\scriptstyle \mathrm{id}_{\mathcal{P}A} \times \{\cdot\}_X} & & \big\uparrow {\scriptstyle \wedge_A} \\ & & \mathcal{P}A \times \mathcal{P}X & \xrightarrow[\mathrm{id}_{\mathcal{P}A}]{} & \mathcal{P}A \times \mathcal{P}A \end{array}$$

Here, t is just the composition of the lower three morphisms, and eq is the equalizer of p and t. We will now show that $\mathcal{P}_X f := \mathrm{eq} \gg \pi_2$ is a power object.

In [MM92], this proof is given by a long chain of natural isomorphisms. Here (and in Lean), we essentially walk through these isomorphisms step by step and manually check that the resulting assignments are inverse to each other.

Let $B \xrightarrow{g} X$ be another object of $\mathcal{E}/X$. By definition of Ω_X, any morphism $f \times g \to \Omega_X$ is uniquely determined by a morphism $A \times_X B \to \Omega$. Let h be such a morphism. Pulling **true** back along h gives us a monomorphism into $A \times_X B$. We can compose it with $\mathrm{pb}_{f,g}$ to receive a monomorphism into $A \times B$. This map then has the following characteristic morphism:

$$\mathbf{d}(h) := \chi_{(h^* \mathbf{true}) \gg \mathrm{pb}_{f,g}} : A \times B \to \Omega$$

Because $\mathbf{d}(h)$, interpreted as an isomorphism, is essentially defined to lie under $\mathrm{pb}_{f,g}$, this implies that

$$\mathbf{d}(h) \wedge \chi_{\mathrm{pb}_{f,g}} = \mathbf{d}(h)$$

and thus by Lemma 6.4 also

$$\widehat{\mathbf{d}(h)} \wedge_{\mathcal{P}A} (g \gg \{\cdot\}_B \gg \mathcal{P}f) = \widehat{\mathbf{d}(h)} : B \to \mathcal{P}A$$

Therefore, $\langle \widehat{\mathbf{d}(h)}, g \rangle : B \to \mathcal{P}A \times X$ equalizes π_1 and t. Hence, there exists a unique $\mathbf{t}(h) : B \to \mathcal{P}_X f$ lifting $\langle \widehat{\mathbf{d}(h)}, g \rangle$ through eq. We want to show that this is the transpose of h.

We can also walk these steps backwards. Any morphism $g \to \mathcal{P}_X f$ is uniquely determined by a morphism $k : B \to \mathcal{P}_X f$. From this, we receive a morphism $\mathbf{di}(k) := k \gg \text{eq} \gg \pi_1 : B \to \mathcal{P}A$. Note that because eq is the equalizer of π_1 and t, this implies

$$\widetilde{\mathbf{di}(k)} \wedge \chi_{\text{pb}_{f,g}} = \widetilde{\mathbf{di}(k)} : A \times B \to \Omega$$

Lastly, set $\mathbf{it}(k) := \text{pb}_{f,g} \gg \widetilde{\mathbf{di}(k)}$.

Intuitively it might be quite easy to see how these are inverses to each other. However, giving a formal proof of that fact is not as easy.

Lemma 6.5. *For every $h : A \times_X B \to \Omega$, we know that $\boldsymbol{it}(\boldsymbol{t}(h)) = h$.*

Proof. First, we examine $\mathbf{di}(\mathbf{t}(h))$. Because $\mathbf{t}(h)$ is defined as the lift of $\langle \widehat{\mathbf{d}(h)}, g \rangle$ into the equalizer Eq, we get the following equality:

$$\mathbf{di}(\mathbf{t}(h)) = \mathbf{t}(h) \gg \text{eq} \gg \pi_1 = \langle \widehat{\mathbf{d}(h)}, g \rangle \gg \pi_1 = \widehat{\mathbf{d}(h)}$$

Plugging this into the definition of **it** and using the pullback pasting law, we get

$$\mathbf{it}(\mathbf{t}(h)) = \text{pb}_{f,g} \gg \chi_{(h^* \mathbf{true}) \gg \text{pb}_{f,g}} = \chi_{h^* \mathbf{true}} = h,$$

which was to be shown. □

We also need the following lemma, which we do not prove here. However, it is proven in the accompanying formalization.

Lemma 6.6. *For morphisms $k : B \to B'$ and $h : B' \to \mathcal{P}_X A$, we have*

$$(id_f \times k) \gg \boldsymbol{it}(h) = \boldsymbol{it}(k \gg g)$$

The last lemma we need shows the other direction of the inverse.

Lemma 6.7. *For every $k : g \to \mathcal{P}f$, we know that $\boldsymbol{t}(\boldsymbol{it}(k)) = k$ holds (where we interpret k as a morphism in $\mathcal{C}$).*

Proof. First, consider $\mathbf{d}(\mathbf{it}(k))$. From the pullback pasting lemma and Theorem 5.6, we know that the following equalities hold.

$$\begin{aligned}\mathbf{d}(\mathbf{it}(k)) = \chi_{((\text{pb}_{f,g} \gg \widetilde{\mathbf{di}(k)})^* \mathbf{true}) \gg \text{pb}_{f,g}} &= \chi_{\text{pb}_{f,g}{}^* (\widetilde{\mathbf{di}(k)}^* \mathbf{true}) \gg \text{pb}_{f,g}} \\ = \chi_{\text{pb}_{f,g}} \wedge \chi_{\widetilde{\mathbf{di}(k)}^* \mathbf{true}} = \chi_{\text{pb}_{f,g}} \wedge \widetilde{\mathbf{di}(k)} &= \widetilde{\mathbf{di}(k)}\end{aligned}$$

Now because k is also a morphism in $\mathcal{E}/X$, we know that $k \gg \text{eq} \gg \pi_2$ holds. Therefore, we know that

$$\begin{aligned}\mathbf{t}(\mathbf{it}(k)) \gg \text{eq} &= \langle \widehat{\mathbf{d}(\mathbf{it}(k))}, g \rangle = \langle k \gg \text{eq} \gg \pi_1, g \rangle \\ &= \langle k \gg \text{eq} \gg \pi_1, k \gg \text{eq} \gg \pi_2 \rangle = k \gg \text{eq}.\end{aligned}$$

Because eq is a monomorphism, we are finished. □

We can now use these lemmas to prove that we $\mathcal{P}f$ is indeed a power object of f in $\mathcal{E}$.

Theorem 6.8. *The morphism $\mathcal{P}_X f$ forms a power object with $\in_f := \boldsymbol{it}(id_{\mathcal{P}_X f})$, where the transpose of a morphism $h : f \times g \to \Omega_X$ is given by $\boldsymbol{t}(h)$.*

Proof. Let $h : f \times g \to \Omega_X$ be arbitrary. Now from Lemmas 6.6 and 6.5, we get the following equality:

$$(\mathrm{id}_f \times \mathbf{t}(h)) \gg \mathbf{it}(\mathrm{id}_{\mathcal{P}_X f}) = stri(\mathbf{t}(h)) = h$$

It is easy to prove the uniqueness of the map using Lemma 6.7. □

7 The Formalization in Lean

In this section, we give an overview of the formalization in Lean of some of the results at hand. First, we give a short introduction to the Lean theorem prover. Then, we describe the formalization at hand. As already mentioned, it is based on [Con25]; however, some crucial definitions have been changed. Then we explain how the proof that `Type u`, Lean's analogue of the category **SET**, is a topos works. Lastly we talk about a major design choice, namely whether the existence of subobject classifiers and power objects should be implemented by giving their data or by claiming their existence through a proposition. We will see how this decision shapes the formalization of the proof of the fundamental theorem.

7.1 The Lean proof assistant

Lean is a proof assistant based on the formal system of dependent type theory, in which mathematical statements and proofs can be formalized. For any type α (except the empty type), we can construct objects $a : \alpha$ of that type. Types can be thought of as analogoues to sets. In that analogy, the typing judgement $a : \alpha$ just turns into the familiar statement $a \in \alpha$. For two types α and β, we can form a function type $\alpha \to \beta$. There are more type constructors, but this one is the only one relevant to us right now.

Arguably the most important type in Lean is `Prop`. Objects of type `Prop` can be thought of as propositions, which can themselves be thought of as a special kind of type. An object $p : P$ for some proposition $P : \text{Prop}$ is a proof that P holds. Moreover, Lean has "proof irrelevance," meaning that any two proofs $p, q : P$ are definitionally equal.

7.2 An overview of the formalization

Let us first describe the results already present in [Con25]. This repository contains the definitions of subobject classifiers and power objects and some basic theorems about those, as well as the definition of a topos. Furthermore, it includes the construction of exponential objects corresponding to Theorem 3.12 and the construction of a subobject classifier in the category `Type u`. Additionally, it contains an incomplete proof of the finite cocompleteness of a topos (Theorem 1.1), missing proofs of Theorem 4.3 and Lemma 4.9. A more detailed distinction between the author's contributions and those of his predecessor can be found in the file Readme.md of the accompanying formalization.

The fundamental building blocks of this formalization are, to no surprise, the definitions of subobject classifiers, power objects, and topoi:

```
variable (C : Type u) [Category.{v} C] [ChosenTerminalObject C]
class Classifier where
{Ω : C}
t_ : ⊤_ ⟶ Ω
char {U X : C} (m : U ⟶ X) [Mono m] : X ⟶ Ω
isPullback {U X : C} (m : U ⟶ X) [Mono m] :
  IsPullback m (from_ U) (char m) t_
uniq {U X : C} {m : U ⟶ X} [Mono m] {χ : X ⟶ Ω}
  (hχ : IsPullback m (from_ U) χ t_) : χ = char m

variable {C : Type u} [Category.{v} C]
  [CartesianMonoidalCategory C] [Classifier C]
class PowerObject (X : C) where
  {pow : C}
  in_ : X ⊗ pow ⟶ Ω
  transpose {Y : C} (f : X ⊗ Y ⟶ Ω) : Y ⟶ pow
  comm {Y : C} (f : X ⊗ Y ⟶ Ω) :
    ((𝟙 X) ⊗ (transpose f)) ≫ in_ = f
  uniq {Y : C} {f : X ⊗ Y ⟶ Ω} {hat' : Y ⟶ pow}
    (hat'_comm : ((𝟙 X) ⊗ hat') ≫ in_ = f) :
    transpose f = hat'

variable (C)
class ChosenPowerObjects where
  [powerObject (X : C) : PowerObject X]

class Topos (C : Type u) [Category.{v} C]
    [CartesianMonoidalCategory C] where
  [hasPullbacks : HasPullbacks C]
  [classifier : Classifier C]
  [cartesianClosed : CartesianClosed C]
  [chosenPowerObjects : ChosenPowerObjects C]
```

These are not the same definitions as in [Con25]. We will elaborate further on this in chapter 7.4.

The first result achieved in this formalization was the proof of theorem 1.1 was finished. Naturally, this process also included a proof of theorem 4.3.

Additionally, the image factorization from theorem 4.13 was constructed in the context of categories with all pushouts, equalizers, and where all monomorphisms are regular.

Using the existence of colimits and the image factorization, the connectives $\wedge_\Omega, \vee_\Omega, \Rightarrow_\Omega, \neg_\Omega$ were constructed. The conjunction, in the formalization referred to as "meet," was quite straightforward to define:

```
def meet : (Ω : C) ⊗ (Ω : C) ⟶ (Ω : C) := χ_ (t_ ⊗ t_)
```

Moreover, theorem 5.6 was also proven.

Constructing $\vee_\Omega$ and $\Rightarrow_\Omega$ took a bit more work. First, the maps "pointing out" the corresponding truth values in $\Omega \times \Omega$ were defined. Then, $\vee_\Omega$ and

$\Rightarrow_\Omega$ were simply defined to be their characteristic morphisms.

```
noncomputable def join_morph : image (coprod.desc ((t_ : (𝟙_ C) → Ω
    ) ⊗ 𝟙 Ω) (𝟙 Ω ⊗ t_)) → Ω ⊗ Ω :=
  (image.ι (coprod.desc ((t_ : 𝟙_ C → Ω) ⊗ (𝟙 Ω)) ((𝟙 Ω) ⊗ (t_ :
    𝟙_ C → Ω))))
noncomputable def join : (Ω : C) ⊗ (Ω : C) → (Ω : C) :=
  χ_ join_morph
noncomputable def imp_morph :=
  equalizer.ι (meet : _ → (Ω : C)) (fst _ _)
noncomputable def imp : (Ω : C) ⊗ (Ω : C) → (Ω : C) :=
  χ_ imp_morph
```

To be able to define **false** and $\neg_\Omega$, we have to assume that the unique monomorphism from the initial into the terminal object is a monomorphism. Proving that fact requires the fundamental theorem [MM92, p. 194].

```
noncomputable def f_ [Mono (initial.to (𝟙_ C))] : (𝟙_ C) → Ω :=
  χ_ (initial.to (𝟙_ C))
noncomputable def pneg [Mono (initial.to (𝟙_ C))] :
  (Ω : C) → (Ω : C) := χ_ (f_)
```

Furthermore, following the path laid out in chapter 6, subobject classifiers and power objects were constructed in the slice category of a topos. However, the rest of the fundamental theorem could not yet be formalized.

7.3 The topos `Type u`

In mathlib, the type `Type u`, whose objects are all types of some universe level `u`, has already been equipped with an instance of the class `Category`.

This category can be thought of as Lean's analogue to the category **SET** and is therefore the perfect candidate for a first example of a topos in Lean.

In `Type u`, a morphism $f : \alpha \to \beta$ between two types α and β is just a function between α and β, in other words an element of the function type $\alpha \to \beta$, which lives at the same universe level. Thus, `Type u` is obviously Cartesian closed, a fact that has already been formalized in Mathlib as well.

Thus, for us it suffices to construct an instance of `Classifier` and instances `PowerObject X` for every type `X : Type u`. In [Con25], an instance of `Classifier` was already constructed as a type containing only two elements `T`, and `F`. While this corresponds nicely to the intuition of $\{0, 1\}$ as a subobject classifier in **SET**, there is a more elegant construction:

We can lift the type `Prop : Type 0` into our universe level using the function `ULift`. Because in Lean, predicates are just defined as functions into `Prop`, functions X : `Type u`→ `ULift`(`Prop`)don't just **correspond to** predicates on X, they essentially **are** predicates on X.

The monomorphism **true** and the characteristic morphism can then be described as follows:

```
abbrev true_ : 𝟙_ (Type u) ⟶ (ULift.{u} Prop) := fun _ => ULift.up
    True
abbrev char {A B : Type u} (f : A ⟶ B) : B ⟶ (ULift.{u} Prop) :=
  fun b => ULift.up (∃ a : A, f a = b)
```

We can then prove a lemma, characterizing how pullbacks of **true** along predicates behave. After that, constructing a subobject classifier is very simple:

```
instance instClassifier : Classifier (Type u) where
  t_ := true_
  char {U X : Type u} (m : U ⟶ X) [Mono m] := char m
  isPullback {U X : Type u} (m : U ⟶ X) [Mono m] := by {
    rw [isPullback_condition]
    aesop
  }
  uniq {U X : Type u} {m : U ⟶ X} [Mono m] {χ : X ⟶ (ULift.{u}
    Prop)} (h : IsPullback m (ChosenTerminalObject.from_ U) χ
    true_) := by {
    rw [isPullback_condition] at h
    aesop
  }
```

For some type `X : Type u`, the type `Set X : Type u` is defined to be the function type `X → Prop`. Thus it is almost trivial to prove that this forms a power object. Furthermore, the construction draws a nice parallel to equation (3.1).

```
instance instPowerObjectType (X : Type u) : PowerObject X where
  pow := Set X
  in_ := fun x => ULift.up (x.1 ∈ x.2)
  transpose {Y : Type u} (f : X ⊗ Y ⟶ Ω) := fun y => {x | (f
    (x,y)).down}
  comm := by aesop
  uniq := by aesop
```

7.4 Subobject Classifiers and Power Objects: Data or Propositions?

As already mentioned, the basic definitions of subobject classifiers, power objects, and topoi from [Con25] were changed.

In [Con25], topoi are defined using instances

```
class HasClassifier : Prop where
  exists_classifier : Nonempty (Classifier C)
class HasPowerObjects : Prop where
  has_power_object (B : C) : Nonempty (PowerObject B)
```

These state the existence of a subobject classifier and power objects as a propositional statement. In the author's project, we directly use the instances `Classifier C` and `PowerObject C` containing the data required to construct a subobject classifier and power objects.

The former case amounts to saying "a topos is a category in which **there exist** a subobject classifier and power objects," and the latter case amounts to saying "a topos is a category, and **these are** a subobject classifier and power objects in it". While in everyday mathematics this definition might seem rather inconsequential, in Lean it certainly is not.

This is because in [Con25] the definition of power objects only depends on the **existence** of a subobject classifier, but not on the **data** given to construct said subobject classifier.

Because Lean does not have the data to construct an explicit instance of `Classifier C` at hand, it has to generate a generic subobject classifier of the form

```
(HasClassifier C).exists_classifier.some
```

using its in-built choice function.

Now it becomes significantly harder to construct power objects in any given category, because we have almost no idea of what morphisms into this generic classifier look like. This is especially frustrating, as we need to construct a subobject classifier in the first place in order to prove that one exists. This means we essentially lose all the data used for this construction.

On the other hand, using the second definition, power objects depend on an explicit instance of the class `Classifier`. This means that constructing power objects is significantly easier because we have much more information about how morphisms $X \times Y \to \Omega$ behave.

For example, in the construction of the power object in a slice category (cf. theorem 6.8), we can use the definition of Ω_X as the morphism

$$\Omega \times X \xrightarrow{\pi_2} X.$$

Consider the following definition:

```
variable {A} {B : Over X} (f : A ⊗ B → Ω)
abbrev powObj_transpose_subObj : Limits.pullback (f.left ≫ fst _
    _) t_ → A.left ⊗ B.left :=
  (pullback.fst (f.left ≫ fst _ _) (t_)) ≫ (pullback_subObj A.hom
    B.hom)
```

Here, `Over X` is the slice category over some object `X` in the category `C`. `f.left` is simply the underlying morphism of the morphism `f`.

Because of the construction from above, Lean knows that the codomain of `f.left` has to be the product of Ω and `X`. It can even infer the type of the postcomposed projection `fst _ _`.

However this solution is also not optimal. As can be seen in the construction of `join_morph`, under the new definitions, Lean often struggles with typeclass inference.

Additionally, the author's version of the definition of a topos contains an instance of `Cartesian Closed`. This might not seem to make sense, considering that Cartesian closedness follows from the other three axioms.

However, in Lean, seemingly redundant data is often included in the definition of some class or structure if that data is of great use when working in that class or structure. For example, in mathlib a monoid `M` is by definition equipped with a power operation

```
npow : ℕ → M → M,
```

which for all `n :ℕ` returns the function sending `m : M` to its `n`-fold multiplication with itself.

Because Cartesian closedness is a central aspect of the theory of topoi, it makes sense to include it in our definition of a topos. In fact, it is likely easier to work with the Cartesian closedness of a topos. This is because the definition of Cartesian closedness and exponentiable objects does not depend on the power object. Thus, we would face issues like the one described above.

This is also where we encounter problems with Mac Lane's and Moerdijk's approach to the fundamental theorem. The proof of it might be a lot easier to formalize if we follow McLarty's example in [McL92] and construct exponential objects in the slice category instead of power objects.

8 Conclusion

We demonstrated that it is possible to do define topoi and some aspects of their internal language in Lean. However, we have also found out that our approach to topos theory was not optimal.

The author advises advise any future Lean user working with topoi to use the Cartesian closedness of topoi to their full advantage, especially when proving the fundamental theorem of topos theory.

There are two possible directions for the future of categorical logic in Lean. One the one hand, one could follow the path of Mac Lane and Moerdijk towards by formalizing the Mitchell-Bénabou-Language and possibly Kripke-Joyal semantics or even Sheaf semantics. [MM92, cf. Chapter VI]

Towards a slightly different direction, one could follow the path laid out by Johnstone in [Joh02] by defining a more general kind of categorical semantics.

References

[Bec+25a] Lars Becker et al. *A blueprint for the formalization of Carleson's theorem on convergence of Fourier series.* 2025. arXiv: 2405.06423 [math.CA]. URL: https://arxiv.org/abs/2405.06423.

[Bec+25b] Lars Becker et al. *Carleson operators on doubling metric measure spaces.* 2025. arXiv: 2508.05563 [math.CA]. URL: https://arxiv.org/abs/2508.05563.

[BT25] Kevin Buzzard and Richard Taylor. *A Lean proof of Fermat's Last Theorem.* Aug. 2025. URL: https://imperialcollegelondon.github.io/FLT/blueprint.pdf.

[Com22] Johan Commelin. In: *Mitteilungen der Deutschen Mathematiker-Vereinigung* 30.3 (2022), pp. 166–170. DOI: doi:10.1515/dmvm-2022-0058. URL: https://doi.org/10.1515/dmvm-2022-0058.

[Con25] Charlie Conneen. *Charredlee/topos: Topos theory in Lean 4.* 2025. URL: https://github.com/CharredLee/topos.

[Dia75] Radu Diaconescu. "Axiom of choice and complementation". eng. In: *Proceedings of the American Mathematical Society* 51.1 (1975), pp. 176–178. ISSN: 0002-9939.

[Gol14] R Goldblatt. *Topoi: The Categorial Analysis of Logic.* eng. 2nd Ed. Vol. V. Studies in Logic and the Foundations of Mathematics. North-Holland, 2014. ISBN: 9780444867117.

[HD20] Jesse Michael Han and Floris van Doorn. "A formal proof of the independence of the continuum hypothesis". eng. In: *Proceedings of the 9th ACM SIGPLAN International Conference on Certified Programs and Proofs.* New York, NY, USA: ACM, 2020, pp. 353–366. ISBN: 1450370977.

[Joh02] Peter T. Johnstone. *Sketches of an elephant a topos theory compendium 2.* eng. Oxford logic guides 44. Oxford: Clarendon Press, 2002. ISBN: 9780198515982.

[Kos11] Ryszard Paweł Kostecki. *An Introduction to Topos Theory.* June 2011. URL: https://www.fuw.edu.pl/~kostecki/ittt.pdf.

[LMW75] F. W Lawvere, C Maurer, and G. C Wraith. *Model Theory and Topoi.* eng. 1st ed. Vol. 445. Berlin, Heidelberg: Springer Berlin / Heidelberg, 1975. ISBN: 9783540071648.

[Mac71] Saunders Mac Lane. *Categories for the working mathematician.* eng. Graduate texts in mathematics 5. New York [u.a: Springer, 1971. ISBN: 3540900365.

[Mah25] Sridhar Mahadevan. *Topos Theory for Generative AI and LLMs*. 2025. arXiv: 2508.08293 [cs.AI]. URL: https://arxiv.org/abs/2508.08293.

[McL92] Colin McLarty. *Elementary categories, elementary toposes*. eng. Oxford: Clarendon Press, 1992. ISBN: 9780198533924.

[Mit65] Barry Mitchell. *Theory of categories*. eng. 1st ed. Vol. v.Volume 17. New York: Academic Press, 1965. ISBN: 9780124992504.

[MM92] Saunders Mac Lane and Ieke Moerdijk. *Sheaves in geometry and logic a first introduction to topos theory*. eng. Universitext. New York u.a: Springer, 1992. ISBN: 0387977104.

[Tao25] Terence Tao. *PFR Blueprint*. July 2025. URL: https://teorth.github.io/pfr/blueprint.pdf.