Stack Semantics: A Language for Relative Category Theory

Nico Beck
nico@zedat.fu-berlin.de

January 4, 2025

Abstract

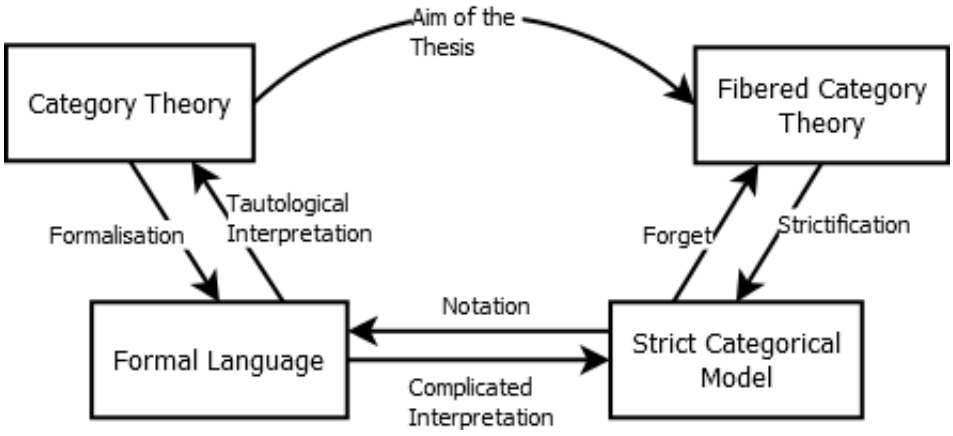
We extend the language of Shulman's *Stack Semantics* [42] so that it can speak about arbitrary indexed categories over a base $\mathcal{S}$ and its slices. We show how that language can be used to do category theory relative to an arbitrary base in a way which looks exactly like ordinary category theory. This gives justification to the principle that most results of category theory generalise to fibered category theory [4].

We take care to demand as little as possible of our base category $\mathcal{S}$. Any site $(\mathcal{S}, W)$ with finite limits such that the self-indexing is a stack will be a suitable model. Classical definitions and results of fibered category theory can be recovered by considering sites $(\mathcal{S}, \mathit{codis})$ of a lex category $\mathcal{S}$ equipped with the codiscrete topology.

Contents

Introduction

It has long been recognised that many constructions and theorems of ordinary category theory can be generalised to fibered categories over a base $\mathcal{S}$ which is sufficiently set-like [21, B3.1][4]. Unfortunately finding the correct translation of concepts and arguments from ordinary category theory into fibered category theory is hard for all but very experienced people. The aim of this thesis is to make the translation process between ordinary category theory and fibered category theory more systematic. We will explain how one can take definitions, theorems and proofs from a standard category theory textbook, such as Emily Riehl's *Category Theory in Context* [40], and turn them step by step into their fibered version. It is of course impossible to describe the translation process in a mathematical precise way when the input consists of informal mathematical text. For that reason we will have to go through a few extra steps: We will specify a formal language which is expressive enough to formalise the results of ordinary category theory we are interested in. We will then explain in a mathematically precise way how that language can be interpreted in the 2-category of Grothendieck fibrations over a base category $\mathcal{S}$. While doing that we will be confronted with a second problem. The intended model $\mathrm{Fib}_{\mathcal{S}}$ of our formal language is not strict enough to make the interpretation process feasible. In order to avoid an endless stream of coherence cells we will have to strictify everything. Here is thus a sketch of our plan:



We will describe the formal language in section 1.1 and its interpretation in the strict categorical model section 1.3. The intended categorical model is the 2-category of Grothendieck fibrations $\mathrm{Fib}_{\mathcal{S}}$ over a lex base category $\mathcal{S}$. The strict categorical model, in which we will actually interpret the language, consists of the 2-categories $[(\mathcal{S}/\Gamma)^{op}, \mathrm{Cat}]$ of (strict) 2-functors, 2-transformations and modifications. We will use a significant amount of 2-category theory and 2-monad theory to mediate between the strict and the non-strict notions. There will be a summary of the strictification procedure in section 1.2. All the missing details can be found in appendix A. If you are willing to take a few 2-categorical results on faith, then you will be able understand the rest of the thesis without diving deeply into 2-category theory. The first chapter contains, additionally to the description of our language, also a gentle and detailed introduction to the interpretation of dependent type theories in

categories in general. The third and the fifth chapter of the thesis show applications of the language. We will in particular show an internal proof of the fibered adjoint functor theorem in chapter 5, and our internal proof will look exactly like the proof of the special adjoint functor theorem from ordinary category theory.

I want to thank Marc Nieper-Wißkirchen and the members of the synthetic algebraic geometry group for inviting me to Augsburg and for being so welcoming in general. I enjoyed participating in the SAG workshops and discussions. I especially want to thank Ingo Blechschmidt for answering all my questions, for giving me all the right hints, and for always being very encouraging.

1 The Language and its Interpretation

The first chapter describes the formal language and its interpretation. We will focus on the language itself and on its formal interpretation in the strictified categorical model. The interplay between the various strict and non-strict notions is discussed in appendix A, although we give a short summary of the most important results in section 1.2.

1.1 The Dependently Typed Language of Category Theory

The first step in our plan is to write down a formal language of naive category theory. The language should allow us to formalise the informal arguments which you typically find in a category theory text book. Our language will be a part of a family of languages called dependent type theories. Those are languages which are very suitable for a categorical semantics due to their highly structured nature. A second advantage is that dependent type theories, such as the *calculus of inductive constructions*, are the foundation of many modern proof assistants. This means that you can get comfortable with those languages by programming in one of those proof assistants. The assistant automatically checks the correctness of your formal proofs, and many assistants have powerful automatic support systems, which can help you when you get stuck. The core part of our language is motivated by Makkai's FOLDS [31] and by the language described in Shulman's paper [42].

A dependent type theory manipulates sequents. A sequent is a finite string of symbols of a particular format. An example of a valid sequent in our language is the following string.

$$C : \mathrm{Cat}, x : C_0, y : C_0, z : C_0, f : C_1(x, y), g : C_1(y, z) \vdash g \circ f : C_1(x, z) \tag{1.1}$$

Every sequent consists of a context and a judgement. The context is written to the left of the "$\vdash$" symbol and the judgement to the right of it. A context consists of a list of variable declarations, together with the information of which sort each variable is. The variables of the context may appear freely in the judgement to the right of it. Classical presentations of first-order languages typically do not use contexts prominently, because there is often only one sort over which all variables vary. The sequents we are interested in will be of the following two forms:

$$\Xi \vdash M : N$$

$$\Xi \vdash M \equiv L : N \tag{1.2}$$

Here Ξ denotes a well-formed context, and M and N can be long complicated strings, which may freely contain the variables listed in the context Ξ. The intuitive meaning of those judgements is: "We judge that M is of type N in the context Ξ." and "We judge that M and L are equal terms of type N in the context Ξ.". It is a feature of type theory that every term always comes with an associated sort[1]. Type theorists believe that this is also a feature of informal mathematical practice. When you write down "n" in an informal proof, then you will implicitly know at the back of your head what n is. The symbol n could for example stand for a natural number or for an integer. This would be denoted by $n : \mathbb{N}$ or $n : \mathbb{Z}$ in a typed language. With that in mind we can now understand the intuitive meaning of the judgement (1.1): "When C is a category, x, y, z are objects of C and f and g are arrows of type $C_1(x, y)$ and $C_1(y, z)$ respectively, then we can judge that the composite $g \circ f$ is a morphism of type $C_1(x, z)$." This is a perfectly sensible and evidently correct statement of naive category theory[2].

Some judgements are correct and some are not. The deduction rules of our language tell us how we can recursively generate correct judgements from old ones. Here is an example of a deduction rule.

$$\frac{\Xi \vdash C : \mathrm{Cat} \qquad \Xi \vdash N : C_1(X, Y) \qquad \Xi \vdash M : C_1(Y, Z)}{\Xi \vdash M \circ N : C_1(X, Z)} \tag{1.3}$$

The idea is that whenever the judgements above the line are correct, then so is the judgement below the line. The symbols Ξ, C, N, M, X, Y, Z are meta-variables which you can replace by a concrete context and concrete terms and types respectively to obtain an applicable instance of the rule.

We work relative to a signature *Sign*. A signature consists of a collection of constant symbols together with declarations of their types. The constants may then be used in deductions without justification.

$$\frac{}{\Xi \vdash M : N} \textit{ (if the signature Sign says so)} \tag{1.4}$$

The constants symbols of a signature are allowed to depend on other types. A valid signature for our language could for example look as follows.

$$\begin{aligned} &\vdash C : \mathrm{Cat} \\ &\vdash X : C_0 \\ &\vdash Y : C_0 \\ x : C_1(X, Y) &\vdash D(x) : \mathrm{Cat} \\ x : C_1(X, Y), y : D(x)_0 &\vdash Z(x, y) : C_0 \end{aligned} \tag{1.5}$$

The signature above declares five constant symbols. Two of them depend on other types of the language. It is important that dependent constants are written in the

[1] The words sort and type are synonyms. A sort is anything which appears to the right of a semicolon, and anything that appears to the left of it is a term.

[2] Naive in the sense of informal, not in the sense of naive.

exact form shown above. We will need the free variables in the strings "$D(x)$" and "$Z(x, y)$" to keep track of reindexings. We won't define what a signature is in a mathematical precise way, since that is more an issue of dependent type theory in general and not of our language in particular.

The set of all derivable sequents is the smallest set of sequents which is closed under the deduction rules of our language. If you want to show that a particular sequent $\mathcal{T}$ is derivable, then you can do this by writing down a derivation tree which ends with $\mathcal{T}$ and all of whose leaves are closed. You can find many examples of such derivation trees in [19, ch 3] if you have never seen them before. Alternatively you can write down a finite list $\mathcal{S}_1, \mathcal{S}_2, ..., \mathcal{S}_n$ of sequents such that $\mathcal{S}_n = \mathcal{T}$ and each sequent $\mathcal{S}_m$ in the list is the result of applying one of the deduction rules of our language to some sequents which appear earlier in the list. Every derivable sequent has such a derivation. We will almost never write down formal derivations, and we will be fairly lax with the rules and syntax of our formal language. We will for example use brackets whenever we need them to disambiguate an expression, and not in a systematic way.

Every dependent type theory comes with a lot of structural rules which specify how substitutions, free and bounded variables and contexts behave. There is for example a contraction rule:

$$\frac{\Xi, x : A, y : A, \Xi' \vdash \mathcal{J}}{\Xi, x : A, \Xi'[x/y] \vdash \mathcal{J}[x/y]}(contr) \tag{1.6}$$

The expression "$\mathcal{J}[x/y]$" denotes the string which you obtain when you replace all instances of the symbol y in the string $\mathcal{J}$ by the symbol x. The structural rules of dependent type theory are interesting in their own right, and they determine much of its semantics. We will not focus on them here, though, and take them for granted, because they are not the topic of the thesis. If you want to see all the structural rules of dependent type theory spelled out then you can for example look into [36, 32, 19, 18]. The list of rules presented here will probably be incomplete at some places. This is not a big problem, since our aim is to eventually use the language informally. We will stay close to the semantics, and this will prevent us from making serious mistakes.

Everything in our language is either a category, an object or a morphism. Categories have a composition operation and identities. The rule (1.3) already expresses that each category has a composition operation. Additionally we add the following deduction rules.

$$\frac{\Xi \vdash C : \mathrm{Cat} \qquad \Xi \vdash X, Y : C_0 \qquad \Xi \vdash N : C_1(X, Y)}{\Xi \vdash N \circ \mathrm{id}_X \equiv N : C_1(X, Y)}$$

$$\frac{\Xi \vdash C : \mathrm{Cat} \qquad \Xi \vdash X : C_0}{\Xi \vdash \mathrm{id}_X : C_1(x, x)}$$

$$\frac{\Xi \vdash C : \mathrm{Cat} \qquad \Xi \vdash X, Y : C_0 \qquad \Xi \vdash N : C_1(X, Y)}{\Xi \vdash \mathrm{id}_Y \circ N \equiv N : C_1(X, Y)}$$

$$\frac{\Xi \vdash N : C_1(X,Y) \qquad \Xi \vdash M : C_1(Y,Z) \qquad \Xi \vdash L : C_1(Z,W)}{\Xi \vdash L \circ (M \circ N) \equiv (L \circ M) \circ N : C_1(X,W)} \tag{1.7}$$

The rules (1.7) express that every object has an identity, that the identities are left and right neutral, and that composition is associative. So far we can not access any categories besides those which appear in the signature *Sign*. To get started let us thus declare that there is always a category of sets and a category of propositions.

$$\frac{}{\mathrm{Set} : \mathrm{Cat}} \qquad \frac{}{\mathrm{Prop} : \mathrm{Cat}} \tag{1.8}$$

We also add rules which allow us to build increasingly complicated proposition from atomic ones.

$$\frac{}{\Xi \vdash \top : \mathrm{Prop}_0} \qquad \frac{}{\Xi \vdash \bot : \mathrm{Prop}_0} \qquad \frac{\Xi \vdash M : C_1(X,Y) \qquad \Xi \vdash N : C_1(X,Y)}{\Xi \vdash M = N : \mathrm{Prop}_0}$$

$$\frac{\Xi \vdash \phi : \mathrm{Prop}_0 \qquad \Xi \vdash \psi : \mathrm{Prop}_0}{\Xi \vdash \phi \wedge \psi : \mathrm{Prop}_0} \qquad \frac{\Xi \vdash \phi : \mathrm{Prop}_0 \qquad \Xi \vdash \psi : \mathrm{Prop}_0}{\Xi \vdash \phi \vee \psi : \mathrm{Prop}_0}$$

$$\frac{\Xi \vdash \phi : \mathrm{Prop}_0 \qquad \Xi \vdash \psi : \mathrm{Prop}_0}{\Xi \vdash \phi \to \psi : \mathrm{Prop}_0} \qquad \frac{\Xi, x : A \vdash \phi : \mathrm{Prop}_0}{\Xi \vdash \exists x : A.\phi : \mathrm{Prop}_0} \qquad \frac{\Xi, x : A \vdash \phi : \mathrm{Prop}_0}{\Xi \vdash \forall x : A.\phi : \mathrm{Prop}_0} \tag{1.9}$$

We allow universal and existential quantification over all types A. This includes the type Cat of categories, types C_0 of objects when C is a category, and types $C_1(x,y)$ of morphisms. On the other hand, we only allow equations of morphisms as propositions. We explicitly do not add the following rule to our language.

$$\frac{\Xi \vdash C : \mathrm{Cat} \qquad \Xi \vdash X, Y : C_0}{\Xi \vdash X = Y : \mathrm{Prop}_0} \tag{1.10}$$

The decision to exclude equalities of objects from the logical part of the language is a purely practical and not an ideological one. It will turn out that there is no good way to interpret propositional equalities of objects in our intended models.

The proposition introduction rules allow us to express quite a lot about a given category C. For example the following sequent is already derivable with the rules we have so far.

$$C : \mathrm{Cat} \vdash \exists t : C_0.\forall x : C_0.\exists! f : C_1(x,t).\top : \mathrm{Prop}_0 \tag{1.11}$$

The proposition expresses that the category C has a terminal object. Be aware that (1.11) does not express that the proposition is true, only that it is a proposition. We can also express that a category has binary products, or pullbacks, power objects, exponentials and all other universal properties which involve only finitely many objects and morphisms in their definition. We could at this point state all the axioms of ETCS [27] for example.

The propositions are not meaningful on their own. We also need to specify what an acceptable proof of proposition is. The logic which we will use is intuitionistic

first-order logic. Intuitionistic means that we do not automatically assume that the law of excluded middle, and all other rules which imply it, hold. The reason for that choice is again a practical one. The law of excluded middle just does not have a sound interpretation in most of the models which we want to consider, and no amount of choice and excluded middle in the meta-theory can change that. By giving up classical reasoning principles we gain many interesting models in which we can soundly interpret our language [7]. To clarify: our meta-language is informal classical mathematics, and you are allowed to use classical reasoning principles in every proof about our formal language and its interpretation. It is only inside the object-language that we restrict ourselves to intuitionistic reasoning principles.

Since everything in our language is a category, an object or a morphism, it is convenient to encode proofs as morphisms in the category Prop. We will work proof irrelevant and hence add the following rule to our language.

$$\frac{\Xi \vdash P : \mathrm{Prop}_1(\phi, \psi) \qquad \Xi \vdash Q : \mathrm{Prop}_1(\phi, \psi)}{\Xi \vdash P \equiv Q : \mathrm{Prop}_1(\phi, \psi)} \tag{1.12}$$

This means that we do not care about the reason why something is true. Proofs contain no data. For a computer system this means that the type checker is allowed to forget what a particular proof term looks like after it has checked its correctness. We will write

$$\Xi \vdash P : \phi \tag{1.13}$$

as a shorthand for the longer sequent $\Xi \vdash P : \mathrm{Prop}_1(\top, \phi)$ and think of P as a proof of ϕ in the context Ξ. We add a lot of rules which explain how we can build and deconstruct proof terms. The rules are fairly standard, so we will not write them all down. The elimination rule of the $\exists$-quantifier for example looks as follows.

$$\frac{\Xi, x : A \vdash \varphi : \mathrm{Prop} \qquad \Xi \vdash \psi : \mathrm{Prop} \qquad \Xi, x : A, p : \varphi \vdash P : \psi \qquad \Xi \vdash Q : \exists x : A.\varphi}{\Xi \vdash \mathrm{elim}_\exists(Q, x.p.P) : \psi} \tag{1.14}$$

You can use any set of rules which encodes first-order intuitionistic logic. At the end of the day it will only matter if a given proposition has a proof or not. We add one non-standard rule to our language which clarifies the meaning of the morphisms in the category Prop. It looks as follows.

$$\frac{\Xi \vdash \phi, \psi : \mathrm{Prop} \qquad \Xi, p : \phi \vdash P : \psi}{\Xi \vdash \nu p.P : \mathrm{Prop}_1(\phi, \psi)} \tag{1.15}$$

The rule looks a bit like the λ-abstraction rule of type theory, but it behaves differently and it semantics and justification are different. We will for that reason use the letter ν instead. One important difference between λ-abstraction and ν-abstraction is, for example, that we cannot stack ν-abstractions, because the sort $\mathrm{Prop}_1(\phi, \psi)$ is not a proposition itself. We will later discover that the category Set satisfies a similar ν-abstraction principle, but in the case of Set we will be able to derive ν-abstraction from more fundamental axioms about Set. This will then hopefully also clarify the meaning of the ν-abstraction rule in the category Prop by analogy.

When we say "there is an internal proof of ϕ" then we mean that the morphism introduction and elimination rules of Prop, which encode first-order intuitionistic logic, allow us to build a term $P : \mathrm{Prop}_1(\top, \phi)$. We will almost never write down formal proof terms, since doing that is very tedious and best done with the help of a computer. We will instead write down informal internal proofs. An informal internal proof should be understood as a description of how one could in principle produce an official proof term if one insists on doing so. In other words, an informal internal proof of ϕ is a building manual of a term $P : \phi$.

Example 1.1. Let us consider the following proposition in our formal language.

$$\forall C : \mathrm{Cat}\,.\forall x, y, z : C_0.\forall f : C_1(x, y).\forall g : C_1(y, z).\, g \circ (f \circ \mathrm{id}_x) = g \circ f : \mathrm{Prop}_0 \tag{1.16}$$

We often omit the symbol "$\vdash$" in a sequent when the context is empty. The proposition above should certainly be constructively provable. An informal internal proof of that statement could look as follows: "Given a category C, and two morphisms $f : C_1(x, y)$ and $g : C_1(y, z)$ we have that $f \circ \mathrm{id}_x = f$ by the right neutrality of id_x and hence $g \circ (f \circ \mathrm{id}_x) = g \circ f$." A formal proof term which encodes the informal proof might look as follows in a proof assistant:

```
lemma Example :
∀C: Cat,∀x y z: C.ob,∀f: C.mor(x,y),∀g: C.mor(y,z), g∘(f∘id x) = g∘f
     :=
λC x y z f g, congr_arg (λh : C.mor(x,y), C.comp g h) (C.rightn f)
```

The proof term is the expression in the last line. The term looks complicated because you have to explain every little step to a computer, exactly as you would have to when explaining the proof to a person who does not know a lot of mathematics yet. With time and practice one can develop tactics and tools which make it easier to produce correct proof terms. ♢

We will assume that the category Set comes with a specific object $\mathbf{1} : \mathrm{Set}_0$. We add the axiom that $\mathbf{1}$ is terminal to our language. Since proofs are morphisms, new axioms can be added by introducing a new constant symbol.

$$\mathrm{ax1} : \forall A : \mathrm{Set}_0.\exists! f : \mathrm{Set}_1(A, \mathbf{1}).\top \tag{1.17}$$

We will use $x : A$ as a shorthand for the longer expression $x : \mathrm{Set}_1(\mathbf{1}, A)$.

We also want to build new interesting categories other than Set and Prop. We extend the language with category constructors to achieve this. Every category constructor should come with the following set of rules.

(i) A category introduction rule. The rule tells you what data you need to form the category in question.

(ii) Object and morphism introduction and elimination rules. The introduction rules bundle data to define objects and morphisms of the new category, while the elimination rules return that data. One should take care that the elimination rules return as much data as the introduction rules take so that nothing is lost.

(iii) Computation rules which tell how eliminators act on constructors, and uniqueness principles, which explain how constructors act on eliminators.

(iv) Equations which explain how composition and identities of the new category look like.

Example 1.2 (The Functor Category). First we need a category introduction rule.

$$\frac{\Xi \vdash C : \mathrm{Cat} \qquad \Xi \vdash D : \mathrm{Cat}}{\Xi \vdash D^C : \mathrm{Cat}} \tag{1.18}$$

Given any two categories, there is a category of functors between them. The object and morphism introduction rules of the category D^C should express that the objects and morphisms are functors and natural transformations respectively. The object introduction rule looks as follows.

$$\frac{\begin{array}{l}\Xi, x : C_0 \vdash M : D_0 \\ \Xi, x : C_0, y : C_0, g : C_1(x,y) \vdash N : D_1(M, M[y/x]) \\ \Xi, x : C_0 \vdash N[\mathrm{id}_x / g] \equiv \mathrm{id}_M : D_1(M,M) \\ \Xi, x : C_0, y : C_0, z : C_0, g : C_1(x,y), h : C_1(y,z) \\ \qquad \vdash N[h/g] \circ N \equiv N[h \circ g / g] : D_1(M, M[z/x])\end{array}}{\Xi \vdash I^0_{\mathrm{func}}(x.M, x.y.g.N) : (D^C)_0} \tag{1.19}$$

The variables x and x, y, g respectively are bounded in the expression $I^0_{\mathrm{func}}(x.M, x.y.g.N)$. The rule looks repulsive, but when you go step by step through the assumptions then you will see that they just specify the data which you collect anyway when you define a new functor in an informal argument. The functor introduction rule bundles all the data and turns it into a new term. The free variables x in M and x, y, f in N have to become bounded, because they are no longer part of the context after we have introduced the new functor. Here are the object elimination rules of the functor category.

$$\frac{\Xi \vdash F : (D^C)_0 \qquad \Xi \vdash X : C_0}{\Xi \vdash F_0 X : D_0} \qquad \frac{\Xi \vdash F : (D^C)_0 \qquad \Xi \vdash N : C_1(X,Y)}{\Xi \vdash F_1 N : D_1(FX, FY)}$$

$$\frac{\Xi \vdash F : (D^C)_0 \qquad \Xi \vdash X : C_0}{\Xi \vdash F\,\mathrm{id}_X \equiv \mathrm{id}_{FX} : C_1(FX, FX)}$$

$$\frac{\Xi \vdash F : (D^C)_0 \qquad \Xi \vdash N : C_1(X,Y) \qquad \Xi \vdash M : C_1(Y,Z)}{\Xi \vdash F(M \circ N) \equiv FM \circ FN : D_1(FX, FZ)} \tag{1.20}$$

The first two rules tell us that we can apply functors to objects and morphisms, and the third and fourth rule allow us to recover the equations which we put into the functor introduction rule. Finally one should explain how the introduction and the elimination rules behave to each other.

$$\begin{aligned} &I^0_{\mathrm{func}}(x.M, x.y.g.N)_0 X \equiv M[X/x] \\ &I^0_{\mathrm{func}}(x.M, x.y.g.N)_1 L \equiv N[L/g] \\ &I^0_{\mathrm{func}}(x.F_0 x, x.y.g.F_1 g) \equiv F \end{aligned} \tag{1.21}$$

I have omitted contexts and types to make the equations more readable. The first two rules are the computation rules and the third rule is the uniqueness principle. The uniqueness principle tells us explicitly that every term of $(C^D)_0$ is equal to one obtained via the object introduction rule. We have to repeat everything with the morphisms now. The introduction and elimination rules for morphisms look as follows.

$$\frac{\begin{array}{ll}\Xi \vdash C, D : \mathrm{Cat} & \Xi, x : C_0 \vdash N : D_1(F_0 x, G_0 x) \\ \Xi \vdash F, G : (D^C)_0 & \Xi, x : C_0, y : C_0, h : C_1(x, y) \vdash N[y/x] \circ F_1 h \equiv G_1 h \circ N\end{array}}{\Xi \vdash I^1_{\mathrm{func}}(x.M) : (D^C)_1(F, G)}$$

$$\frac{\Xi \vdash N : (D^C)_1(F, G) \qquad \Xi \vdash X : C_0}{\Xi \vdash N_X : D_1(F_0 X, G_0 X)}$$

$$\frac{\Xi \vdash N : (D^C)_1(F, G) \qquad \Xi \vdash M : C_1(X, Y)}{\Xi \vdash N_Y \circ F_1 M \equiv G_1 M \circ N_X : D_1(F_0 X, G_0 Y)} \tag{1.22}$$

The computation rule and the uniqueness principle look as follows.

$$\begin{aligned} I^1_{\mathrm{func}}(x.M)_X &\equiv M[X/x] \\ I^1_{\mathrm{func}}(x.N_x) &\equiv N \end{aligned} \tag{1.23}$$

The only rules that are missing are rules which tell us what composition and identities in the functor category look like.

$$\begin{aligned} I^1_{\mathrm{func}}(x.M) \circ I^1_{\mathrm{func}}(x.N) &\equiv I^1_{\mathrm{func}}(x.(M \circ N)) \\ \mathrm{id}_F &\equiv I^1_{\mathrm{func}}(x.\,\mathrm{id}_{F_0 x}) \end{aligned} \tag{1.24}$$

We have written down the complete set of rules of the functor category constructor. ♢

Example 1.3. We can use the rules to construct the identity functor of any category. You can check that the following sequent is derivable in our language.

$$C : \mathrm{Cat} \vdash I^0_{\mathrm{func}}(x.x, x.y.g.g) : (C^C)_0 \tag{1.25}$$

When we apply that functor to some object $X : C_0$ and use the computation rules then we see that it behaves as expected. We will denote the identity functor by $1_C : (C^C)_0$. ♢

Example 1.4. We can internally define the composition of functors. The following sequent is derivable.

$$C, D, E : \mathrm{Cat}, F : (D^C)_0, G : (E^D)_0 \vdash I^0_{\mathrm{func}}(x.F_0(G_0 x), x.y.h.F_1(G_1 h)) : (E^C)_0 \tag{1.26}$$

Applying that functor two morphisms and objects of C shows that it behaves as expected. We will typically write $GF : (E^C)_0$ to denote the composite of two functors. ♢

Example 1.5. We can use the rules of the functor category to define the whiskering of a transformation by a functor. The following sequent is derivable.

$$C, D, E : \mathrm{Cat},\ F, G : (D^C)_0, \alpha : (D^C)_1(F, G), H : (E^D)_0 \\ \vdash I^1_{\mathrm{func}}(x.H_1(\alpha_x)) : (E^C)_1(HF, HG) \tag{1.27}$$

We could of course expand HF and HG by using the expression of the previous example, but it becomes unreadable fairly quickly. We will use the informal notation $H\alpha : (E^C)_1(HF, HG)$ as a shorthand for the whiskering of α with H. ♢

Example 1.6. We can combine the rules of the functor category with that of the product of categories to derive the following sequent.

$$C : \mathrm{Cat} \vdash I^0_{\mathrm{func}}(x.(x, x), x.y.f.(g, g)) : ((C \times C)^C)_0 \tag{1.28}$$

The term internally represents the diagonal of C, and we will often denote that functor by Δ_C. Its interpretation will be exactly the diagonal of the interpretation of C. The rules of the binary product of categories can be found in appendix B. ♢

The most important category constructors which we will use in this thesis are listed below. You can find their rules in appendix B. They all behave exactly as they do in ordinary category theory, so you do not have to learn their rules by heart. You can look them up, though, whenever you feel unsure about a step in an informal internal argument.

$$\frac{\Xi \vdash C : \mathrm{Cat} \qquad \Xi \vdash D : \mathrm{Cat}}{\Xi \vdash C \times D : \mathrm{Cat}} \qquad \frac{\Xi \vdash C : \mathrm{Cat} \qquad \Xi \vdash X : C_0}{\Xi \vdash C/X : \mathrm{Cat}}$$

$$\frac{\Xi \vdash A, B, C : \mathrm{Cat} \qquad \Xi \vdash F : (C^A)_0 \qquad \Xi \vdash G : (C^B)_0}{\Xi \vdash F/G : \mathrm{Cat}}$$

$$\frac{}{\mathrm{sCat} : \mathrm{Cat}} \qquad \frac{\Xi \vdash C : \mathrm{sCat}}{\Xi \vdash [C] : \mathrm{Cat}} \qquad \frac{\Xi \vdash A : \mathrm{Set} \qquad \Xi, x : A \vdash C : \mathrm{Cat}}{\Xi \vdash \prod_{x:A} C : \mathrm{Cat}}$$

$$\frac{\Xi \vdash C : \mathrm{Cat}}{\Xi \vdash \mathrm{Fam}(C) : \mathrm{Cat}} \qquad \frac{}{J : \mathrm{Cat}}\ (J \text{ is a finite category})$$

$$\frac{\Xi \vdash C : \mathrm{Cat}}{\Xi \vdash C^{op} : \mathrm{Cat}} \qquad \frac{\Xi \vdash F : (C^C)_0}{\Xi \vdash F\,\mathrm{alg} : \mathrm{Cat}} \qquad \frac{\Xi \vdash C : \mathrm{Cat} \qquad \Xi, x : C_0 \vdash \phi : \mathrm{Prop}_0}{\Xi \vdash \mathrm{Full}(C, x.\phi) : \mathrm{Cat}} \tag{1.29}$$

The categories listed above are in order of appearance: the binary product of two categories, the slice category, the comma category, the category of small categories and functors between them, the externalisation of a small category, the product of a small family of categories, the category of families of a category C, finite categories (there is one for each finite category J in our meta-language), the opposite of a category, the category of algebras for an endofunctor and the full subcategory of a category cut out by some predicate on its type of objects. The reader is encouraged

to come up with their own constructors. Once we understand how the semantics works, we will see that there is nothing stopping us from introducing more and more complicated categories, like that of internal groups, or internal algebras for an internal ring, or internal sheaf topoi for an internal site.

We have completed the first step of our plan. We have a formal dependently typed language $L(Sign)$ of naive category theory. The language here is a minimal working version, and we will make many additions to $L(Sign)$ later.

1.2 A Summary of the Strictification Process

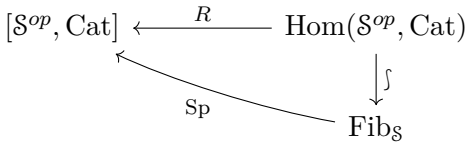
Let us now start from the other side. This section shortly summarise how the strictification process works. All results in this section come out of the existing work on 2-monads. Precise references, details and pointers to the literature can be found in appendix A. Given a category $\mathcal{S}$ we may look at the 2-categories $\mathrm{Fib}_{\mathcal{S}}$ and $\mathrm{Fib}_{\mathcal{S}}^{sp}$ of Grothendieck fibrations and split Grothendieck fibrations with base $\mathcal{S}$. The morphisms in $\mathrm{Fib}_{\mathcal{S}}^{sp}$ are assumed to preserve the splitting strictly. There is also a 2-category $[\mathcal{S}^{op}, \mathrm{Cat}]$ of indexed categories, 2-transformations and modifications and a 2-category $\mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})$ of weak indexed categories, pseudo-transformations and modifications. Finally, there is a 2-category $\mathrm{Ps}(\mathcal{S}^{op}, \mathrm{Cat})$ whose 0-cells are strict, but which has the weak 1-cells of $\mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})$. Correspondingly there is a 2-category whose objects are split fibrations but whose 1-cells are allowed to preserve the splitting only up to coherence cells. We denote that 2-category by $\mathrm{Fib}_{\mathcal{S}}^{ps}$. Indexed categories and Grothendieck fibrations are related via the Grothendieck construction. There is a commuting diagram of 2-functors.

$$\begin{array}{ccccc}
[\mathcal{S}^{op}, \mathrm{Cat}] & \longrightarrow & \mathrm{Ps}(\mathcal{S}^{op}, \mathrm{Cat}) & \longrightarrow & \mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat}) \\
\downarrow\wr & & \downarrow\wr & & \downarrow\wr \\
\mathrm{Fib}_{\mathcal{S}}^{sp} & \longrightarrow & \mathrm{Fib}_{\mathcal{S}}^{ps} & \longrightarrow & \mathrm{Fib}_{\mathcal{S}}
\end{array}$$

The vertical 2-functors, all given by the Grothendieck construction, are Cat-enriched equivalences [20, theorem 10.6.16]. What this means is that it does not hurt to work with indexed categories instead of Grothendieck fibrations.

We want to use our language to prove results about $\mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})$, but it will be easier to interpret the language in the 2-category $[\mathcal{S}^{op}, \mathrm{Cat}]$. The forgetful 2-functor $J : [\mathcal{S}^{op}, \mathrm{Cat}] \to \mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})$ is unfortunately not a biequivalence. The two 2-categories are genuinely different. It turns out that the relation between the two 2-categories is well captured by formal homotopy theory. There is a Cat-enrichment compatible model structure on the underlying 1-category of $[\mathcal{S}^{op}, \mathrm{Cat}]$ whose weak equivalences are precisely the 1-cells w such that Jw is an equivalence in $\mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})$. The fibrant objects in that model structure are called *coflexible*, and the objects which are equivalent to a fibrant object are the *semicoflexible* objects [14]. An indexed category C is semicoflexible precisely when any pseudo-1-cell $A \rightsquigarrow C$ pointing into C can be replaced by an isomorphic strict 1-cell. The 2-functor J is the 2-localisation of $[\mathcal{S}^{op}, \mathrm{Cat}]$ at the class of weak equivalences described above [43, p. 13]. The 2-functor J has a right 2-adjoint R and the underlying functor of RJ together with underlying transformation of $RJ \to 1$ are a functorial fibrant

replacement for the model structure on the underlying category of $[\mathcal{S}^{op}, \mathrm{Cat}]$. The right 2-adjoint is, up to the Grothendieck construction, the well-known right adjoint splitting of a fibration. The following diagram of 2-functors commutes up to an invertible 2-transformation.



The fibers of $\mathrm{Sp}\, C$ are $(\mathrm{Sp}\, C)_\Gamma = \mathrm{Fib}_{\mathcal{S}}(y\Gamma, C)$ and the reindexing operations are defined through whiskering in the 2-category $\mathrm{Fib}_{\mathcal{S}}$. The 2-functor Sp is of course not a biequivalence, but it is hom-cat-wise an equivalence and thus yields a biequivalence of $\mathrm{Fib}_{\mathcal{S}}$ with its essential image in $[\mathcal{S}^{op}, \mathrm{Cat}]$.

Proposition 1.7. [43, p. 12] *The 2-category* $\mathrm{Fib}_{\mathcal{S}}$ *is biequivalent to the full sub-2-category of* $[\mathcal{S}^{op}, \mathrm{Cat}]$ *generated by the semicoflexible objects. The inverse is* $\int J$.

So working with strict indexed categories is in some sense more and not less general than working with weak indexed categories[3]. Some of the constructions of our language will lead us out of the class of semicoflexible indexed categories. The idea, that one should work with fibrant (=coflexible) strict indexed categories in the injective model structure instead of Grothendieck fibrations, comes from [41].

A model of our language will be more than a category, it will be a lex site $(\mathcal{S}, W)$. We will need the topology W on $\mathcal{S}$ to define the interpretation of the propositions $\phi : \mathrm{Prop}_0$. The codiscrete topology, where all covers are generated by isomorphisms, will always a valid choice. We will later learn that n-separatedness of an indexed category C relative to our topology W will have an effect on the behaviour of C in the internal language. We are thus forced to consider how the various 2-functors, which have been introduced above, relate to n-separatedness. We use the following general definition.

Definition 1.8. Let K be a 2-category and let W be a collection of 1-cells in K. A 0-cell C in K is n-W-separated, where $n = 0, 1, 2$, if for each $w : A \to B$ in W the functor

$$w^* : K(B, C) \to K(A, C) \tag{1.30}$$

is faithful, fully faithful or an equivalence respectively.

The covering sieves of the topology W can be seen as 1-cells $w : S \to y\Gamma$ in each of the 2-categories which we have mentioned above. Here $y\Gamma$ is either the discrete indexed category $\mathcal{S}(-, \Gamma)$ or the discrete fibration associated with it. The 1- and 2-separated objects in $\mathrm{Fib}_{\mathcal{S}}$ are precisely prestacks and stacks as they are usually defined. It turns out that 0- and 1-separatedness does not depend on the surrounding 2-category, but 2-separatedness does.

[3]Our 2-categorical notions are per default always the strict once. The term "indexed category" without qualifier will always refer to a strict 2-functor $C : \mathcal{S}^{op} \to \mathrm{Cat}$. The first few paragraphs of appendix A contain an overview of the 2-categorical naming conventions in this thesis. A "weak indexed category" is a pseudo-functor $C : \mathcal{S}^{op} \to \mathrm{Cat}$, i.e. one with extra coherence isomorphisms.

Proposition 1.9. *An indexed category C is 0- or 1-separated in $[\mathcal{S}^{op}, \mathrm{Cat}]$ if and only if JC is 0- or 1-separated in $\mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})$. If JC is 2-separated, then C is 2-separated. The reverse holds if C is semicoflexible, but not in general.*

The reason that 2-separatedness depends on the surrounding 2-category is that the category $[\mathcal{S}^{op}, \mathrm{Cat}](S, C)$ of descent contains in general less objects than $\mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})(S, JC)$ when C is not semicoflexible. All comparison isomorphisms in the first category must be identities. We will call an indexed category C a stack when JC is 2-separated. This is the more important than 2-separatedness in $[\mathcal{S}^{op}, \mathrm{Cat}]$.

Corollary 1.10. *The 2-categories of n-separated Grothendieck fibrations is biequivalent to the 2-category of n-separated semicoflexible indexed categories.*

These are almost all the 2-categorical facts that we will need. The results above heavily rely and follow from existing work on 2-monads [6, 24, 25], on 2-limits [39, 23, 5, 9] and on the homotopy theory of 2-categories [26, 33, 12]. Most space in appendix A is spend on a review of weighted 2-limits and 2-monad theory. We use that theory in appendix A to prove that all the category constructors listed in (1.29) preserve 1-separatedness, and that some even preserve 2-separatedness and semicoflexibility.

1.3 Interpretation in the Strict Categorical Model

The language $L(Sign)$ has a tautological interpretation in ordinary category theory. To be precise: Fix an inaccessible cardinal λ and let V_λ be the λth set in the von-Neumann hierarchy. Denote by Set_λ the category whose objects are the elements of V_λ and whose morphisms are the functions between them. The set V_λ is a Grothendieck universe and the category Set_λ will behave like a category of ZFC-sets. We can interpret our language $L(Sign)$ in the 2-category Cat of small categories functors and transformations of our meta-language. The interpretation of a term $C : \mathrm{Cat}$ is a small category, the interpretation of $\mathrm{Set} : \mathrm{Cat}$ is the small category Set_λ, the interpretation of $\mathrm{Prop} : \mathrm{Cat}$ is the category which has only two objects 0 and 1 and a single arrow $0 \leq 1$ between them. The interpretation of an object term $X : C_0$ is an actual object in the interpretation of C, and the interpretation of a morphism term is an actual morphism. One can recursively interpret all the expressions of the language $L(Sign)$ once one has assigned a meaning to the constants in the signature $Sign$. For example, a proposition $\forall x : C_0.\phi$ in the empty context will be interpreted as 1 when $\phi[t/x]$ is 1 for all objects x in the interpretation of C_0 and as 0 else. The interpretation process is tautological by design: Our intention was after all that $L(Sign)$ should formalise a part of our informal meta-language, namely ordinary category theory. The tautological interpretation is not the interpretation we are most interested in.

Definition 1.11. A model of the language $L(Sign)$ is a lex[4] site $(\mathcal{S}, W)$ together with an interpretation of all the constant symbols in the signature $Sign$.[5]

We can not define yet what an "interpretation of the constant symbols of $Sign$" is, because explaining that already requires explaining a lot of the interpretation

[4]Lex stands for left exact. It means that the base category $\mathcal{S}$ should have finite limits.

[5]Warning: This definition is subject to change up until the end of chapter 2.

process. However, we will slowly work our way towards it. The interpretation process is unfortunately so complicated that we have to do it in two steps.

First Step: Stages and Forcing

We fix a lex site $(\mathcal{S}, W)$. It is not good enough to look at $[\mathcal{S}^{op}, \mathrm{Cat}]$, we will have to consider the whole system of 2-categories $[(\mathcal{S}/\Gamma)^{op}, \mathrm{Cat}]$ where Γ ranges over the objects of $\mathcal{S}$. We will call the objects of our base $\mathcal{S}$ stages. We fix a terminal object in $\mathcal{S}$ and call it the empty stage. The empty stage will be denoted by empty space, and when we have to make the empty stage visible then we will put brackets around it.

Definition 1.12. We define a signature $Sign_\Gamma$ for each stage Γ of the base site $(\mathcal{S}, W)$. The signature contains constant symbols $C : \mathrm{Cat}$, $X : C_0$ and $N : C_1(X, Y)$ for all 0-cells C in $[(\mathcal{S}/\Gamma)^{op}, \mathrm{Cat}]$ and for all objects and morphisms X and $N : X \to Y$ in the fiber $C(\mathrm{id}_\Gamma)$ respectively. The signature does not contain any dependent constants.

We will write

$$\Gamma; \Xi \vdash \mathcal{J} \tag{1.31}$$

to denote that "$\Xi \vdash \mathcal{J}$" is derivable in the language $L(Sign_\Gamma)$. This will make it easier to keep the languages $L(Sign_\Gamma)$ apart. The languages at the different stages are not unrelated to each other. A morphism $u : \Delta \to \Gamma$ in the base category $\mathcal{S}$ allows us to restrict the constant symbols of $Sign_\Gamma$ to constant symbols of $Sign_\Delta$. Given some $C : \mathrm{Cat}$ at level Γ let u^*C be the indexed category at level Δ whose vth fiber is $(u^*C)(v) = C(uv)$. This is exactly the 2-functor which you obtain when you precompose $C : (\mathcal{S}/\Gamma)^{op} \to \mathrm{Cat}$ with the 1-cell $(\mathcal{S}/\Delta)^{op} \to (\mathcal{S}/\Gamma)^{op}$ induced by u. Applying C to the morphism $u : u \to \mathrm{id}_\Gamma$ in $\mathcal{S}/\Gamma$ gives us a functor $u^* : C(\mathrm{id}_\Gamma) \to u^*C(\mathrm{id}_\Delta)$ which we can use use to restrict object and morphism constants. Let us denote the resulting morphism of signatures by $u^* : Sign_\Gamma \to Sign_\Delta$. One can extend the action of u^* to an action $u^* : L(Sign_\Gamma) \to L(Sign_\Delta)$ on the languages: Given a derivable sequent $\Xi \vdash \mathcal{J}$ of the language $L(Sign_\Gamma)$ we get a derivable sequent $u^*\Xi \vdash u^*\mathcal{J}$ of the language $L(Sign_\Delta)$ by applying u^* to all the constant symbols which appear in the strings $\mathcal{J}$ and Ξ respectively. When $v : \Theta \to \Delta$ and $u : \Delta \to \Gamma$ are two composable morphisms in the base, then the following diagram will commute strictly.

$$\begin{array}{ccc} L(Sign_\Gamma) & \xrightarrow{u^*} & L(Sign_\Delta) \\ & \searrow^{(uv)^*} & \downarrow{\scriptstyle v^*} \\ & & L(Sign_\Theta) \end{array}$$

It will also be the case that $\mathrm{id}^*_\Gamma : L(Sign_\Gamma) \to L(Sign_\Gamma)$ is the identity. Those facts are the main reason why we work with indexed categories instead of split Grothendieck fibrations. It has often been observed that a universe-like object in the semantics can be used to solve coherence issues in the interpretation of dependent type theories, and for us that role is played by the 2-category Cat of small categories.

We want to define the interpretation of terms in context. The interpretation operation $[\![-]\!]_\Gamma$ should take a whole sequent $\Xi \vdash M : N$ of the language $L(Sign_\Gamma)$ together with a list of values $\underline{L}$ for the free variables declared in the context Ξ and return an interpretation

$$[\![\Xi \vdash M : N, \underline{L}]\!]_\Gamma \tag{1.32}$$

of the term M. The idea is that the free variables get replaced by the values $\underline{L}$ in the interpretation process. When the context and the list of values $\underline{L}$ are empty, then we will write $[\![M : N]\!]_\Gamma$ for the interpretation of the term M. We will define $[\![-]\!]_\Gamma$ by a huge nested recursion on the structure of the language $L(Sign_\Gamma)$. The outer-most recursion is a recursion on the length of the context Ξ. Let us assume that we have already defined interpretations $[\![M : N]\!]_\Gamma$ of terms in the empty context.

Definition 1.13. We define the relation "$\underline{L}$ is a valid assignment of values to the variables in the context Ξ" recursively on the length of Ξ. It is a relation between well-formed contexts of the language $L(Sign_\Gamma)$ and lists of constant symbols from the signature $Sign_\Gamma$. The recursion conditions are:

(i) The empty list is a valid assignment of values to the variables in the empty context.

(ii) When $K : \mathrm{Cat}$ is in the signature $Sign_\Gamma$ and when $\underline{L}$ is a valid assignment of values to the variables in $\Xi[K/x]$, then $K, \underline{L}$ is a valid assignment of values to the variables in $x : \mathrm{Cat}, \Xi$.

(iii) When $C : \mathrm{Cat}$ is derivable in $L(Sign_\Gamma)$, when K is an object in $[\![C : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma)$ and when $\underline{L}$ is a valid assignment of values to the variables in the context $\Xi[[\![C : \mathrm{Cat}]\!]_\Gamma/C, K/x]$, then $K, \underline{L}$ is a valid assignment of values to the variables in the context $x : C_0, \Xi$.

(iv) When $C : \mathrm{Cat}$, $X : C_0$ and $Y : C_0$ are all derivable in $L(Sign_\Gamma)$, when K is a morphism $K : [\![X : C_0]\!]_\Gamma \to [\![Y : C_0]\!]_\Gamma$ in the fiber $[\![C : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma)$ and when $\underline{L}$ is a valid assignment of values to the variables in the context

$$\Xi[[\![C : \mathrm{Cat}]\!]_\Gamma/C, [\![X : C_0]\!]_\Gamma/X, [\![Y : C_0]\!]_\Gamma/Y, K/x]$$

then the list $K, \underline{L}$ is a valid assignment of values to the variables in the context $x : C_1(X, Y), \Xi$.

To understand the definition, and to explain why it has to be so complicated, let us look at an example.

Example 1.14. Assume that $C, D : \mathrm{Cat}$ are two constants from the signature $Sign_\Gamma$. The context

$$x : (C \times D)_0, f : (D^{C \times D})_0, y : D_0, g : D_1(y, f_0 x) \tag{1.33}$$

is a well-formed context of the language $L(Sign_\Gamma)$. Let us try to find out what a valid assignment of values to the variables in that context looks like by using the recursion conditions. The third rule applies, because the context starts with a variable declaration of shape $x : (C \times D)_0$. The rule tells us that the first value of

the list must be some object K in the fiber $[\![C \times D : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma)$. The remaining three values must be a valid assignment of values to the variables in the modified context

$$f : (D^{[\![C\times D:\mathrm{Cat}]\!]_\Gamma})_0, y : D_0, g : D_1(y, f_1 X) \tag{1.34}$$

We can now continue by using the third rule a second time. Note that we have to repeatedly evaluate subexpressions of the context Ξ if we want to find out what valid list of values for its variables is. This is the reason why the definition of the relation looks so complicated. ♢

Definition 1.15. Assume as before that $[\![-]\!]_\Gamma$ is already defined on terms in the empty context. Then we extend it to non-empty contexts via the following recursive rule.

$$\begin{aligned}&[\![x : A,\Xi \vdash M : N, K, \underline{L}]\!]_\Gamma \\ &\quad = [\![\Xi[[\![A : B]\!]_\Gamma/A, K/x] \vdash M[[\![A : B]\!]_\Gamma/A, K/x] : N[[\![A : B]\!]_\Gamma/A, K/x], \underline{L}]\!]_\Gamma\end{aligned} \tag{1.35}$$

Implicit in that definition is the assumption that there is some way to compute the type B of A. We will generally not prove such statements about our language, because they are very hard to prove and often wrong when the language is not carefully designed to be unambiguous. Finding slick ways to omit as much information as possible from the notation while still keeping the overall language unambiguous is a very subtle problem. The process of inferring missing information from syntax is called elaboration, and people spend their whole careers studying that. Since we are mostly interested in the semantics of our language and want to use dependent type theory informally, we will simply ignore all subtle questions about syntax, elaboration and meta-properties of our language and just assume that there is a version of it where it all works out. The many elegant implementations of dependent type theory on the computer indicate that this can indeed be done.

The final step is to describe the interpretation $[\![A : B]\!]_\Gamma$ of terms in the empty context. The interpretation of constant symbols from the signature $Sign_\Gamma$ is the expected one. When $A : B$ is a part of the signature, then $[\![A : B]\!]_\Gamma = A$. The interpretation of the internal composition operation and the identities is also unsurprising.

$$\begin{aligned}&[\![\mathrm{id}_X : C_1(X, X)]\!]_\Gamma = \mathrm{id}_{[\![X:C_0]\!]_\Gamma} \\ &[\![M \circ N : C_1(X, Z)]\!]_\Gamma = [\![M : C_1(Y, Z)]\!]_\Gamma \circ [\![N : C_1(X, Y)]\!]_\Gamma\end{aligned} \tag{1.36}$$

where the second $\circ$ is the composition of $[\![C : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma)$. Next we need to interpret the category constructors. First some general comments: It will be extremely important for the second interpretation step that the interpretation functions $[\![-]\!]_\Gamma$ are strictly compatible with the reindexing operations. The diagram below should always commute strictly!

$$\begin{array}{ccc} L(Sign_\Gamma) & \xrightarrow{[\![-]\!]_\Gamma} & [(\mathcal{S}/\Gamma)^{op}, \mathrm{Cat}] \\ {\scriptstyle u^*}\downarrow & & \downarrow{\scriptstyle u^*} \\ L(Sign_\Delta) & \xrightarrow[[\![-]\!]_\Delta]{} & [(\mathcal{S}/\Delta)^{op}, \mathrm{Cat}] \end{array}$$

Much of the interpretation $[\![-]\!]_\Gamma$ is determined by the requirement that $u^*[\![-]\!]_\Gamma = [\![-]\!]_\Delta u^*$ should hold strictly. Say, for example, we have a category constructor $Cons(A_1, ..., A_n) : \mathrm{Cat}$ in our language which takes a list of terms $\underline{A}$ as input. Say we have defined the global fiber $[\![Cons(\underline{A}) : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma)$ at each level Γ. Then all the other fibers of the indexed category $[\![Cons(\underline{A}) : \mathrm{Cat}]\!]_\Gamma$ are completely determined. To see this let $u : \Delta \to \Gamma$ be any object of the slice $\mathcal{S}/\Gamma$. We can make the following computation.

$$\begin{aligned}[\![Cons(\underline{A}) : \mathrm{Cat}]\!]_\Gamma(u) &= (u^*[\![Cons(\underline{A}) : \mathrm{Cat}]\!]_\Gamma)(\mathrm{id}_\Delta) \\ &= [\![u^* Cons(\underline{A}) : \mathrm{Cat}]\!]_\Delta(\mathrm{id}_\Delta) \\ &= [\![Cons(u^*\underline{A}) : \mathrm{Cat}]\!]_\Delta(\mathrm{id}_\Delta) \end{aligned} \tag{1.37}$$

The first equation holds by the definition of the action of u^* on indexed categories, the second equation holds because $u^*[\![-]\!]_\Gamma = [\![-]\!]_\Delta u^*$ and the third equation holds by the definition of the reindexing action of u on the languages. What this means is that we only have to describe the generic fiber of $Cons(\underline{A}) : \mathrm{Cat}$ at every level Γ. As long as we do it in a way which varies nicely with the stages we will be fine.

Example 1.16 (The Functor Category). As an example let us explain the interpretation of the functor category constructor. The discussion above tells us that we only need to specify what the global fiber $[\![D^C : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma)$ at each stage looks like. The object and morphisms terms of D^C at stage Γ will be interpreted as the objects and morphisms in the fiber $[\![D^C : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma)$. The following definition is thus very reasonable.

$$[\![D^C : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma) = [(\mathcal{S}/\Gamma)^{op}, \mathrm{Cat}]([\![C : \mathrm{Cat}]\!]_\Gamma, [\![D : \mathrm{Cat}]\!]_\Gamma) \tag{1.38}$$

The remaining fibers of $[\![D^C : \mathrm{Cat}]\!]_\Gamma$ are determined by the condition $[\![-]\!]_\Delta u^* = u^*[\![-]\!]_\Gamma$. Let us next sketch how we can interpret the object constructor of the functor category. The interpretation

$$[\![I^0_{\mathrm{func}}(x.M, x.y.g.N) : (D^C)_0]\!]_\Gamma \tag{1.39}$$

must be a 1-cell $[\![C : \mathrm{Cat}]\!]_\Gamma \to [\![D : \mathrm{Cat}]\!]_\Gamma$. It is again enough to describe the global component of this 1-cell, because all other components are already determined by the global components at lower stages:

$$\begin{aligned}[\![I^0_{\mathrm{func}}(x.M, x.y.g.N) : (D^C)_0]\!]_\Gamma(u) &= u^*[\![I^0_{\mathrm{func}}(x.M, x.y.g.N) : (D^C)_0]\!]_\Gamma(\mathrm{id}_\Delta) \\ &= [\![u^* I^0_{\mathrm{func}}(x.M, x.y.g.N) : (D^C)_0]\!]_\Delta(\mathrm{id}_\Delta) \\ &= [\![I^0_{\mathrm{func}}(x.u^*M, x.y.g.u^*N) : (u^* D^{u^* C})_0]\!]_\Delta(\mathrm{id}_\Delta) \end{aligned} \tag{1.40}$$

When X is an object in the fiber $[\![C : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma)$ then $X : C_0$ is part of the signature $Sign_\Gamma$ and we can thus make the following definition.

$$[\![I^0_{\mathrm{func}}(x.M, x.y.f.N) : (D^C)_0]\!]_\Gamma(\mathrm{id}_\Gamma)(X) = [\![M[X/x] : C_0]\!]_\Gamma \tag{1.41}$$

This is the only possible way to define the action on objects if we want that $[\![-]\!]_\Gamma$ respects the judgemental equality "$\equiv$". We define the action on morphisms in a

similar way. We do not have to define coherence cells since we work in the strict world $[(\mathcal{S}/\Gamma)^{op}, \mathrm{Cat}]$ where all coherence cells are identities. Writing down the right interpretations of the functor elimination rule and the morphism introduction and elimination rules is left to the reader. The main point is that huge parts of the interpretation are already determined by the condition $[\![-]\!]_\Delta u^* = u^*[\![-]\!]_\Gamma$ and the requirement that $[\![-]\!]_\Gamma$ should respect "$\equiv$". Very little creative input from our side is needed. $\diamondsuit$

An explicit description of the interpretation of the other category constructors listed in (1.29) can be found in appendix B. We will now describe the interpretation of the two most interesting category constructors, namely Prop and Set.

Definition 1.17. [29, section III.7] Remember that we work relative to a fixed site $(\mathcal{S}, W)$. A sieve S on an object Γ is local if it satisfies the following condition: Whenever $u : \Delta \to \Gamma$ is a morphism and there is a cover $v_i : \Theta_i \to \Delta$ such that each uv_i is in the sieve S, then so is u.

To define $[\![\mathrm{Prop} : \mathrm{Cat}]\!]_\Gamma$ it is enough to define its global fibers at each stage. We let $[\![\mathrm{Prop} : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma)$ be the preorder of local sieves on the object Γ. Reindexing is defined via the standard pullbacks of sieves. To define $[\![\mathrm{Set} : \mathrm{Cat}]\!]_\Gamma$ we make the following definition.

$$[\![\mathrm{Set} : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma) = \mathrm{Fib}_\mathcal{S}(y\Gamma, \partial_1) \tag{1.42}$$

As you can see we are using the right adjoint splitting $\mathrm{Sp}(\partial_1)$ of the self-indexing as our internal category of sets. The Grothendieck fibration ∂_1 is the codomain fibration $\partial_1 : \mathcal{S}^\mathbf{2} \to \mathcal{S}$ of the base category $\mathcal{S}$. The fibered Yoneda lemma tells us that evaluating at the generic object id_Γ of $y\Gamma$ yields an equivalence of categories.

$$\mathrm{Fib}_\mathcal{S}(y\Gamma, \partial_1) \xrightarrow{\sim} \mathcal{S}/\Gamma \tag{1.43}$$

You can thus think of a set $\Gamma \vdash A : \mathrm{Set}$ as of an object π_A in the slice $\mathcal{S}/\Gamma$ plus a little bit of extra data, which allows us to avoid coherence isomorphisms. That extra data is irrelevant up to equivalence but not up to stricter notions of equality. We will soon discuss in more detail what kind of extra data other than its display map an internal set $\Gamma \vdash A : \mathrm{Set}$ contains. The internal reindexing operations of $[\![\mathrm{Set} : \mathrm{Cat}]\!]_\Gamma$ are defined through whiskering in the 2-category $\mathrm{Fib}_\mathcal{S}$.

We also have to define interpretations of the object constructors of Prop. This is the place where the stack semantics from Shulman's paper [42] comes in.

Lemma 1.18. [29, section III.7] *A local sieve S on Γ contains a morphism $u : \Delta \to \Gamma$ if and only if u^*S is the maximal sieve.*

Proof. Assume that u^*S is maximal. The maximal sieve of Δ is a covering sieve, and for each $v : \Theta \to \Delta$ it holds that $uv \in S$. The locality condition tells us that u must also be a part of S. Conversely, when u is part of S then u^*S is maximal by definition of the pullback of sieves. $\square$

We will write $\Gamma \Vdash \phi$ to denote that the interpretation of a formula ϕ without free variables is the maximal sieve at level Γ. The relation "$\Vdash$" is a relation between the

stages of our base category and closed formulas of the languages $L(Sign_\Gamma)$. Lemma 1.18 tells us that we can define $\Vdash$ instead of defining the sieves $[\![\phi : \mathrm{Prop}_0]\!]_\Gamma$ directly. So let us define $\Vdash$ by recursion on the number of logical connectives of the closed formula ϕ.

Definition 1.19 (Forcing)**.** [42, Definition 7.2]. The relation $\Vdash$ between stages Γ and closed formulas ϕ of the languages $L(Sign_\Gamma)$ is defined recursively through the following conditions.

(i) $\Gamma \Vdash \top$ holds always.

(ii) $\Gamma \Vdash \bot$ holds precisely when Γ can be covered by the empty cover.

(iii) $\Gamma \Vdash \varphi \to \psi$ holds if and only if for each $u : \Delta \to \Gamma$ for which $\Delta \Vdash u^*\varphi$ holds it is also the case that $\Delta \Vdash u^*\psi$ holds.

(iv) $\Gamma \Vdash \varphi \wedge \psi$ holds if and only if both $\Gamma \Vdash \varphi$ and $\Gamma \Vdash \psi$ hold.

(v) $\Gamma \Vdash \varphi \vee \psi$ holds if and only if there is a cover $u_i : \Delta_i \to \Gamma$ such that for each i at least one of $\Delta_i \Vdash u_i^*\varphi$ or $\Delta_i \Vdash u_i^*\psi$ hold.

(vi) $\Gamma \Vdash N = M$ holds if and only if there is a cover $u_i : \Delta_i \to \Gamma$ such that $[\![u_i^*N]\!]_{\Delta_i} = [\![u_i^*M]\!]_{\Delta_i}$ for each i.

(vii) $\Gamma \Vdash \forall x : C_0.\varphi$ holds if and only if for each $u : \Delta \to \Gamma$ and constant $X : u^*C$ we have that $\Delta \Vdash u^*\varphi[X/x]$ holds.

(viii) $\Gamma \Vdash \exists x : C_0.\varphi$ holds if and only if there are a cover $u_i : \Delta_i \to \Gamma$ and constants $X_i : u_i^*C$ such that $\Delta_i \Vdash u_i^*\varphi[X_i/x]$ holds for each i.

(ix) The forcing conditions for $\forall$- and $\exists$-quantification over morphisms and over the very large type Cat are analogues to those for objects.

We call "$\Vdash$" the forcing relation. To understand the forcing conditions better, let us look at a very special case.

Example 1.20. Let Set_λ be the category of λ-small sets and equip it with the canonical topology. A family of functions is covering in the canonical topology if and only if the functions are jointly surjective. We can associate a J-indexed family of categories $(C_j)_{j\in J}$ to each 2-functor $C : \mathrm{Set}_\lambda/J \to \mathrm{Cat}$. The category C_j is the generic fiber of the reindexing j^*C of C along the inclusion of the point $j : (\,) \to J$. The operation of sending indexed categories C to $(C_j)_{j\in J}$ can be extended to a 2-functor $[(\mathrm{Set}_\lambda/J)^{op}, \mathrm{Cat}] \to \prod_{j\in J} \mathrm{Cat}$, and that 2-functor is a biequivalence when we restrict our attention to those indexed categories which are semicoflexible stacks in the canonical topology. This means that we can think of terms $C : \mathrm{Cat}$, $X : C_0$ and $N : C_1(X, Y)$ at stage J as of actual J-indexed families of categories, objects and morphisms respectively. It becomes apparent under this identification that the forcing relation expresses exactly "fiberwise truth". Say for example we have a J-indexed family of categories $C = (C_j)_{j\in J}$ and a J-indexed family of morphisms $m_j : (C_j)_1(x_j, y_j)$ in in the fibers. What is the external meaning of the following statement?

$$J \Vdash \forall z : C_0.\forall g, h : C_1(z, x).(mg = mh \to g = h) \tag{1.44}$$

When we remove the first block of quantifiers and the implication arrow at once then we get the following: "For every function $u : I \to J$, for any I-indexed family of objects $z_i \in (C_{u(i)})_0$ and for any two families $g_i, h_i : (C_{u(i)})_1(z, x_{u(i)})$ of morphisms, whenever $I \Vdash (u^*m)g = (u^*m)h$ holds then so does $I \Vdash g = h$." The translation of $J \Vdash g = h$ is that $g_j = h_j$ holds for all $j \in J$. By letting u vary over the points $j : (\ ) \to J$ of J we see that the internal statement (1.44) translates to "each morphism m_j is monic". In fact, when we look at the interpretation of any formula $\phi : \mathrm{Prop}_0$ at the empty stage in the model $(\mathrm{Set}_\lambda, can)$, then it turns out that this interpretation is equivalent to the tautological interpretation of ϕ which we sketched at the beginning of section 1.3. This means that the tautological interpretation of the language $L(Sign)$ is a special case of the more general complicated interpretation which we defined in this section, namely the special case where the model is $(\mathrm{Set}_\lambda, can)$ and we only consider semicoflexible stacks as 0-cells. That the forcing relation expresses fiberwise truth is also the right intuition for other models $(\mathcal{S}, W)$. The problem is that in most categories $\mathcal{S}$ there are not enough test maps out of the empty stage, which is why we also include other test maps $u : \Delta \to \Gamma$ in the definition of the forcing relation. $\diamondsuit$

Remark 1.21. The interpretation of the forcing semantics in the special model $(\mathrm{Set}_\lambda, can)$ turned out to be equivalent to the tautological interpretation of the language $L(Sign)$ in Cat. The proof of that fact felt constructive. One can entertain the thought that it might be possible to repeat the definition of the language $L(Sign)$ (maybe without the category constructors) and the definition of the forcing relation *inside* the internal language which we are developing. To make this plausible we would first have to demonstrate that the internal language is expressive enough to internalise significant parts of the content of this thesis, or alternatively of [42, section 7]. One would then expect that the same result holds internally, namely that the interpretation of the stack semantics in the special (now internal) model (Set, can) is the tautological interpretation. This idea is called the idempotence of the stack semantics. Precise versions of the statement can be found in the original paper [42, lemma 7.20] about the stack semantics and in Ingo Blechschmidt's master thesis. We will not follow this idea in the rest of the thesis, though. $\diamondsuit$

Definition 1.22. We define $[\![\phi : \mathrm{Prop}_0]\!]_\Gamma$ to be the sieve on Γ which contains precisely those $u : \Delta \to \Gamma$ for which $\Delta \Vdash u^*\phi$ holds.

Proposition 1.23. [42, lemma 7.3] *This is an allowed definition. The resulting sieve* $[\![\phi : \mathrm{Prop}_0]\!]_\Gamma$ *is local and hence an actual object of* $[\![\mathrm{Prop} : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma)$.

Proof. This is in essence [42, lemma 7.3] so we will not do a detailed proof here. The proof proceeds by induction over the number of connectives in the closed formula $\phi : \mathrm{Prop}$. Let us do one of the cases as an example. To see that $[\![M = N : \mathrm{Prop}_0]\!]_\Gamma$ is local assume that $u : \Delta \to \Gamma$ is a morphism and $v_i : \Theta_i \to \Delta$ is a cover such that uv_i is in $[\![M = N : \mathrm{Prop}_0]\!]_\Gamma$ for each i. This means that $\Theta_i \Vdash v_i^*u^*M = v_i^*u^*N$ holds for each i. The definition of the forcing relation now tells us that we may find covers $w_{ij} : \Lambda_{ij} \to \Theta_i$ of the Θ_i such that M and N become actually equal after restricting to the Λ_{ij}. The collection of morphisms v_iw_{ij} is a cover of Δ and the definition of the forcing relation now tells us that $\Delta \Vdash u^*M = u^*N$ holds. This means that the

morphism u is already contained in the sieve $[\![M = N : \mathrm{Prop}_0]\!]_\Gamma$, which is what we wanted to show. The other cases are also lengthy but not complicated. □

We still have to define the interpretation $[\![P : \phi]\!]_\Gamma$ of proof terms. We will not have to construct anything since each $[\![\mathrm{Prop} : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma)$ is a preorder. For each proof term $P : \phi$ there is at most one morphism in $[\![\mathrm{Prop} : \mathrm{Cat}]\!]_\Gamma$ which can possibly be its interpretation. We just have to check that that morphism is present.

Proposition 1.24. [42, lemma 7.4] *We can extend the interpretation functions* $[\![-]\!]_\Gamma$ *to proof terms.*

Proof. This is in essence [42, lemma 7.4] so we do not show a full proof. The proof proceeds by induction on the number of constructors and eliminators which appear in the proof term $P : \phi$. Let us do the existence elimination rule (1.14) as an example. We like to define an interpretation of a proof term $\mathrm{elim}_\exists(Q, x.p.P) : \psi$. The only way to get such a term is the elimination rule (1.14), and so we may assume that there are terms $\psi : \mathrm{Prop}_0$, $x : A \vdash \varphi : \mathrm{Prop}_0$, $x : A, p : \varphi \vdash P : \psi$ and $Q : \exists x : A.\varphi$. The interpretation should be a morphism $[\![\top : \mathrm{Prop}_0]\!]_\Gamma \to [\![\psi : \mathrm{Prop}_0]\!]_\Gamma$, so our job is to check that $[\![\psi : \mathrm{Prop}_0]\!]_\Gamma$ is the maximal sieve. We can assume by induction that we already have a sound interpretation of all proof terms which contain less introduction and elimination rules than $\mathrm{elim}_\exists(Q, x.p.P)$. This includes in particular $Q : \exists x : A.\varphi$ and so we see that the sieve $[\![\exists x : A.\varphi : \mathrm{Prop}_0]\!]_\Gamma$ is maximal. The forcing relation tells us that we may find a cover $u_i : \Delta_i \to \Gamma$ and constants $\Delta_i \vdash x_i : u_i^* A$ such that $[\![u_i^*\varphi[x_i/x] : \mathrm{Prop}_0]\!]_{\Delta_i}$ is maximal for each i. This means that there are constants $p_i : u_i^*\varphi[x_i/x]$ in the signature $Sign_{\Delta_i}$. We can derive $u_i^* P[x_i/x, p_i/p] : u_i^*\psi$ in the language $L(Sign_{\Delta_i})$. The proof term "$u_i^* P[x_i/x, p_i/p]$" contains necessarily less eliminators and constructors as the string "$\mathrm{elim}_\exists(Q, x.p.P)$" and we may hence use the induction hypothesis to conclude that each sieve $[\![u_i^*\psi : \mathrm{Prop}_0]\!]_{\Delta_i}$ must be maximal. By locality we see that $[\![\psi : \mathrm{Prop}_0]\!]_\Gamma$ is maximal, which is what we wanted to show. □

Corollary 1.25. [42, lemma 7.4] *The forcing relation is sound with respect to intuitionistic reasoning. if* $\Gamma \Vdash \phi$ *holds and there is an intuitionistic proof that* ϕ *implies* ψ*, then also* $\Gamma \Vdash \psi$ *holds.*

Proof. Informal intuitionistic proofs can be encoded by the proof terms in the language $L(Sign_\Gamma)$ and we have just checked that those proof terms have a sound interpretation. □

Definition 1.26. The interpretation $[\![\mathbf{1} : \mathrm{Set}]\!]_\Gamma$ of the special term $\mathbf{1} : \mathrm{Set}_0$ is the fibered functor $y\Gamma \to \partial_1$ which sends an object $u : \Delta \to \Gamma$ of its total category to the identity id_Δ. It acts on cartesian arrows in $y\Gamma$ as expected. We will be more precise in the next section.

We have now fully described the interpretations $[\![-]\!]_\Gamma$ of the languages $L(Sign_\Gamma)$. When $Sign$ is an arbitrary signature and $m : Sign \to Sign_{(\ )}$ is a morphism of signatures, then we can also interpret the language $L(Sign)$ in the model $(\mathcal{S}, W)$. To do that we extend m to a morphism of languages $L(m) : L(Sign) \to L(Sign_{(\ )})$ and

let the interpretation of $L(Sign)$ in the model $(\mathcal{S}, W)$ be the composite $[\![-]\!]_{(\ )} \circ L(m)$. This means one first turns a sentence of $L(Sign)$ into a sentence of $L(Sign_{(\ )})$ by using m to replace constant symbols and then applies $[\![-]\!]_{(\ )}$ afterwards to get an interpretation in the model $(\mathcal{S}, W)$. The signature $Sign$ has to be very simple and can not contain constants which depend on types, since the same is true for the signature $Sign(\ )$. We will lift that restriction in the next section.

Remark 1.27. When $(\mathcal{S}, W)$ is a positive Heyting category equipped with the coherent topology, when m is the unique morphism $m : \varnothing \to Sign_{(\ )}$ out of the empty signature, when we ignore all category constructors other than Prop and Set and when we ignore the fact that we have strictified the self-indexing of $\mathcal{S}$, then the the language $L(\varnothing)$ and its interpretation $[\![-]\!]_{(\ )} \circ L(m)$ in $(\mathcal{S}, W)$ are the language and interpretation described in the paper [42] by Shulman. ♢

Second Step: Removing the Stage

We have a bunch of languages $L(Sign_\Gamma)$ and a bunch of interpretation functions $[\![-]\!]_\Gamma$. This is not very satisfying, and it would be nice if we could get rid of the stages Γ and combine all the languages $L(Sign_\Gamma)$ to one big internal language of the model $(\mathcal{S}, W)$. Fortunately, there is a collection of very powerful notation tricks which do exactly that. They come from the 1-categorical semantics of dependent type theory and from existing work on categories with families therein. Since everything in this section has already been studied in great detail [17, 18, 13] we will allow ourselves to be a bit less formal and mix syntax and semantics a little.

What kind of data does an internal set contain? Say we have derived the following judgement.

$$\Gamma \vdash A : \mathrm{Set}_0$$

Then A will be an object in the category $\mathrm{Fib}_{\mathcal{S}}(\Gamma, \partial_1)$. The fibered Yoneda lemma tells us that there is an equivalence $\mathrm{Fib}_{\mathcal{S}}(\Gamma, \partial_1) \xrightarrow{\sim} (\partial_1)_\Gamma = \mathcal{S}/\Gamma$ which varies 2-naturally in Γ. A set at stage Γ contains almost but not quite the same data as an object in the slice category $\mathcal{S}/\Gamma$. Given one such A, we write $\pi_A : \Gamma.A \to \Gamma$ to denote its associated object in the slice category. The object π_A is the result of evaluating the fibered functor $A : y\Gamma \to \partial_1$ at the generic object id_Γ in the fiber $(y\Gamma)_\Gamma = \mathcal{S}(\Gamma, \Gamma)$ of $y\Gamma$. We call π_A the display map of $\Gamma \vdash A : \mathrm{Set}_0$ and $\Gamma.A$ the stage extension of Γ by A. The fibered functor $A : y\Gamma \to \partial_1$ must also send each $u : \Delta \to \Gamma$, viewed as an object in the fiber $(y\Gamma)_\Delta = \mathcal{S}(\Delta, \Gamma)$, to an object in $(\partial_1)_\Delta = \mathcal{S}/\Delta$. There is a unique cartesian morphism $u \to \mathrm{id}_\Gamma$ lying above u in the discrete fibration $y\Gamma$, and $A : y\Gamma \to \partial_1$ must send that arrow to a cartesian arrow $A(u) \to A(\mathrm{id}_\Gamma)$ lying above u in the codomain fibration. In other words, the functor A chooses a pullback of π_A along u. The rest of $A : y\Gamma \to \partial_1$ is completely determined by those choices. We see that a 1-cell $A : y\Gamma \to \partial_1$ contains exactly the same data as an object $\pi_A : \Gamma.A \to \Gamma$ in the slice together with a choice of a pullback of π_A along u for each morphism $u : \Delta \to \Gamma$ other than the identity. The

pullback diagrams associated with $\Gamma \vdash A : \mathrm{Set}_0$ must look as follows.

$$\begin{array}{ccc} \Delta.u^*A & \xrightarrow{q(A,u)} & \Gamma.A \\ {\scriptstyle x_{u^*A}}\downarrow & \lrcorner & \downarrow{\scriptstyle x_A} \\ \Delta & \xrightarrow[u]{} & \Gamma \end{array}$$

The notation $q(A, u)$ is an ad-hoc name which will be replaced by a more elegant notation later. The maps $q(A, u)$ are part of the data of $\Gamma \vdash A : \mathrm{Set}$ and we have that $q(A, \mathrm{id}_\Gamma) = \mathrm{id}_{\Gamma.A}$.

Example 1.28. The maps $q(\mathbf{1}, u)$ of the internal set $\Gamma \vdash \mathbf{1} : \mathrm{Set}$ are $q(\mathbf{1}, u) = u$. The display map of $\Gamma \vdash \mathbf{1} : \mathrm{Set}$ is the identity of Γ. ♢

Remember that $\Gamma; \Xi \vdash N : A$ is a short-hand for the sequent $\Xi \vdash N : \mathrm{Set}_1(\mathbf{1}, A)$ in the language $L(Sign_\Gamma)$. We can use the fibered Yoneda lemma to compute what this means in the model. A term $\Gamma \vdash N : A$ of an internal set $\Gamma \vdash A : \mathrm{Set}$ is a morphism in $\mathrm{Set}(\mathrm{id}_\Gamma)$, which is the same thing as a 2-cell $N : \mathbf{1} \Rightarrow A$ in $\mathrm{Fib}_{\mathcal{S}}$. The fibered Yoneda lemma tells us that such a 2-cell is uniquely determined by its component $N(\mathrm{id}_\Gamma) : \mathrm{id}_\Gamma \to \pi_A$ in the slice category $\mathcal{S}/\Gamma$. A morphism $\mathrm{id}_\Gamma \to \pi_A$ in the slice category $\mathcal{S}/\Gamma$ is exactly the same thing as a section of the display map π_A of the internal set A. Hence we see that the elements $\Gamma \vdash N : A$ of an internal set are in one-to-one correspondence with the sections of its display map π_A.

Definition 1.29. The category with families $CwF(\mathcal{S})$ attached to a lex category $\mathcal{S}$ consists of the system of types $A : \Gamma \to \partial_1$ and terms $N : \mathbf{1} \to A$ together with the reindexing actions u^* on those types and terms. For a precise definition of a category with families see [13].

Our models of the language $L(Sign_{(\ )})$ includes in particular the standard[6] category with families associated with the base category category $\mathcal{S}$. We might thus suspect that Martin-Löf's type theory [32] is a proper sublanguage of $L(Sign)$, and we will step by step discover that this is indeed the case.

Lemma 1.30. [18, section 3.1] *Let $\mathcal{S}$ be a lex category and let $\Gamma \vdash A, B : \mathrm{Set}$ be two constants from the signature $Sign_\Gamma$ of $\mathcal{S}$ at stage Γ. Then there are natural bijections between the following sets.*

(i) Constants $\Gamma.A \vdash N : \pi_A^* B$ in the signature $Sign_{\Gamma.A}$.

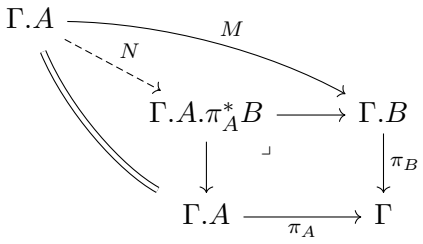
(ii) Constants $\Gamma \vdash L : \mathrm{Set}_1(A, B)$ in the signature $Sign_\Gamma$.

(iii) Morphisms $M : \pi_A \to \pi_B$ in the slice $\mathcal{S}/\Gamma$.

Proof. This is a folklore result about categories with families. Here is a short proof for completeness. The bijection between terms $\Gamma.A \vdash N : \pi_A^* B$ and morphisms $M : \pi_A \to \pi_B$ in the slice category $\mathcal{S}/\Gamma$ follows from staring at the following

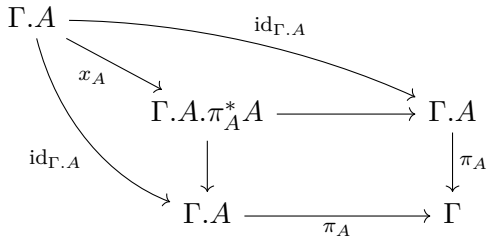
[6] It is the category with families that is most often used in the categorical semantics of extensional dependent type theory, for example in [17, 18]. For the interpretation of intensional type theories the left adjoint splitting of the codomain fibration seems to be more appropriate [28].

diagram.



The one-to-one correspondence between morphisms $M : \pi_A \to \pi_B$ in the slice and terms $\Gamma \vdash L : \mathrm{Set}_1(A, B)$ follows from the fibered Yoneda lemma, or more explicitly from the fully faithfulness of the functor $\mathrm{Fib}_{\mathcal{S}}(y\Gamma, \partial_1) \to \mathcal{S}/\Gamma$ which evaluates at the generic object. □

Construction 1.31 (Variables). Given an internal set $\Gamma \vdash A : \mathrm{Set}_0$ one constructs a very special constant $\Gamma.A \vdash x_A : \pi_A^* A$ at the lower stage $\Gamma.A$. That constant is called "the variable of A". Lemma 1.30 tells us that a term of the internal set $\Gamma.A \vdash \pi_A^* A : \mathrm{Set}_0$ is exactly the same thing as a section of the display map of $\pi_A^* A$. We can thus define x_A via the following diagram.



The rectangle is one of the cartesian arrows in the self-indexing which are part of the data of the internal set $\Gamma \vdash A : \mathrm{Set}_0$. ♢

Instead of giving each variable x_A a name which tells us to which internal set it belongs, we will specify a local name for x_A in the context. When we write

$$\Gamma, x : A; \Xi \vdash \mathcal{J}$$

then the left hand side denotes the stage $\Gamma.A$ while all occurrences of x on the right-hand side have to be interpreted as the variable x_A of A. The second notation trick which we will use is that we will omit reindexings along display maps in the notation. When we write

$$\Gamma, x : A, y : B; \Xi \vdash \mathcal{J}$$

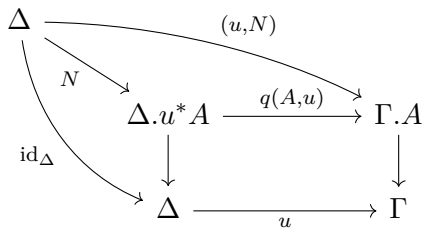
then this means that we have a set $\Gamma; \Xi \vdash A : \mathrm{Set}$, and a second set $\Gamma.A; \pi_A^* \Xi \vdash B : \mathrm{Set}$. All occurrences of A and x in the judgement $\mathcal{J}$ have to be read as $\pi_B^* \pi_A^* A$ and $\pi_B^* x_A$ respectively, while all occurrences of y and B in $\mathcal{J}$ have to be read as x_B and $\pi_B^* B$ respectively. There is never any ambiguity about how many reindexings along display maps one has to add so that an expression makes sense, although we will not prove this formally. There is an important lemma which tells us that morphisms into a context extension can be decomposed.

Lemma 1.32 (Morphisms into Stage Extensions)**.** [18, section 3.1] *We have that*

$$\mathcal{S}(\Delta, \Gamma.A) = \coprod_{u \in \mathcal{S}(\Delta,\Gamma)} \mathrm{El}(\Delta, u^*A)$$

holds naturally where $\mathrm{El}(\Delta, u^*A)$ *denotes the set of constants* $\Delta \vdash N : u^*A$*. We write* $(u, N) : \Delta \to \Gamma.A$ *to denote the morphism associated to a pair* $u \in \mathcal{S}(\Delta, \Gamma)$ *and* $N \in \mathrm{El}(\Delta, u^*A)$*.*

Proof. Proving the lemma involves looking at the following diagram for a while.



Given a pair u and N the morphism (u, N) is the composite of N with $q(A, u)$. Conversely, given a map $f : \Delta \to \Gamma.A$ one sets $u = \pi_A \circ f$ and takes N to be the unique section of π_{u^*A} such that $q(A, u) \circ N = f$. One can show that the two constructions are inverse to each other. □

We remember that the reindexing of an internal set $\Gamma.A \vdash B$: Set along a map $(u, N) : \Delta \to \Gamma.A$ is defined via precomposition with the associated 1-cell $y(u, N) : y\Delta \to y(\Gamma.A)$ in the 2-category $\mathrm{Fib}_{\mathcal{S}}$ of fibrations above $\mathcal{S}$. When we omit the letter y, as one often does in category theory, then we are justified to write $\Delta \vdash B(u, N)$: Set instead of $(u, N)^*B$.

Lemma 1.32 can be used to recursively decompose morphisms of type $\Delta \to \Gamma.A.B$ or into even longer stage extensions. A morphism $\Delta \to \Gamma.A.B$ contains exactly the same data as a map $u : \Delta \to \Gamma$ together with a term $\Delta \vdash N : Au$ and a term $\Delta \vdash M : B(u, N)$. We write $(u, N, M) : \Delta \to \Gamma.A.B$ instead of $((u, N), M)$ to denote the resulting morphism. A morphism $\Delta \to (\ ).A_1.A_2.....A_n$, which points into an object which is build from the empty stage by means of stage extensions, is completely determined by a list of terms $M_1, ..., M_n$ such that term M_1 is of type $\Delta \vdash M_1 : A_1 t_\Delta$, the term M_2 is of type $\Delta \vdash M_2 : A_2(t_\Delta, M_1)$ and so on. Here $t_\Delta : \Delta \to (\ )$ denotes the unique map into the empty stage. Since t_Δ does not contain any information we omit it from the notation and write $(M_1, ..., M_n) : \Delta \to (\ ).A_1.A_2.....A_n$ to denote the map $(t_\Delta, M_1, ..., M_n)$. Morphisms between objects of the shape $(\ ).A_1....A_n$ can be completely described by lists of terms. A category with families is democratic when all its objects can be build from the empty stage via stage extensions. Our category with families $CwF(\mathcal{S})$ is democratic since we can write any stage Γ as $\Gamma = (\ ).A$ by choosing pullbacks of $\Gamma \to (\ )$ along all other maps with codomain $(\ )$.

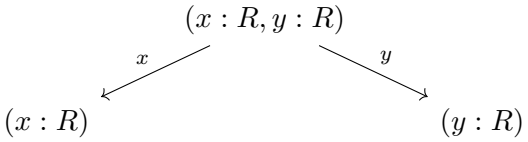
Example 1.33. Let us see what happens when we combine all the notation tricks. We can write $x : A, y : B(x) \vdash C(x, y)$: Set and it makes sense. Let me talk you through it: The expression on the left of the "$\vdash$" symbol denotes the stage $(\ ).A.B$

where $(\ ) \vdash A : \mathrm{Set}$ and $(\ ).A \vdash B : \mathrm{Set}$. The symbols x and y are local names for the variables of the internal sets A and B respectively. Following the notation conventions we have described in the paragraph above we can interpret the substring "(x)" in "$B(x)$" as the morphism $(t, x_A) : (\ ).A \to (\ ).A$. This map is exactly the identity map of $(\ ).A$, so $B(x)$ is just a different notation for B. Similarly "(x, y)" denotes the morphism $(t, \pi_B^* x_A, x_B) : (\ ).A.B \to (\ ).A.B$ which happens to be the identity of $(\ ).A.B$. Hence $C(x, y)$ is an alternative way to denote C, and we see that the judgement "$x : A, y : B(x) \vdash C(x, y) : \mathrm{Set}$" translates to $(\ ).A.B \vdash C : \mathrm{Set}$ in the variable free notation. ♢

Example 1.34. Let us look at a simple signature with dependent constants, and see if we can make sense of it using our notation tricks. What could be the meaning of the following judgements?

$$\begin{aligned} &\vdash R : \mathrm{Set} \\ &\vdash 0 : R \\ &\vdash 1 : R \\ x : R, y : R &\vdash x + y : R \\ x : R, y : R &\vdash x \cdot y : R \end{aligned}$$

The first judgement says that we have a fibered functor $R : y(\ ) \to \partial_1$. Up to reindexing choices this is the same thing as an object π_R in the slice category $\mathcal{S}/(\ ) = \mathcal{S}$. The judgements $\vdash 0 : R$ and $\vdash 1 : R$ tell us that we must specify two sections of the display map $\pi_R : (\ ).R \to (\ )$. This means we need two morphisms $0, 1 : (\ ) \to (\ ).R$ from the terminal object into $(\ ).R$. The expression "$x : R, y : R$" denotes the stage $(\ ).R.\pi_R^* R$ and the symbols x and y denotes the variables $\pi_R^* x_R$ and $x_{\pi_R^* R}$ respectively. Both variables are different sections of the display map $(x, y) : (x : R, y : R, z : R) \to (x : R, y : R)$ of the twice weakened set $x : R, y : R \vdash R : \mathrm{Set}$. It makes sense to read $x + y$ in $x : R, y : R \vdash x + y : R$ as $+(x, y)$ where (x, y) is the morphism $(x, y) : (x : R, y : R) \to (x : R, y : R)$ determined by lemma 1.32. That morphism is the identity, and so we see that $(\ ).R.R \vdash + : R$ should be a section of the display map $(x, y) : (x : R, y : R, z : R) \to (x : R, y : R)$. A section $+$ of the display map $(x, y) : (x : R, y : R, z : R) \to (x : R, y : R)$ contains exactly the same data as a context morphism $a : (x : R, y : R) \to (z : R)$. To see this note that any section $+$ gives us a morphism $a = (x + y)$ and any morphism a gives us a section $+ = (x, y, a(x, y))$ with the help of lemma 1.32. One can show with the help of the same lemma that the span below must be a product in $\mathcal{S}$.



This means that $x : R, y : R \vdash x + y : R$ denotes essentially a morphism $R \times R \to R$ in our base category $\mathcal{S}$. Similar reasoning applies to the interpretation of $x : R, y : R \vdash x \cdot y : R$. The signature of our example thus describes approximately the underlying data of a ring object in $\mathcal{S}$ in the categorical sense. The only extra data is reindexing information. ♢

Example 1.35. The last example showed us that we can get very pretty looking formulas for morphisms in $\mathcal{S}$ if we consequently use variables and omit reindexings along display maps. Even better, the composition of stage morphism in the base corresponds to substitution. If we work again in the signature of example 1.34, then we can write down the following stage morphisms.

$$(v, v+w, v\cdot w+1) : (v : R, w : R) \to (x : R, y : R, z : R)$$
$$(x\cdot(x+z), 1) : (x : R, y : R, z : R) \to (a : R, b : R)$$

What does the composite of those two morphisms look like? It turns out that we can compute it by substituting "v" for "x", "$v+w$" for "y" and "$v\cdot w+1$" for z in the expression $(x\cdot(x+z), 1)$. The resulting map is the following.

$$(v\cdot((v+w)+(v\cdot w+1)), 1) : (v : R, w : R) \to (a : R, b : R)$$

This makes sense if we think of the morphism $(x\cdot(x+z), 1) : (x : R, y : R, z : R) \to (a : R, b : R)$ as of the function which sends triples (x, y, z) in R^3 to pairs $(x\cdot(x+z), 1)$ in R^2. ♢

Example 1.36. We can also express that R is a ring. The axioms are the usual ones, we just write them down in first-order logic. As an example, we would want the following to be true at the empty stage.

$$(\ ) \Vdash \forall x, y, z : R.\, (x+y)\cdot z = x\cdot z + x\cdot z \tag{1.45}$$

It will turn out that the internal statement above holds precisely when the corresponding diagram in $\mathcal{S}$ commutes, provided that Set is 1-separated with respect to the topology W. ♢

Remark 1.37. Not only can the composite of morphisms between stage extensions be computed via substitution, also the reindexing of sets can be computed via substitution. Say for example we have a category depending on internal sets $x : A, y : B(x) \vdash C(x, y) : \mathrm{Cat}$ and a stage morphism $(M, N) : (z : D, w : E) \to (x : A, y : B)$ in the base $\mathcal{S}$. The variables z and w can appear freely in the terms M and N. The pull-back of C along (M, N) is exactly $z : D, w : E \vdash C(M, N) : \mathrm{Cat}$. We see that $(M, N)^* D = "D(x, y)"[M/x, N/y]$ in this case, and if we consequently use the variable notation for morphisms in the base then this will always work out. We will not prove it though, since that is a well-established fact in the semantics of dependent type theory[7], and because writing down a formal and rigorous proof is really hard (it is in general extremely hard to work mathematically rigorously with syntax). ♢

We will from now on use the notation introduced above and consequently build stages from the empty stage through stage extensions. The effect is that the difference between stage and context effectively disappear. The languages $L(Sign_\Gamma)$ merge together into one big internal language of the model $(\mathcal{S}, W)$. Remember that the languages $L(Sign_\Gamma)$ manipulate expressions of the following form.

$$\Gamma; \Xi \vdash \mathcal{J}$$

[7] To be fair, it is a well-established fact for many specific type theories, but of course not exactly for our type theory.

The two rules below relate stage and context.

$$\frac{\Gamma.A; \pi_A^*\Xi \vdash \pi_A^*\mathcal{J}}{\Gamma; x : A, \Xi[x/x_A] \vdash \mathcal{J}[x/x_A]} \qquad \frac{\Gamma; x : A, \Xi \vdash \mathcal{J}}{\Gamma.A; \pi_A^*\Xi[x_A/x] \vdash \pi_A^*\mathcal{J}[x_A/x]} \tag{1.46}$$

If we use all the notation tricks which we described in this section, then Γ will look just like an ordinary context, and the variable x_A will look and behave like a normal variable of dependent type theory. Since we also omit reindexing along display maps from the notation, the rules above become the following double rule.

$$\frac{\Gamma; x : A, \Xi \vdash \mathcal{J}}{\Gamma, x : A; \Xi \vdash \mathcal{J}}$$

What this means is that you may place the semicolon anywhere you like, as long as the context on the left consists of declarations of the form $x : A$ where A is a set. Let us try to make this preciseish.

Definition 1.38. The signature $Sign_{(\mathcal{S},W)}$ associated to a model $(\mathcal{S}, W)$ contains the following constant symbols.

(i) The constant symbols of the signature $Sign_{(\ )}$. Included are a constant symbol $A : \mathrm{Set}$ for each fibered functor $A : y(\ ) \to \partial_1$ and a constant symbol $N : A$ for each 2-cell $N : \mathbf{1}_{(\ )} \Rightarrow A$ in $\mathrm{Fib}_{\mathcal{S}}$.

(ii) A dependent constant $x : A \vdash C(x) : \mathrm{Cat}$ for each indexed category C at stage $(\ ).A$. To be absolutely clear: "$C(x)$" really means the string $concat$("C", "(x)"). The name C must appear together with the variable x in the syntax so that we can substitute terms for x and keep track of reindexings. The signature $Sign_{(\ ).A}$ of the last section is only in spirit, but not literally, a part of $Sign_{(\mathcal{S},W)}$.

(iii) The signature $Sign_{(\mathcal{S},W)}$ has dependent constants $x : A \vdash X(x) : C(x)$ and $x : A \vdash N(x) : C(x)_1(X(x), Y(x))$ for all objects X and morphisms N in the fiber $C(\mathrm{id}_{(\ ).A})$ of C.

(iv) Continue with the constant symbols depending on two variables, on three variables and so on and never stop.

The signatures $Sign_\Gamma$, defined in the first step, do not contain constant symbols which depend on types. They are not subsignatures of $Sign_{(\mathcal{S},W)}$. In contrast $Sign_{(\mathcal{S},W)}$ contains many constant symbols which depend on the points of internal sets. The recipe for interpreting statements of the language $L(Sign_{(\mathcal{S},W)})$ in $(\mathcal{S}, W)$ looks approximately as follows.

Definition 1.39. To interpret a judgement $\Xi \vdash N : M$ of the language $L(Sign_{(\mathcal{S},W)})$ in the model $(\mathcal{S}, W)$, go through the following steps.

(a) First divide the context Ξ into a small and a large context by replacing one of the commas by a semicolon. You can place it anywhere you like, as long as all the declarations to the left of it are of the form $x : A$ with $A : \mathrm{Set}$. We call the left half of the context the small context.

(b) Interpret the small context as a stage Γ of the model $(\mathcal{S}, W)$ by using the techniques described in this section. You may at some places already need the interpretation functions $[\![-]\!]_\Gamma$. Work yourself downwards one context extension at a time until you reach the semicolon.

(c) Replace all variables which have been declared in the small context by the appropriate constant symbols x_A at stage Γ and insert missing reindexings along display maps.

(d) Choose a valid list of values for the remaining variables in the large context to the right of the semicolon. The values should come from the signature $Sign_\Gamma$. Finally interpret the judgement by using the definition of $[\![-]\!]_\Gamma$ in the previous section.

Remark 1.40. To interpret the language $L(Sign)$ over an arbitrary signature, you can proceed as follows. First fix a model $(\mathcal{S}, W)$ and a morphism of signatures $m : Sign \to Sign_{(\mathcal{S},W)}$. Extend the morphism of signatures to a morphism of languages $L(m) : L(Sign) \to L(Sign_{(\mathcal{S},W)})$ and combine $L(m)$ with the interpretation function $[\![-]\!]_{(\mathcal{S},W)}$ whose definition is sketched above. The signature $Sign$ can contain category, object and morphism symbols depending on types $\mathrm{Set}(\mathbf{1}, A)$ since the same is true for the signature $Sign_{(\mathcal{S},W)}$. ♢

This completes the most tedious part of the thesis, namely the interpretation of the language in the strictified categorical models. We will now take a closer look at some parts of the language and their external meanings.

2 Stacks, Choice and Sets

The main topic of this chapter is the relation between stacks, prestacks and various choice principles in the internal world. We will add more rules to the language $L(Sign)$ and reduce the collections of models $(\mathcal{S}, W)$ a little. A main theme is that the external stack condition corresponds to a weak version of choice in the internal world. It allows us to choose data in an internal category which is determined up to unique isomorphisms. I have had some trouble turning this into a precise axiom which is general enough to be used in all situations, which is why we will see several versions of "1-definite choice" throughout this chapter.

2.1 Separation Conditions

Let us take a second look at the meaning of propositional equality. The definition of the forcing relation tells us that $\Gamma \Vdash N = M$ holds if and only if N and M are equal after passing to a cover of Γ. This does not necessarily imply that N and M are actually equal morphisms when they do not live in an indexed category which is 0-separated. Look at the following rule:

$$\frac{\Gamma \vdash N, M : C_1(X, Y) \qquad \Gamma \vdash P : N = M}{\Gamma \vdash N \equiv M : C_1(X, Y)}(ext) \tag{2.1}$$

This rule is the extensionality of propositional equality, and we very much want it to hold. Intuitively it says that when we prove that two morphisms are equal, then

they are actually equal. The object introduction rule (1.19) of the functor category, for example, is fairly useless if we can not use internal reasoning to produce the necessary judgmental identities.

Proposition 2.1. *An indexed category* $\Gamma \vdash C : \mathrm{Cat}$ *is 0-separated if and only if the rule* (ext) *is sound for* C *and all of its reindexings.*

Proof. The extensionality principle states that morphisms which are equal on a cover are actually equal. This is the case if and only if the functors to descent data are faithful, which is the definition of being 0-separated. □

The extensionality rule is the reason why our internal logic does not include equations $X = Y : \mathrm{Prop}$ of objects $X, Y : C_0$. We could without problem add propositional equations of objects together with the forcing condition "$\Gamma \Vdash X = Y : \mathrm{Prop}$ if and only if there is a cover $u_i : \Delta_i \to \Gamma$ on which X and Y become equal" to the language, and the internal logic would still be sound. But there are hardly any indexed categories C for which this equality is extensional. It does not even work for the indexed category $\mathrm{Set} = \mathrm{Sp}(\partial_1)$. An internal set $\Gamma \vdash A : \mathrm{Set}$ is an object π_A in the slice $\mathcal{S}/\Gamma$ equipped with choices of pullbacks along morphisms $u : \Delta \to \Gamma$. Two internal sets B, A at the same level Γ can be exactly equal after restriction to a cover and still not be exactly equal at stage Γ. The biggest class of indexed categories for which the propositional equality of objects would definitely be extensional are the discrete indexed categories.

To find out what 1-separatedness means from the internal perspective, let us look at the following internal statement.

$$\Gamma \Vdash \exists! f : C_1(X, Y).\, \phi \tag{2.2}$$

Here ϕ is some well-formed formula $\Gamma, f : C_1(X, Y) \vdash \phi : \mathrm{Prop}$ which may contain f as a free variable. Let us assume that C is already 0-separated. By using the definition of the forcing relation we can compute the meaning of (2.2). The result is: "There is a cover $u_i : \Delta_i \to \Gamma$ such that for each i there is a unique constant $\Delta_i \vdash f_i : u_\alpha^* C_1(u_i^* X, u_i^* Y)$ for which $\Delta_i \Vdash u_i^* \phi[f_i/f]$ holds and the same stays true at all lower stages." As with propositional equations, the morphism f only exists on a cover and not at the stage Γ itself. The morphisms f_i assemble into a morphism of decent data. That they agree on overlaps follows from the uniqueness part of "$\exists! f : C_1(x, y).\phi$". A morphism of descent data can be glued to a morphism at stage Γ when the category C is 1-separated. This means that we can soundly interpret the following rules when C is 1-separated.

$$\frac{\Xi, f : C_1(X, Y) \vdash \phi : \mathrm{Prop} \qquad \Xi \vdash P : \exists! f : C_1(X, Y).\phi}{\Xi \vdash \imath f.\phi : C_1(X, Y)}$$

$$\frac{\Xi \vdash, f : C_1(X, Y) \vdash \phi : \mathrm{Prop} \quad \Xi \vdash P : \exists! f : C_1(X, Y).\phi}{\Xi \Vdash \phi[\imath f.\phi/\phi]} \tag{2.3}$$

The operator $\imath$ is the iota-operator of definite description. The idea is that the term "$\imath f.\phi$" names the unique morphism f for which $\phi(f)$ holds. Our discussion above shows the following.

Proposition 2.2. [42, Theorem 7.9] *When an indexed category* $\Gamma \vdash C : \mathrm{Cat}$ *is 1-separated then* C *and all of its reindexings satisfy the principle of definite description for morphisms (2.3).*

We want that all our categories satisfy definite description. Hence we will only consider indexed categories which are prestacks relative to the site $(\mathcal{S}, W)$. This means that we also have to check that all the category constructors listed in (1.29) produce prestacks. We do this in appendix A and B.

Definition 2.3. We add the extensionality of equality and definite description to the rules of the language $L(Sign)$. We remove all category constants from $Sign_{(\mathcal{S},W)}$ which do not name prestacks. We restrict our models to those sites $(\mathcal{S}, W)$ for which the self-indexing of $\mathcal{S}$ is a prestack[8].

We also want to find out what the internal meaning of 2-separatedness is. Here it is best to start with an example.

Example 2.4. Assume that C is defined at the global stage and assume, as we always do from now on, that C is 1-separated. Let us look at the internal statement that C has binary products.

$$\forall x : C_0.\forall y : C_0.\exists p : C_0.\exists \pi : C_1(p,x).\exists \pi' : C_1(p,y).\ulcorner x \leftarrow p \rightarrow y \textit{ is a product}\urcorner$$

The Quine-corners $\ulcorner ... \urcorner$ indicate that the informal sentence in between should be translated into a formal sentence by the reader. We may remove the outer most block of $\forall$-quantifiers first. For each object Γ in the base and for all constants $\Gamma \vdash x, y : C_0$ the following must hold.

$$\Gamma \Vdash \exists p : C_0.\exists \pi : C_1(p,x).\exists \pi' : C_1(p,y).\ulcorner x \leftarrow p \rightarrow y \textit{ is a product}\urcorner$$

Reindexings along the unique map $t_\Gamma : \Gamma \rightarrow (\ )$ are omitted from the notation. We may next remove the block of existential quantifiers. The forcing condition says that there is a cover $u_i : \Delta_i \rightarrow \Gamma$ with spans $u_i^* x \leftarrow p_i \rightarrow u_i^* y$ in the Δ_ith fiber such that

$$\Delta_i \Vdash \ulcorner u_i^* x \leftarrow p_i \rightarrow u_i^* y \textit{ is a product}\urcorner$$

holds for each i. One can check, using that C is a prestack, that this is the case if and only if the $u_i^* y \leftarrow p_i \rightarrow u_i^* y$ are actually product diagrams in the corresponding fibres of C which are additionally stable under further reindexings. We thus see that the translation of the internal statement "C has binary products" is that C has reindexing-stable products of objects $\Gamma \vdash x, y : C_0$, but only after passing to a cover $u_i : \Delta_i \rightarrow \Gamma$ of Γ. Since the universal property of the products on the cover is reindexing stable we can compare them on the intersections Δ_{ij}. There are comparison isomorphisms $\theta_{ij} : p_i|_{\Delta_{ij}} \cong p_j|_{\Delta_{ij}}$, and the uniqueness clause in the universal property of a product implies that they satisfy the cocycle condition. The p_i together with the comparison isomorphisms become an object $\{p_i, \theta_{ij}\}$ in the category of descent data for the cover $u_i : \Delta_i \rightarrow \Gamma$. We can glue the descent data to an actual product at level Γ when the indexed category C is a stack. This

[8]Warning: This definition will be subject to further change up until the end of the chapter 2.

shows that the internal statement "C has binary products" translates to the external statement "C has fiberwise products which are preserved by reindexing" when C is a stack. ♢

Corollary 2.5. *Assume that $C : \mathrm{Cat}$ is a stack. The following are equivalent:*

(i) "C has binary products" holds internally.

(ii) The diagonal $\Delta_C : C \to C \times C$ in $[\mathcal{S}^{op}, \mathrm{Cat}]$ has a right adjoint as a 1-cell $J\Delta_C$ in $\mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})$.

Proof. We have computed that "C has binary products" holds internally if and only if each fiber of C has reindexing stable binary products. One can use those fiberwise products to define the components of a right adjoint 1-cell of $\Delta_C : C \times C \to C$, but there is no guarantee that the choices of products in each fiber will commute strictly with the reindexings. The consequence is that we only get a 1-cell in $\mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})$, and it need not be the case that this 1-cell can be replaced by an isomorphic strict one. □

On the other hand let us think about what it could mean to internally choose binary products for C. One way to express that "C comes equipped with a choice of products" is that there should be an internal functor $\mathrm{prod}_C : (C^{C\times C})_0$ together with an internal transformation $\varepsilon_C : ((C \times C)^{C\times C})_1(\Delta_C \,\mathrm{prod}_C, 1_{C\times C})$ such that "$(\mathrm{prod}_C, \varepsilon_C)$ is a right adjoint of the diagonal Δ_C" holds internally. If $C : \mathrm{Cat}$ comes equipped with such data, then we can use it to define actual product constructors. We could make sense of the following rules without having to add anything new to our language.

$$\frac{\Xi \vdash X, Y : C_0}{\Xi \vdash X \times Y : C_0} \qquad \frac{\Xi \vdash X, Y : C_0}{\Xi \vdash \pi_{X,Y} : C_1(X \times Y, X)}$$

$$\frac{\Xi \vdash X, Y : C_0}{\Xi \vdash \pi'_{X,Y} : C_1(X \times Y, X)} \qquad \frac{\Xi \vdash N : C_1(Z, X) \qquad \Xi \vdash M : C_1(Z, Y)}{\Xi \vdash (N, M) : C_1(Z, X \times Y)} \tag{2.4}$$

We would understand $\Xi \vdash X \times Y : C_0$ as a shorthand for the longer term $\Xi \vdash \mathrm{prod}_C(X, Y) : C_0$. We would interpret $\Xi \vdash \pi_{X,Y} : C_1(X \times Y, X)$ and $\Xi \vdash \pi'_{X,Y} : C_1(X \times Y, Y)$ as the two components $\Xi \vdash \varepsilon_C(X, Y)_1 : C_1(X \times Y, X)$ and $\Xi \vdash \varepsilon_C(X, Y)_2 : C_1(X \times Y, Y)$ of the counit, and we would understand $\Xi \vdash (N, M) : C_1(Z, X \times Y)$ as an abbreviation for the longer term $\Xi \vdash \imath f.(\pi_{X,Y} \circ f = N \wedge \pi'_{X,Y} \circ f = M) : C_1(Z, X \times Y)$. All the rules (2.4) would be satisfied, and one could also derive appropriate computation rules and a uniqueness principle. Conversely, if we had a sound interpretation of the rules (2.4) then we could use them together with the functor introduction rule to internally construct a right adjoint of Δ_C at the global stage. Hence we see that an internal right adjoint $(\mathrm{prod}_C, \varepsilon_C)$ of the diagonal is exactly the data which we would need if we wanted to add a binary product type constructor for C to our language.

An "internal choice of binary products" for the indexed category $C : \mathrm{Cat}$ is thus the same thing as a right adjoint of Δ_C in the 2-category $[\mathcal{S}^{op}, \mathrm{Cat}]$. The stack condition

unfortunately only produces a right adjoint 1-cell of $J\Delta_C$ in $\mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})$. This is the place where semicoflexibility becomes useful. Remember that C is semicoflexible if and only if every pseudo-cell which points into C can be replaced by an isomorphic strict one.

Definition 2.6. We say that an indexed category satisfies *1-definite choice* if it is a semicoflexible stack.

Those are precisely the categories for which one can expect that the mere propositional existence of some universal construction implies the actual presence of a corresponding constructor. We will make this much more precise in the next section. We do not demand that all our indexed categories are semicoflexible stacks. One does not expect in constructive mathematics that choice principles always hold, so this is fine from the internal perspective. It also forced upon us by the external reality, because some of the category constructors which we introduced in (1.29) just will not produce semicoflexible stacks[9]. We want that at least the category Set : Cat satisfies 1-definite choice, and we thus shrink the collection of models again.

Definition 2.7. A model is a site $(\mathcal{S}, W)$ such that $\mathcal{S}$ has finite limits and the self-indexing is a stack. This definition will not change anymore from now on.

Included in our revised notion of a model are pretopoi with the coherent topology, elementary topoi with the coherent topology, the tautological model $(\mathrm{Set}_\lambda, can)$, Grothendieck topoi $\mathrm{Sh}_\lambda(C, J)$ with the canonical topology or the coherent topology and any lex category $\mathcal{S}$ with the codiscrete topology. The last one feels slightly off. The requirement that the self-indexing be a stack prevents our topology from being too fine, but it does not prevent it from being very coarse. What will prevent the topology from being too coarse are the various "subsingelton types as propositions" principle which we will meet in chapter 4. On the other hand, it is very advantageous that lex categories with the codiscrete topology are included in our class of models, because it means that our language will be able to reproduce results of fibered category theory under minimal assumptions on the base.

2.2 1-Definite Choice

When we internally show that something exists $\Gamma \vdash P : \exists x : A.\phi$ then this does not guarantee that we can construct an actual term $\Gamma \vdash M : A$ together with a proof $\Gamma \vdash Q : \phi[M/x]$. Such a term does not even need to exist in a given model of our language. The forcing relation only tells us that we get terms after passing to a cover of Γ. The lack of an actual term $\Gamma \vdash M : A$ can also be understood from a constructive perspective. A proof term $P : \exists x : A.\phi$ encodes nothing more than the mere knowledge that some construction procedure exists which produces a term $N : A$ which satisfies $\phi(x)$. All knowledge about the actual construction is lost. All we can do with that knowledge is producing new mere knowledge that

[9]For example, the externalisation $[C] : \mathrm{Cat}$ of an internal category $C : \mathrm{sCat}_0$ is typically not a stack. The typical failure of internal small categories to be stacks, and thus to satisfy 1-definite choice, is discussed in detail in [10], and it is also the underlying reason behind the warning in the last paragraph of [19, section 7.2].

some other construction is possible. There is no way to extract actual information about the construction from the proof term P. This is a consequence of our decision to work proof irrelevant. Constructivists often call existence in the propositional sense "mere existence", and they reserve the phrases "there exists" or "there is" for situations in which they have access to actual data which validates $\exists x : A.\varphi$. We will not follow that convention, but the difference between existence in the propositional truncated sense and actual existence will nonetheless haunt us throughout the rest of the thesis. Both 1-separatedness and (2-separatedness + semicoflexibility) are conditions on an internal category C which allow us to bridge the gap between mere existence and actual data. 1-separatedness allows us to produce an actual term $\imath f.\phi$ for a morphism which merely exists, and (2-separatedness + semicoflexibility) tells us that if a construction which is determined up to a 1-contractible choice merely is possible, then there is an actual constructor. Even 0-separatedness can be seen in that light: It allows us to lift the mere equality of two morphisms from a cover to an actual equality at the stage we are at.

The difference between mere and actual existence is much more relevant in constructive mathematics than in classical mathematics, because constructive mathematics lacks strong choice principles. It is what makes constructive category theory a bit different from classical category theory at some places. One of the places where this is especially apparent is the classical characterisation of equivalences.

Example 2.8. Assume we have a term $\Gamma \vdash F : (D^C)_0$ at some stage Γ. Here are three different ways in which F can be an equivalence, ordered by increasing strength.

(a) F can be fully faithful and merely essentially surjective on objects from the internal perspective. In the formal language this looks as follows.

$$\begin{aligned}\Gamma \Vdash (\forall x, y : C_0.\forall g : D_1(F_0x, F_0y).\exists! h : C_1(x, y).F_1h = g)\\ \wedge (\forall d : D_0.\exists c : C_0.\exists \alpha : D_1(F_0c, d).\ulcorner \alpha \textit{ is an isomorphism}\urcorner)\end{aligned} \tag{2.5}$$

(b) It can be the case that F merely has an inverse. This can also be expressed formally.

$$\begin{aligned}\Gamma \Vdash \exists G : (C^D)_0.\, \exists \alpha : (C^C)_1(1_C, GF).\exists \beta : (D^D)_1(FG, 1_D).\\ \ulcorner \alpha \textit{ and } \beta \textit{ are isomorphisms}\urcorner\end{aligned} \tag{2.6}$$

(c) F might actually have an inverse. There might be terms $\Gamma \vdash G : (C^D)_0$, $\Xi \vdash \alpha : (C^C)_1(1_C, GF)$ and $\Xi \vdash \beta : (D^D)_1(FG, 1_D)$ together with a proof

$$\Gamma \Vdash \ulcorner \alpha \textit{ and } \beta \textit{ are isomorphisms}\urcorner \tag{2.7}$$

which expresses internally that the data (F, G, α, β) is an equivalence of categories.

$\diamondsuit$

Part (c) can never be a mere property of F because G, α and β are extra data. If we wanted to "prove" (c) internally, then we would have to build a large type of inverses

equivalences of F and inhabit that type. Our language $L(Sign)$ unfortunately lacks constructors which would allow us to build that large type of inverse. The "proof" would also not be a proof in the sense that non-type theorists understand the word, since it itself would contain data. The only way to turn "F actually has an inverse" into a mere proposition is by cheating and doing the last $\exists$-quantification in the meta-language. We are then necessarily working half-internally, half-externally, which can be confusing.

We can not consequently work with existence in the sense of (c), because our language lacks the constructors which would allow us to build the very large types which we would need for that.

Example 2.9. Say we want to "prove" that any category with a pullback constructor and a chosen terminal object can be equipped with an equalizer constructor. In type theory one would do this by building the very large type of categories equipped with pullbacks and a terminal object and the very large type of categories equipped with equalizers respectively. A "proof" would then be a term

$$P : (\textit{Categories with pullbacks and 1}) \to (\textit{Categories with equalizers}) \tag{2.8}$$

which preserves the underlying categories. The term P contains data, namely an actual construction recipe. Its type is not a mere proposition. We are not able to build very large types of categories equipped with extra structure, and we will thus have to be satisfied with either the propositional truncated versions of such a proof, or alternatively with statements which are only half-internal. Working with mere propositions is not as bad as it sounds at first, because 1-definite choice will make sure that we can obtain actual data at the end, provided that the data in question is unique enough. ♢

Part (a) and (b) of example (2.8) are classically equivalent. The classical proof starts as follows: "For each $d : D_0$ choose some $c_d : C_0$ together with an isomorphism $\alpha_d : F_0 c_d \to d$." We see immediately why that proof does not work in a constructive setting: we can not choose. It is interesting to translate (a), (b) and (c) of (2.8) via the forcing semantics to see what their external meaning is. We assume as always that C and D are already 1-separated. Part (c) holds if and only if F is an actual equivalence in the 2-category $[(\mathcal{S}/\Gamma)^{op}, \mathrm{Cat}]$. Part (b) holds if and only if there is a cover $u_i : \Delta_i \to \Gamma$ such that the reindexings $u_i^* F$ are actual equivalences. Part (a) holds if and only if each component of F is a fully faithful functor, and for each y in some fiber $D(u)$ of D there is a cover $v_i : uv_i \to u$ and objects $x_i \in C(uv_i)$ such that $F(x_i)$ is isomorphic to $v_i^* u^* y$ in $D(uv_i)$. All those properties are genuinely different, but the hierarchy collapses if C is a semicoflexible stack, because in that case (a) implies (c).

Proposition 2.10. *If* $\Gamma \vdash C : \mathrm{Cat}$ *satisfies 1-definite choice, then (a) and (b) are internally equivalent. Formally:*

$$\begin{aligned}\Gamma \Vdash \forall D : \mathrm{Cat}\,.\forall F : (D^C)_0.&\\ &(\ulcorner F \textit{ is essentially surjective on objects and fully faithful}\urcorner\\ &\to \ulcorner F \textit{ has an inverse}\urcorner)\end{aligned} \tag{2.9}$$

Proof. We have to show that given any functor term $\Delta \vdash F : (D^{u^*C})_0$ at any lower stage $u : \Delta \to \Gamma$, if (a) holds then so does (b). The indexed category $\Delta \vdash u^*C$ is still a semicoflexible stack since both properties are reindexing stable. The translation of part (a) tells us that the functor F is fiberwise fully faithful and essentially hits all objects in the fibers of D at least after passing to a cover. We can define an inverse G of F as follows. Given some object $d \in D(\mathrm{id}_\Delta)$ we choose a cover $v_i : \Theta_i \to \Delta$ and objects $c_i \in v_i^* u^* C(\mathrm{id}_{\Theta_i})$ together with isomorphisms $\alpha_i : Fc_i \to v_i^* d$. The isomorphisms α_i together with the fiberwise fully faithfulness of F allow us to compare the c_i at the intersections Θ_{ij} of the cover. We get descent data (c_i, θ_{ij}) which we can glue to an object c at level Δ since u^*C is a stack. The isomorphisms α_i allow us to define an isomorphism of descent data $(v_i^* Fc, \mathrm{id}) \cong (Fc_i, F\theta_{ij}) \to (v_i^* d, \mathrm{id})$ in D, which glues to an isomorphism $\alpha : Fc \to d$ because D is a prestack. We can now use the pairs (c, α_d) to define the id_Δth fiber of the inverse of G, and all lower fibers can be constructed in a similar way. We get an inverse equivalence of F in $\mathrm{Hom}((\mathcal{S}/\Delta)^{op}, \mathrm{Cat})$ which can be strictified because u^*C is semicoflexible. Since F actually has an inverse it is in particular internally true that F merely has an inverse. □

Remark 2.11. The annoying difference between mere existence and actual data is not just an artefact of the short-comings of constructive mathematics, it expresses a genuine phenomenon in the 2-categories of stacks and prestacks. The paper [10] studies "various notions of equivalences of indexed categories and how they relate to Giraurd's stacks". The definition (1.1) of a "weak equivalence functor" $F : A \to B$ in that paper is exactly the external translation of the internal statement "F is fully faithful and essentially surjective on objects", provided that A and B are already prestacks. The definition (1.8) of "strongly equivalent indexed categories" in [10] corresponds modulo strictification exactly to part (c) of example 2.8. The paper [10] also contains a definition (1.9) of "locally equivalent indexed categories" where two indexed categories A and B are locally equivalent if and only if there is a cover $\Delta_i \to (\ )$ such that $A|_{\Delta_i}$ and $B|_{\Delta_i}$ are actually equivalent for each i. This can also be nicely expressed in our internal language. Two prestacks $A, B : \mathrm{Cat}$ are locally equivalent in the sense of [10] if and only if it is internally true that A and B are merely equivalent.

$$(\ ) \Vdash \exists F : (B^A)_0 . \exists G : (A^B)_0 . \exists \alpha : (B^B)_1(FG, 1_B) . \exists \beta : (A^A)_1(1_A, GF) . \ulcorner \alpha \textit{ and } \beta \textit{ are isomorphisms} \urcorner \tag{2.10}$$

This is even weaker than (a) of example 2.8 since we do not have an actual internal functor pointing in either direction. ♢

Example 2.12. The lack of choice is also noticeable when we look at internal adjunctions. Let us assume that we have an internal functor $\Gamma \vdash U : (D^C)_0$. Here are three different ways in which U can have a left adjoint, ordered by increasing strength:

(a) One characterisation goes via universal elements. It looks as follows.

$$\Gamma \Vdash \forall d : D_0 . \exists c : C_0 . \exists \eta : D_1(d, U_0 c) . \ulcorner (\eta, c) \textit{ is initial in } d/U \urcorner \tag{2.11}$$

We will sometimes say that "U pointwise has a left adjoint solution" in an informal argument to mean the above.

(b) It could be the case that U merely has a left adjoint.

$$\Gamma \Vdash \exists F : (C^D)_0.\exists \eta : (D^D)_1(1_D, FU). \\ \ulcorner (F,\eta) \textit{ satisfy the universal property of a left adjoint} \urcorner \tag{2.12}$$

(c) It could also be the case that there are actual terms $\Gamma \vdash F : (C^D)_0$ and $\Gamma \vdash \eta : (C^C)_1(1_C, UF)$ such that (b) becomes true when we insert them. This can never be a mere property of U, at least not internally.

In classical category theory (a) and (b) are equivalent. The classical proof starts with "For each $d : D_0$ choose an initial object (Fd, η_d) in d/U...". The proof fails in constructive mathematics because one can not choose. Conditions (a), (b) and (c) are genuinely different in our internal language and in a general model, except when C satisfies 1-definite choice. ♢

Proposition 2.13. *Assume that $\Theta \vdash C : \mathrm{Cat}$ is a stack and that $\Theta \vdash U : (D^C)_0$ is an internal functor. The following are equivalent:*

(i) It holds internally that "all the comma-categories d/U have an initial object".

(ii) Each component of U has a left adjoint and the left adjoints satisfy the Beck-Chevalley condition.

(iii) JU has a left adjoint in the 2-category $\mathrm{Hom}((\mathcal{S}/\Theta)^{op}, \mathrm{Cat})$.

Proof. Assume without loss of generality that Θ is the empty stage. If not then we replace the model $(\mathcal{S}, W)$ by $\mathcal{S}/\Theta$ with the induced topology. We compute the external meaning of the internal statement (i) by using the forcing semantics. We remove the quantifiers block by block. The first translation step is that for each stage Γ and for each constant $\Gamma \vdash d : D_0$ the following must hold.

$$\Gamma \Vdash \exists c : C_0.\exists \eta : D_1(d, U_0 c).\forall x : C_0.\forall g : D_1(d, U_0 x).\exists! f : C_1(c, x).(U_1 f) \circ \eta = g \tag{2.13}$$

We may find a cover $v_i : \Delta_i \to \Gamma$ and terms $\Delta_i \vdash c_i : C_0$, $\Delta_i \vdash \eta_i : D_1(v_i^* d, U_0 c_i)$ such that the following holds for each i.

$$\Delta_i \Vdash \forall x : C_0.\forall g : D_1(v_i^* d, U_0 x).\exists! f : C_1(c_i, x).U_1 f \circ \eta_i = g \tag{2.14}$$

The external meaning of (2.14) is that the pair (c_i, η_i) is initial among all such pairs at the given stage, and that the same stays true after reindexing to lower stages. One has to use definite description and the extensionality of the propositional equality to arrive at that translation of (2.14). In particular, we have to assume that D is already 1-separated. The reindexing stability of the universal property of the pairs (c_i, η_i) means that we can compare the c_i after restricting them to the overlaps Δ_{ij} of our cover $u_i : \Delta_i \to \Gamma$. We get isomorphisms $\theta_{ij} : c_i|_{\Delta_{ij}} \to c_j|_{\Delta_{ij}}$ which will satisfy the cocycle condition because of the uniqueness clause in the universal property.

The η_i assemble into a morphism of descent data $(\eta_i) : (u_i^* d, \mathrm{id}) \to (Uc_i, \theta_{ij})$. We can glue the (c_i, θ_{ij}) to an object at the level Γ because C is a stack. The fact that D is a prestack means that we can lift the morphism (η_i) of descent data to a morphism $\eta : d \to Fc$ at the level Γ. The pair (c, η) will satisfy the correct universal property because it does so on a cover and truth in the stack semantics is local. The universal pairs (c_d, η_d), where d ranges over the objects of D_Γ, allow us to define a left adjoint F_Γ of the component U_Γ of U. The fact that the universal property of the pairs (c_d, η_d) is reindexing stables implies that the F_Γ satisfy the usual Beck-Chevalley condition. The Beck-Chevalley condition is exactly what is needed to assemble the component-wise left adjoints $F_\Gamma \dashv U_\Gamma$ to a left adjoint 1-cell of U in $\mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})$. □

Let us next look at the following set of rules which introduce a left adjoint whenever a functor merely pointwise has a left adjoint solution.

$$\frac{\vdash U : D^C \qquad \Vdash \ulcorner d/U \textit{ has an initial object for all } d : D_0 \urcorner}{\vdash \mathbf{L}_U : (C^D)_0}$$

$$\frac{\textit{same assumptions}}{\vdash \eta_U : D^D(1_D, U(\mathbf{L}_U))} \qquad \frac{\textit{same assumptions}}{\Vdash \ulcorner (\mathbf{L}_U, \eta_U) \textit{ is a left adjoint of } U \urcorner} \tag{2.15}$$

Definition 2.14. We add the rules (2.15) to the language $L(Sign)$. The rules can only be applied in the empty context and if C is a semicoflexible stack. We also add a constructor $(\mathbf{R}, \varepsilon)$ to our language which constructs right adjoints in the empty context under analogues conditions.

The interpretation of the terms $\mathbf{L}_U$ and η_U has already been indicated. Whenever we stumble upon a term $\mathbf{L}_U$, then it must be the case that the interpretation of U actually has a left adjoint 1-cell as a 1-cell JU in $\mathrm{Hom}((\mathcal{S}/(\ ))^{op}, \mathrm{Cat})$. Since the codomain is semicoflexible we can strictify the left adjoint of JU and get a left adjoint 1-cell of U in $[(\mathcal{S}/(\ ))^{op}, \mathrm{Cat}]$. We choose one such left adjoint 1-cell together with a unit and take them to be the interpretation of the terms $\mathbf{L}_U$ and η_U. We need to remember that choice and make the same one whenever $\mathbf{L}_U$ and η_U appear again in the translation process.

A constructor which can only be applied in the empty context is always a bit unsatisfying. Let me show you the coherence problem that we would have to solve if we would want to use $(\mathbf{L}, \eta)$ in non-empty contexts. Say, for example, we have a family of functors $n : \mathbb{N} \vdash U : (D^C)_0$ depending on internal natural numbers $n : \mathbb{N}$. Assume it holds internally that each $U(n)$ pointwise has a left adjoint and that C satisfies 1-definite choice. If we were allowed to use $(\mathbf{L}, \eta)$ in non-empty contexts, then we could form a new term $n : \mathbb{N} \vdash \mathbf{L}_U : (C^D)_0$. A specific internal natural number such as $3 : \mathbb{N}$ gives us a stage morphism $3 : (\ ) \to (n : \mathbb{N})$. We can reindex $\mathbf{L}_U$ along that stage morphism to get a term $3^*\mathbf{L}_U : (C[3/n]^{D[3/n]})_0$ in the empty context. On the other hand, it is also internally true that $U[3/n]$ pointwise has a left adjoint solution, since the truth of statements in the internal language is reindexing stable. Hence we can also derive the judgement $\mathbf{L}_{U[3/n]} : C[3/n]^{D[3/n]}$. Ideally there should be a strict equality $3^*\mathbf{L}_U = \mathbf{L}_{U[3/n]}$ because reindexing in

the semantics should correspond exactly to substitution in the syntax. However, there is no reason why the interpretation of $3^*\mathbf{L}_U$ should be exactly equal to the interpretation of $\mathbf{L}_{U[3/n]}$ if our interpretation recipe for the $\mathbf{L}$-constructor is not more sophisticated than choosing a random left adjoint 1-cell whenever a term $\mathbf{L}_U$ appears. The two functors will only be canonical isomorphic. If we want to use $(\mathbf{L}, \eta)$ in non-empty contexts, then we have to find a way to choose left adjoints at all stages at once in a way which is strictly compatible with reindexings[10]. If we restrict the use of $(\mathbf{L}, \eta)$ to the empty stage then the coherence problem can never appear, because there is no way to substitute something in $\mathbf{L}_U$ when U does not contain free variables.

Remark 2.15. A different way to express the 1-definite choice principle would be the following set of rules.

$$\frac{\Xi \vdash C : \mathrm{Cat} \qquad \Xi \Vdash \ulcorner C \text{ is contractible} \urcorner}{\Xi \vdash \imath_1 C : C_0} \qquad \frac{\Xi \vdash \imath_1 C : C_0}{\Xi \Vdash \ulcorner C \text{ is contractible} \urcorner} \tag{2.16}$$

The idea is that $\imath_1 C : C_0$ denotes "the" object of the contractible category C. The rules above can, in combination with enough category constructors, also be used to produce actual adjoints. We could, given $U : (C^D)_0$ and $d : D_0$, cut out the full subcategory of initial objects in d/U and choose one for each d by using the 1-definite description operator $\imath_1$. I have, unfortunately, not found a way to give a sound interpretation of the rules (2.16) above. The result below, though, expresses a similar idea as a mere internal proposition. ♢

Proposition 2.16. *Assume* $\Gamma \vdash A : \mathrm{Set}_0$ *is an internal set and* $\Gamma, x : A \vdash C_x : \mathrm{Cat}$ *is a stack at level* $\Gamma.A$*. Then the following holds internally.*

$$\Gamma \Vdash (\forall x : A. \ulcorner C_x \text{ is contractible} \urcorner) \to \ulcorner \textstyle\prod_{x:A} C_x \text{ is contractible} \urcorner \tag{2.17}$$

Proof. If $\Gamma \Vdash \forall x : A. \ulcorner C_x$ *is contractible*$\urcorner$ holds, then this means that there is a cover of $u_i : \Delta_i \to \Gamma.A$ such that the generic fiber of C and all of its lower fibers become actually contractible when restricted to that cover. That means there are terms $\Delta_i \vdash c_i : (Cu_i)_0$ such that every other object in that fiber is isomorphic to c_i in a unique way, and the same stays true at lower stages. We may glue the c_i to an object $\Gamma.A \vdash c : C_0$ at stage $\Gamma.A$ because $\Gamma.A \vdash C : \mathrm{Cat}$ is a stack by assumption. An term $\Gamma, x : A \vdash c_x : (C_x)_0$ is the same thing as a term $\Gamma \vdash (c_x)_{x:A} : \prod_{x:A} C_x$ and hence we see that $\Gamma \vdash \prod_{x:A} C_x : \mathrm{Cat}$ is actually, and thus in particular merely, contractible. □

2.3 Every Set is the Coproduct of its Points

This section introduces a new axiom about the category of internal sets, which has a lot of useful consequences. The results of this section are in no relation to the choice principles of the last section, but we need the results in the next chapter, which is why we discuss them here. The axiom holds in every model $(\mathcal{S}, W)$ of our language.

[10] I have not been able to do that. I suspect that one might have to strictify the 2-category $\mathrm{St}(\mathcal{S}, W)$ of stacks as a whole, and not just its 0-cells as we did.

Axiom 1 (Internal)**.** *Every set is the coproduct of its points. More formally:*

$$\forall A, B : \mathrm{Set}_0.\forall b : (\textstyle\prod_{x:A}\mathrm{Set})_1((\mathbf{1})_{x:A}, (B)_{x:A}).$$
$$\exists! f : \mathrm{Set}_1(A, B).\forall x : A.f \circ x = b_x \qquad (2.18)$$

The axiom, together with 1-definite description, allows us to derive the following rule.

$$\frac{\Xi \vdash A, B : \mathrm{Set}_0 \qquad \Xi, x : A \vdash N : B}{\Xi \vdash \nu x.N : \mathrm{Set}_1(A, B)} \qquad (2.19)$$

The term "$\nu x.N$" should be understood as a shorthand for the more lengthy expression "the unique map f such that $(f \circ x)_{x:A} = (N)_{x:A}$". Or more formally:

$$\Xi \vdash \nu x.N :\equiv \imath f.(\forall x : A.f \circ x = N) : \mathrm{Set}_1(A, B) \qquad (2.20)$$

The informal meaning of axiom 1 and the ν-abstraction rule is that in order to define a function $f : A \to B$ of internal sets it is enough to describe its action on points $x : \mathrm{Set}(\mathbf{1}, A)$ of A. The ν-abstraction rule is, even though it looks similar, different from the usual λ-abstraction of type theory. One difference is for example that $\mathrm{Set}_1(A, B)$ is not itself a set, which prevents us from stacking ν-abstractions.

There is also a rule in the other direction.

$$\frac{\Xi \vdash f : \mathrm{Set}_1(A, B) \qquad \Xi \vdash a : A}{\Xi \vdash f(a) : B} \qquad (2.21)$$

We read "$f(a)$" as a short-hand for "$f \circ a$" so that the rule above is just an instance of the composition rule. One can check that $\nu x.(f(x)) \equiv f$ and $(\nu x.N)(L) \equiv N[L/x]$ hold judgementally.

Proposition 2.17. *Every model* $(\mathcal{S}, W)$ *satisfies axiom 1.*

Proof. Let us translate the statement via the stack semantics. If we remove the first block of universal quantifiers then it says that for each stage Γ, sets $\Gamma \vdash A, B : \mathrm{Set}_0$ and term $\Gamma, x : A \vdash b_x : B$ the following holds.

$$\Gamma \Vdash \exists! f : \mathrm{Set}_1(A, B).\, (b_x)_{x:A} = (f)_{x:A} \circ (x)_{x:A} \qquad (2.22)$$

We can translate (2.22) by using that Set is a prestack: The morphism f exists at the current stage, it is unique and the same stays true at lower stages. The overall translation looks as follows: "For all stages Γ, internal sets $\Gamma \vdash A, B : \mathrm{Set}_0$ and terms $\Gamma.A \vdash b : B$ there is a unique $\Gamma \vdash f : \mathrm{Set}_1(A, B)$ such that $b = \pi_A^* f \circ x_A$." This is a general fact which holds without any assumptions on $\mathcal{S}$. We have proven it in lemma 1.30. □

Theorem 2.18. *Axiom 1 has the following internal consequences:*

(i) A function is determined by its action on points. If $f, g : A \to B$ are two functions such that $f(x) = g(x)$ for each point $x : A$, then f and g are equal.

(ii) A surjective monomorphism is an isomorphism in Set*.*

(iii) A surjective function is an extremal epimorphism.

(iv) The monomorphisms in Set *are precisely the injective functions and the isomorphisms in* Set *are precisely the bijective functions.*

(v) A set is a subsingleton set if and only if any two of its elements are equal.

(vi) Limits in Set *can be detected by* $\mathbf{1}$.

Proof. We will write down informal internal proofs of all the statements. Part (i) follows directly from the uniqueness clause in the definition of a coproduct. The points $x : \mathbf{1} \to A$ are the coprojections of $A = \bigsqcup_{x:A} \mathbf{1}$. If $f(x) = g(x)$ for all $x : A$ then that means that the composites of f and g with all coprojections are equal, which means that f and g must be equal by the uniqueness of the induced map. For part (ii) assume that $m : S \to A$ is a surjective monomorphism. Then it is true that for each $x : A$ there is a unique $y : S$ such that $m(y) = x$. By using 1-definite description and ν-abstraction we can define a function $r : A \to S$ such that $r(x) = \imath y.(my = x)$. We have that $m(r(x)) = x$ for each point $x : A$ by definition of r. Part (i) implies that $m \circ r = \mathrm{id}_A$. Note also that $r(m(y)) = y$ for each point $y : S$ because y is the unique $z : S$ such that $mz = my$. Hence it also holds that $r \circ m = \mathrm{id}_S$ and m is an isomorphism as claimed. To see part (iii) assume that $e : A \to B$ is a surjective function. If $f, g : B \to C$ are two functions such that $f \circ e = g \circ e$, then f and g must agree on points since e is surjective. This does imply that $f = g$ by part (i). So e is an epimorphism. To see that e is extremal assume that $m : S \to A$ is a monomorphism and that e factors as $e = m \circ e'$. Then m is surjective and (ii) implies immediately that m is an isomorphism. Let us next do part (iv). A monic map is automatically injective. So assume $m : A \to B$ is injective and $f, g : C \to A$ are two maps such that $m \circ f = m \circ h$. Then $m(fx) = m(gx)$ for each point $x : A$ and since m is injective we see that f and g agree on points. This implies that $f = g$ by part (i). An isomorphism is clearly bijective. When f is bijective then it is in particular a surjective monomorphism, and hence an isomorphism by part (ii). Let us next do part (v). A subsingleton set is a set A such that the unique map $A \to \mathbf{1}$ is monic. Assume that $A \to \mathbf{1}$ is monic and $x, y : A$ are two points of A. The composites of $x, y : \mathbf{1} \to A$ with $A \to \mathbf{1}$ are definitely equal by the universal property of $\mathbf{1}$, and hence x and y must be equal. Conversely assume that all points of A are equal. Then $A \to \mathbf{1}$ is injective, and part (iv) implies that it is a monomorphism. Finally let us do part (vi). Assume we have a diagram $D : (\mathrm{Set}^C)_0$ together with a cone $\lambda : \Delta L \Rightarrow D$ above D. Assume that $\mathbf{1}$ perceives that cone λ as a limit cone. By that we mean that for each transformation $\beta : \Delta\mathbf{1} \Rightarrow D$ there is a unique point $y : \mathbf{1} \to L$ such that $\lambda \circ \Delta y = \beta$. We can show under those conditions that λ, L is an actual limit of D in Set. Let A be any other set together with a transformation $\alpha : \Delta A \Rightarrow D$. Given any point $x : A$ we get a transformation $\alpha \circ \Delta x : \Delta\mathbf{1} \Rightarrow D$. The assumption now tells us that $\alpha \circ \Delta x$ induces a point $y : L$ of the set L. We can let $f : A \to L$ be the map which sends $x : A$ to the associated point of y, and one can check easily that $\lambda \circ \Delta f = \alpha$ holds, and that f is the only function with that property. The official term for f looks as follows.

$$\nu x.(\imath y.(\lambda \circ \Delta y = \alpha \circ \Delta x)) : \mathrm{Set}_1(A, L) \tag{2.23}$$

Here Δ denotes the diagonal $\Delta : \mathrm{Set} \to \mathrm{Set}^C$, which can be internally constructed. □

We are slowly starting to use the internal language in an informal way, and we are also starting to do bigger steps when we prove internal statements. As an example, here is what some of the informal internal statements of theorem 2.18 look like in our formal language. The statement (vi) takes the following form.

$$\begin{aligned}&\forall C : \mathrm{Cat}\,.\forall D : (\mathrm{Set}^C)_0.\forall L : \mathrm{Set}_0.\forall \lambda : (\mathrm{Set}^C)_1(\Delta_0 L, D).\\ &\quad((\forall \alpha : (\mathrm{Set}^C)_1(\Delta_0 \mathbf{1}, D).\exists! y : \mathrm{Set}_1(\mathbf{1}, L).\, \lambda \circ \Delta_1 y = \alpha)\\ &\quad\to (\forall A : \mathrm{Set}_0.\forall \alpha : (\mathrm{Set}^C)_1(\Delta_0 A, D).\exists! f : \mathrm{Set}_1(A, L).\, \lambda \circ \Delta_1 f = \alpha)) : \mathrm{Prop}_0\end{aligned} \tag{2.24}$$

It would look even longer if we inserted the official definition of Δ into the expression. The informal internal proof of (vi) should be understood as a manual on how to construct an official proof term of the proposition in the language $L(Sign)$ if one insists on doing so. You can try to produce such a proof term by hand, but it is probably more advisable to learn how to use a proof assistant if you are interested in formalisation. An easier example is statement (i), which officially looks as follows in our language.

$$\begin{aligned}&\forall A, B : \mathrm{Set}_0.\forall f, g : \mathrm{Set}_1(A, B).\\ &\qquad((\forall x : \mathrm{Set}_1(\mathbf{1}, A).\, f \circ x = g \circ x) \to (f = g))\end{aligned} \tag{2.25}$$

Some of the properties listed in theorem 2.18 are part of being of a "constructively well-pointed category" as defined in [42]. We are not yet able to show that Set satisfies all the conditions of a "constructively well-pointed" category. In particular we can not show internally that "every epimorphism is surjective" and that "$\mathbf{1}$ is projective", and it need in fact not be true in every model[11].

3 Application: The Coherence Problem of Extensional Dependent Type Theory

One of the standard models of dependent type theory are categories with families [38, Def 5.1]. Turning a category $\mathcal{S}$ into a category with families already requires some kind of strictification result because the reindexing operations in a category with families are strict so that they match the syntax of dependent type theory better. That strictification can be done in an ad-hoc way in many special cases. When $\mathcal{S}$ is a general category then one can use the left or the right adjoint splitting of its codomain fibration to obtain a category with families. The category with families which we implicitly used in section 1.3 was the one obtained via the right-splitting of the codomain fibration ∂_1.

[11] Assume for example that the topology W is the codiscrete topology. In that case the translation of "e is epi" is that the morphism e is epi in its fiber and in all lower fibers, while the translation of "e is surjective" is that e is split epi.

Once one has a category with families $CwF(\mathcal{S})$ one still has to interpret the type constructors of the type theory in $CwF(\mathcal{S})$. One usually starts with some knowledge about the base category $\mathcal{S}$. The base might be locally cartesian closed, it might be a pretopos, a Heyting-category or a regular category for example. In each case $\mathcal{S}$ and its self-indexing admit some universal construction such as binary products or quotients, but those constructions are only well-defined up to canonical isomorphisms. The task is then to lift the weakly defined constructions on $\mathcal{S}$ to corresponding strict constructions on its category with families $CwF(\mathcal{S})$. This is the coherence problem of extensional dependent type theory.

Example 3.1. Say we want to equip $CwF(\mathcal{S})$ with product type constructors. Some of the rules look as follows.

$$\frac{\Gamma \vdash X : Type \qquad \Gamma \vdash Y : Type}{\Gamma \vdash X \times Y : Type} \qquad\qquad \frac{\Gamma \vdash t : X \qquad \Gamma \vdash s : X}{\Gamma \vdash (t, s) : X \times Y} \tag{3.1}$$

If we want to interpret those rules in $CwF(\mathcal{S})$ then we need a way to associate a type $X \times Y \in Type(\Gamma)$ to each pair of types $X, Y \in Type(\Gamma)$, and we need a way to produce a new term $(t, s) \in Term(\Gamma, X \times Y)$ from terms $t \in Term(\Gamma, X)$ and $s \in Term(\Gamma, Y)$. All of those operations should commute strictly with reindexings to make a recursive interpretation of the type theory in $CwF(\mathcal{S})$ possible. The first few paragraphs of [17] explain what goes wrong if one does not make sure that all constructors commute strictly with reindexings. If $\mathcal{S}$ is lex then we know that each slice $\mathcal{S}/\Gamma$ has binary products and that those products are preserved by pullbacks up to canonical isomorphisms. The question is now: Given a lex category $\mathcal{S}$, can its category with families be equipped with product type constructors? How do we define them concretely? This is an instance of the coherence problem of dependent type theory. You can look in the introduction of [28] for a longer explanation of the coherence problem in general and also for more examples. ♢

The coherence problem of extensional dependent type theory was first solved by Hofmann in the paper [17]. Hofmann's paper looks at Martin Löf's extensional type theory with $\mathbf{1}, \Pi, \Sigma$ and Eq constructors and shows that $CwF(\mathcal{S})$ can be equipped with those constructors if $\mathcal{S}$ is a locally cartesian closed category. Hofmann shows that $CwF(\mathcal{S})$ admits $\mathbf{1}, \Pi, \Sigma$ and Eq types by constructing them by hand [17, Theorem 2]. The main aim of this section is to demonstrate that the language $L(Sign)$ and its semantics in $[\mathcal{S}^{op}, \mathrm{Cat}]$ can be used to solve the coherence problem of extensional type theory in a more systematic way by going through the following steps.

1. One reformulates the rules of the type constructor in question by means of an adjunction in $[\mathcal{S}^{op}, \mathrm{Cat}]$. We will for example show that $CwF(\mathcal{S})$ admits product types if and only if the diagonal $\Delta : \mathrm{Set} \to \mathrm{Set} \times \mathrm{Set}$ has a right adjoint.

2. Since Set satisfies 1-definite choice the actual existence of such an internal adjoint is equivalent to the mere pointwise existence of adjoint solutions (see example 2.12 part (a)). One can use the constructors $(\mathbf{L}, \eta)$ and $(\mathbf{R}, \varepsilon)$ to obtain an actual adjoint from adjoint solutions which merely exist pointwise.

3. The existence of pointwise solutions is a mere proposition in the internal language, which can be mechanically translated via the forcing relation. The result is in each case the already well-established 1-categorical semantics of the type constructor in question.

All this will become much clearer once we have gone through an example. Going through the steps 1-3 is interesting in its own right, because it allows us to characterise and reason about the type constructors of dependent type theory in the internal language $L(Sign)$ itself. We will see that the categorical description of a type constructors in the internal language is often simpler and more conceptual than the external categorical description of that type constructor. This is because the language $L(Sign)$ and its interpretation automatically manage the stages and stability under slicing.

We only do one example in this chapter to avoid repetition, but a treatment of a lot of other constructors can be found in appendix B. The strategy laid out above works well for type constructors which are determined up to unique isomorphisms. This excludes universes and also many constructors of intensional dependent type theory, because they are typically only determined up to homotopy equivalences or even weak equivalences [28].

3.1 Example: Strong Σ-Types

The rules of strong Σ-types look as follows [32].

$$\frac{\Gamma \vdash A : \mathrm{Set}_0 \qquad \Gamma, x : A \vdash B : \mathrm{Set}_0}{\Gamma \vdash \sum_{x:A} B : \mathrm{Set}_0} \qquad \frac{\Gamma \vdash N : A \qquad \Gamma \vdash M : B[N/x]}{\Gamma \vdash (N, M) : \sum_{x:A} B}$$

$$\frac{\Gamma \vdash P : \sum_{x:A} B}{\Gamma \vdash P_1 : A} \qquad \frac{\Gamma \vdash P : \sum_{x:A} B}{\Gamma \vdash P_2 : B[P_1/x]} \tag{3.2}$$

The computation rules are $(N, M)_1 \equiv N$ and $(N, M)_2 \equiv M$ and the uniqueness principle is $(P_1, P_2) \equiv P$. Intuitively $\sum_{x:A} B_x$ is the disjoint union of all the sets B_x.

Proposition 3.2. *If we add the rules (3.2) to the language $L(Sign)$, then we can internally show that "*Set *merely has coproducts indexed by internal sets". In fact, we can construct an internal functor which chooses them.*

Proof. We write down an informal internal proof. The objects of the internal category Fam(Set) : Cat are families of sets $(B_x)_{x:A}$ indexed by the points $x : \mathrm{Set}_1(\mathbf{1}, A)$ of an internal set. One can define an internal functor Δ which sends a set A to the object $\Delta A \equiv (A)_{x:\mathbf{1}}$ in Fam(Set). We say that Set merely has small coproducts when that functor Δ pointwise merely has a left adjoint. Equivalent to that is the following internal statement:

$$\forall A : \mathrm{Set}_0 . \forall B : (\textstyle\prod_{x:A} \mathrm{Set})_0 . \exists P : \mathrm{Set}_0 . \exists i : (\textstyle\prod_{x:A} \mathrm{Set})_1 (B, (P)_{x:A}) . \\ \ulcorner \textit{the pair } (P, i) \textit{ is initial among all such pairs} \urcorner \tag{3.3}$$

Version (3.3) looks closer to the typical definition of small coproducts in naive category theory. We will now show that we can produce an actual left adjoint $\bigsqcup$ of $\Delta : \mathrm{Set} \to \mathrm{Fam}(\mathrm{Set})$ when we are allowed to use the rules (3.2). The functor $\bigsqcup$ sends an object $(B_x)_{x:A}$ to the set $\sum_{x:A} B_x$ and a morphism $(u, (f_x)_{x:A}) : (B_x)_{x:A} \to (C_y)_{y:D}$ in Fam(Set) the function $\nu p.(u(p_1), f_{p_1}(p_2)) : \sum_{x:A} B_x \to \sum_{y:D} C_y$. The $(B_x)_{x:A}$th component of the unit $i : 1 \Rightarrow \bigsqcup \Delta$ is the morphism $(\imath t.\top, \nu b.(x, b)) : (B_x)_{x:A} \to (\sum_{x:A} B_x)_{x:\mathbf{1}}$ of families. To see that the pair $(\bigsqcup, i)$ satisfies the correct universal property take a set C together with a morphism $(\imath t.\top, (f_x)_{x:A}) : (B_x)_{x:A} \to (C)_{x:\mathbf{1}}$ in Fam(Set). One then can check with the help of axiom 1 that $\nu p.f_{p_1}(p_2)$ is the unique function $g : \sum_{x:A} B_x \to C$ which satisfies $\Delta g \circ i_{(B_x)_{x:A}} = (\imath t.\top, (f_x)_{x:A})$. □

Proposition 3.3. *The internal small coproducts, which we constructed in the last proposition, satisfy the following property internally: "Whenever A is a set and there is an $(x : A)$-indexed family of diagrams*

$$\begin{array}{ccc} D_x & \xrightarrow{j_x} & E \\ {\scriptstyle g_x}\downarrow & & \downarrow{\scriptstyle f} \\ B_x & \xrightarrow[i_x]{} & C \end{array}$$

such that the lower horizontal maps are the coprojections of a coproduct of $(B_x)_{x:A}$, then the upper horizontal maps are the coprojections of a coproduct if and only if each of the squares is a pullback."

Remark 3.4. The extra condition seems unmotivated, but it is actually a part of the standard definition of an infinitary extensive category [11]. In fact when "Set has small limits" already holds internally then the extra condition is just the statement that "Set is infinitary extensive". Extensive categories have long been recognised to be those categories where coproducts behave set-like, so it is only fitting that our internal characterisation of strong Σ-types involves that property. ♢

Proof. We make an informal internal argument. Assume we are given any set A and an $(x : A)$-indexed family of diagrams of the format shown above. Assume first that the upper and the lower horizontal maps are the coprojections of a coproduct. We can without loss of generality replace the upper and the lower coproducts by the canonical ones which we constructed from the strong Σ-types. Our diagrams look as follows.

$$\begin{array}{ccc} D_x & \longrightarrow & \sum_{x:A} D_x \\ {\scriptstyle g_x}\downarrow & & \downarrow{\scriptstyle f} \\ B_x & \longrightarrow & \sum_{x:A} B_x \end{array}$$

We need to check that each square is a pullback. It is enough to check them against the test object $\mathbf{1}$ because $\mathbf{1}$ can detect limits in Set. So fix some $x : A$ and two points $p : \sum_{x:A} D_x$ and $y : B_x$ such that $f(p) = (x, y)$. We can make the following computation.

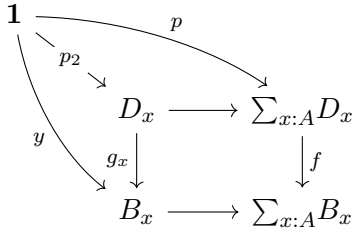
$$x = (x, y)_1$$

$$
\begin{aligned}
&= (fp)_1 \\
&= (f(p_1, p_2))_1 \\
&= (p_1, g_{p_1}(p_2))_1 \\
&= p_1
\end{aligned} \tag{3.4}
$$

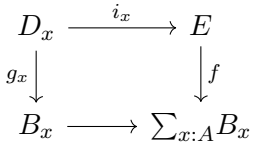
This shows that p_2 is of type $p_2 : D_x$. To see that $g_x p_2 = y$ make the following computation.

$$
\begin{aligned}
y &= (x, y)_2 \\
&= (fp)_2 \\
&= (x, g_x p_2)_2 \\
&= g_x p_2
\end{aligned} \tag{3.5}
$$

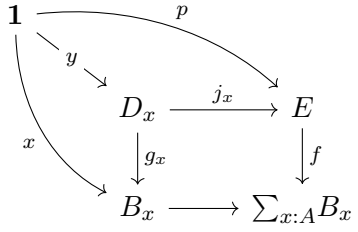
This means that the point p_2 fits into the diagram below, and it follows from the computation rule of the Σ-type that any other points which fits must be equal to p_2.



We have successfully shown that the xth square is a pullback diagram. For the other direction let us assume that we have an $(x : A)$-indexed family of pullback squares of the following form.



We like to show that the functions i_x are the coprojections of a coproduct. This is the same as showing that the canonical comparison map $\theta : \sum_{x:A} D_x \to E$ is an isomorphism. It is enough to check that θ is a bijective function, since the isomorphisms in Set are precisely the bijective functions. The action of θ on points is described by $\theta(x, y) = i_x(y)$. Given some point $p : \mathbf{1} \to E$ we get a point $fp : \mathbf{1} \to \sum_{x:A} B_x$ and hence a point $x = (fp)_1$ of A. One can look at the relevant pullback diagram



to obtain a point $y : D_{(fp)_1}$. The resulting point $(x, y) : \sum_{x:A} D_x$ clearly satisfies $\theta(x, y) = p$. This shows that θ is surjective. To see that θ is injective assume that (x, y) and (x', y') are two points such that $\theta(x, y) = \theta(x', y')$. We can compute:

$$\begin{aligned} x &= (x, g_x y)_1 \\ &= (f\theta(x, y))_1 \\ &= (f\theta(x', y'))_1 \\ &= x' \end{aligned} \tag{3.6}$$

That $y = y'$ follows now from the uniqueness clause in the universal property of the xth pullback diagram. □

We have established that we can internally construct small infinitary extensive coproducts for Set in $L(Sign)$ + (3.2). The opposite is also true, although we have to formulate it in a slightly different way.

Proposition 3.5. *If we add the axiom "*Set *has small coproducts which satisfy the extra condition of proposition (3.3)" to the language $L(Sign)$, then we can derive the rules (3.2) of strong Σ-types internally.*

Proof. What this means is that we can translate the rules (3.2) in a faithful way into our language $L(Sign)$ and make them derived rules. Let $\Delta : \mathrm{Set} \to \mathrm{Fam(Set)}$ be the internal functor which sends a set A to the indexed family $(A)_{x:\mathbf{1}}$ of sets. It is by assumption internally true that all the comma-categories $\mathrm{Fam(Set)}/\Delta$ have an initial object. Since Δ is well-defined in the empty context and since Set satisfies 1-definite choice, we are allowed to use the $(\mathbf{L}, \eta)$ constructor to obtain a functor $\mathbf{L}_\Delta : \mathrm{Fam(Set)} \to \mathrm{Set}$. We can make sense of the rules in (3.2) with the help of the chosen internal left adjoint $(\mathbf{L}_\Delta, \eta_\Delta)$ of Δ. The following rule can be derived from the rules of Fam(Set) and $(\mathbf{L}_\Delta, \eta_\Delta)$.

$$\frac{\Xi \vdash A : \mathrm{Set}_0 \qquad \Xi, x : A \vdash B_x : \mathrm{Set}_0}{\Xi \vdash \mathbf{L}_\Delta (B_x)_{x:A} : \mathrm{Set}_0} \tag{3.7}$$

If we use $\sum_{x:A} B$ as a short-hand for $\mathbf{L}_\Delta (B)_{x:A}$ then we have already explained the Σ-type introduction rule of (3.2). The first component of the unit $\eta_\Delta (B)_{x:A}$ contains no interesting information, but the second component contains a family of morphisms of the following type.

$$\frac{\Xi \vdash A : \mathrm{Set}_0 \qquad \Xi, x : A \vdash B_x : \mathrm{Set}_0}{\Xi, x : A \vdash (\eta_\Delta (B_x)_{x:A})_2(x) : \mathrm{Set}_1(B_x, \sum_{x:A} B_x)} \tag{3.8}$$

Let us write i_x instead of the lengthy official term $(\eta_\Delta (B_x)_{x:A})_2(x)$. We can now make sense of the second rule in (3.2).

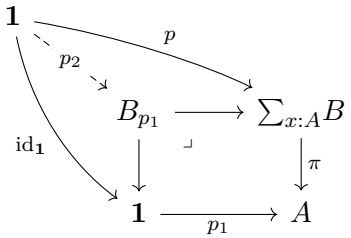
$$\frac{\Xi \vdash N : \mathrm{Set}_1(\mathbf{1}, A) \qquad \Xi \vdash M : \mathrm{Set}_1(\mathbf{1}, B_N)}{\Xi \vdash (N, M) : \mathrm{Set}_1(\mathbf{1}, \sum_{x:A} B_x)} \tag{3.9}$$

We just let (N, M) denote the internal composite $i_N \circ M$. It is harder to make sense of the elimination rules. This is where we will use the assumption that Set is infinitary extensive. First note that we can internally construct a projection

$\pi : \sum_{x:A} B_x \to A$. The $(x : A)$th component of π is the function $\nu y.x : B_x \to A$ which sends every point of B_x to $x : A$. We get the following $(x : A)$-indexed family of diagrams in Set.

$$\begin{array}{ccc} B_x & \xrightarrow{i_x} & \sum_{x:A} B_x \\ \downarrow & & \downarrow \pi \\ \mathbf{1} & \xrightarrow[x]{} & A \end{array}$$

The lower horizontal maps are the coprojections of a coproduct because of axiom 1. The upper horizontal maps are also the coprojections of a coproduct by assumption. The assumption that Set is infinitary extensive now tells us that each of the squares is a pullback diagram. Given a point $p : \mathrm{Set}(\mathbf{1}, \sum_{x:A} B_x)$ we can write p_1 to denote the composite $\pi \circ p$, and we write p_2 to denote the induced point in the diagram below.



This explains how we can internally make sense of the elimination rules of the strong Σ-type. One can also derive the computation rules and the uniqueness principle internally. □

All the results together tell us that adding strong Σ-types to the language $L(Sign)$ is essentially the same thing as adding the mere axiom that "Set merely has infinitary extensive small coproducts". We obtain in particular the following corollary.

Corollary 3.6. *Let $(\mathcal{S}, W)$ be a lex site such that the self-indexing of $\mathcal{S}$ is a stack. The following are equivalent:*

(i) The category with families $CwF(\mathcal{S})$ associated to $\mathcal{S}$ admits strong Σ-types.

*(ii) It holds in the model $(\mathcal{S}, W)$ that "*Set *merely has small coproducts which behave like the coproducts in an infinitary extensive category".*

The corollary holds in particular when W is the codiscrete topology, which means that the corollary characterises the meaning of strong Σ-types in any lex category $\mathcal{S}$. The external meaning of the second condition (ii) can be mechanically computed via the stack semantics. When one does this, then one gets the following fibered characterisation of an extensive Grothendieck fibration (taken from [43]): "For every arrow $u : \Delta \to \Theta$ in $\mathcal{S}$ and for every object x lying in the self-indexing above Δ there is an opcartesian arrow $\varphi : x \to y$ lying above u in the self-indexing. The opcartesian lifts must satisfy the following Beck-Chevalley condition. Whenever there is a commuting square

$$\begin{array}{ccc} C & \xrightarrow{\tilde{\varphi}} & D \\ {\scriptstyle\tilde{\psi}}\downarrow & & \downarrow{\scriptstyle\psi} \\ A & \xrightarrow[\varphi]{} & B \end{array}$$

lying above a pullback diagram in the base such that $\tilde{\psi}$ and ψ are cartesian and φ is opcartesian, then $\tilde{\varphi}$ will also be opcartesian. Additionally it must hold that for each commuting square of arrows

$$\begin{array}{ccc} X & \xrightarrow{\varphi} & U \\ {\scriptstyle\alpha}\downarrow & & \downarrow{\scriptstyle\beta} \\ Y & \xrightarrow[\psi]{} & V \end{array}$$

in the total category of the self-indexing where ψ is cocartesian and α and β are vertical it holds that φ is cocartesian if and only if the square is a pullback." The first part is the translation of "Set merely has small coproducts" in the form (3.3). The second part is the translation of the internal extensitivity condition.

We have recovered the usual necessary and sufficient categorical conditions on $\mathcal{S}$ so that $CwF(\mathcal{S})$ has strong Σ-types. We did none of the tedious computations that are done in the proof of [17, Theorem 2]. The strictification process is hidden in the interpretation of the $\mathbf{L}_\Delta$ constructor and the semicoflexibility of Set. It turns out that "Set merely has small products which behave like the coproducts in an infinitary extensive category" is always true in any model, and we are thus allowed to use strong Σ-types in all of our internal arguments.

3.2 The Other Constructors

Going through all the constructors of dependent type theory one by one takes a lot of space, so we will just look at the results. Detailed internal proofs can be found in appendix C.

$CwF(\mathcal{S})$ admits:	Internal Characterisation:
strong Σ-types	"Set is infinitary extensive"
binary product types	"Set has binary products"
extensional identity types	"Set has equalizers of pairs $x, y : \mathbf{1} \rightrightarrows A$"
singelton type	"Set has a terminal object"
empty type	"Set has a strict initial object"
weak sum types	"Set has coproducts"
Π-types	"Set has small products"

The table should be understood as follows: Whenever $(\mathcal{S}, W)$ is a model[12], then the category with families $CwF(\mathcal{S})$ associated with $\mathcal{S}$ will admit the type constructor listed on the left hand side if and only if the mere proposition shown on the right hand side holds in the model $(\mathcal{S}, W)$. For example $CwF(\mathcal{S})$ will have an empty type if and only if

$$(\mathcal{S}, W) \models \ulcorner \text{Set } \textit{has a strict initial object} \urcorner \tag{3.10}$$

[12]Remember that in particular $(\mathcal{S}, \mathit{codis})$ is a model when $\mathcal{S}$ is any category with finite limits.

is true[13]. The external meaning of the internal mere propositions on the right-hand side of the table can be mechanically computed via the forcing relation. One always gets the usual well-known external characterisation of the type constructor in question. For example

$$(\mathcal{S}, W) \models \ulcorner \text{Set } \textit{has small products} \urcorner \tag{3.11}$$

translates to "each reindexing functor u^* of the self-indexing has a right adjoint Π_u and the right adjoints satisfy the usual Beck-Chevalley condition". The proof that the type constructor on the left hand side correspond to the categorical construction on the right hand side can in each case be carried out in the internal language with heavy use of the ν-abstraction rule and its internal consequences. It is the ν-abstraction rule, or equivalently the axiom that every set is the coproduct of its points, which allows us to internally switch between the rules of type theory, which are formulated in terms of points, and the rules of the corresponding categorical constructions, which are formulated in terms of general arrows.

The extensional identity types mentioned in the table also have an alternative characterisation. In the presence of Σ-types one can internally construct all equalizers from the equalizers of points. Since every model $(\mathcal{S}, W)$ has strong Σ-types it follows that $CwF(\mathcal{S})$ admits extensional identity types if and only if

$$(\mathcal{S}, W) \models \ulcorner \text{Set } \textit{has equalizers} \urcorner \tag{3.12}$$

is true. More advanced type constructors are easier to characterise when one already internally assumes that strong Σ-types and finite limits are available in Set. Since every model $(\mathcal{S}, W)$ admits them anyway it only makes sense to add them officially to our language.

Axiom 2 (internal)**.** *The category* Set *is finitely complete and infinitary extensive.*

This means we can always use strong Σ-types and all finite limits in our internal arguments in the language $L(Sign)$. Using the additional axiom one can internally derive the following characterisations of more advanced type constructors.

$CwF(\mathcal{S})$ admits:	Internal Characterisation:
bracket types	"Set has coequalizers of pairs $\pi_1, \pi_2 : A \times A \rightrightarrows A$ and the coequalizers are subsingleton sets"
effective quotient types	"Set has effective quotients of internal equivalence relations"
natural numbers	"Set has a natural number object"

[13]The meaning of the statements on the left hand side of the table depends only on the category $\mathcal{S}$ while the meaning of the statements on the right hand side seems to depend on the topology W, so you might suspect that there is something wrong. But, as a matter of fact, the translation of all internal statements of the form "C has admits some universal construction" typically do not depend on the topology W as long as C is a stack.

Let us denote the full subcategory of Set whose objects are the subsingleton sets by SubS[14]. The bracket types allows us in particular to construct an actual left-adjoint $[-]$ of the internal inclusion functor SubS $\to$ Set. One can, using the bracket types and the strong Σ-types, also internally construct pullback stable image factorizations in Set. This gives us an alternative characterisation of bracket types: The category with families $CwF(\mathcal{S})$ associated to a model $(\mathcal{S}, W)$ admits bracket types if and only if

$$(\mathcal{S}, W) \models \ulcorner \text{Set } \textit{is regular} \urcorner \tag{3.13}$$

is true. The proof of that fact can again be carried out internally. Quotient types can also be characterised as an internal left adjoint, namely of the internal functor Set $\to$ Setoid which equips a set A with its diagonal. More information and detailed proofs can be found in appendix C.

4 Subsingleton Sets as Propositions

If we work in a model $(\mathcal{S}, W)$ which admits all of the constructors mentioned in the last chapter then the internal category SubS of subsingleton sets will behave a lot like the category Prop of propositions. We have for any two points $x, y : A$ a subsingleton set $\mathrm{Eq}_A(x, y)$ which measures how equal x and y are. The category of subsingleton sets is closed under limits taken in Set and one can produce colimits in SubS by using the internal left adjoint $[-] : \text{Set} \to \text{SubS}$. It so happens that the type constructors in the category of subsingleton sets model first-order intuitionistic logic when one translates statements in first-order logic into type-theoretic expressions via the following correspondence [2].

Logical Connective:	Type Constructor:
$\top$	$\mathbf{1}$
$\bot$	$\varnothing$
$A \wedge B$	$A \times B$
$A \vee B$	$[A + B]$
$\exists x : A.B_x$	$[\sum_{x:A} B_x]$
$\forall x : A.B_x$	$\prod_{x:A} B_x$
$A \to B$	$\prod_{x:A} B$
$s =_A t$	$\mathrm{Eq}_A(s, t)$

We are in the interesting situation that we have two internal large categories which model first-order intuitionistic reasoning, namely SubS and Prop. Both categories are very similar. The logical connectives in both categories correspond to various (co)limits, they are both preorders and they both satisfy the ν-abstraction rule[15].

[14] We can construct SubS for example by using the Full$(C, x.\phi)$ constructor of (1.29)

[15] The category SubS inherits the ν-abstraction rule from Set-

It would be nice to find conditions on our model $(\mathcal{S}, W)$ under which the logics of SubS and of Prop agree. Discussing such conditions is the aim of this chapter.

Let us fix an arbitrary model $(\mathcal{S}, W)$. We can internally construct a functor $K : \mathrm{Set} \to \mathrm{Prop}$. The functor sends a set A to the coproduct $KA \equiv \exists_{x:A}\top$ in Prop. Intuitively KA is the internal statement that A is inhabited. When $f : A \to B$ is a function between sets, then any point $x : A$ gives us a point $f(x) : B$. In particular B is inhabited whenever A is inhabited, and this describes the action of K on morphisms. Let us collect a few properties of the internal functor K.

Proposition 4.1. *The following are provable in the language $L(Sign)$ and thus hold in any model.*

(i) "The functor K preserves the terminal object and binary products."

(ii) "The functor K preserves infinitary coproducts."

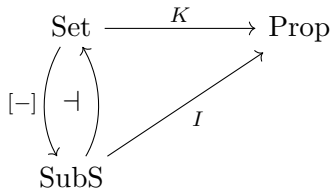
(iii) "When $x, y : \mathbf{1} \rightrightarrows A$ are two points of an internal set, then K sends their equalizer $\mathrm{Eq}_A(x, y)$ to the proposition $x = y$."

Proof. All of those can be shown by informal internal arguments. We have that $\exists x : \mathbf{1}.\top$ is true. This means that $\top$ and $K(\mathbf{1})$ are logically equivalent and hence isomorphic propositions. Given two points $x : A$ and $y : B$ we can construct a point $(x, y) : A \times B$, and given a point $p : A \times B$ we can construct points $p_1 : A$ and $p_2 : B$. This shows that $\exists x : A.\top \wedge \exists y : B.\top$ is equivalent to $\exists p : A \times B.\top$, so K preserves binary products. Assume we are given a family B_x of sets indexed by $x : A$. The rules of the strong Σ-type tell us that any point $p : \sum_{x:A} B_x$ gives us points $p_1 : A$ and $p_2 : B_{p_1}$ and conversely. This shows that $\exists x : A.\exists y : B_x.\top$ is equivalent to $\exists p : \sum_{x:A} B_x.\top$, so K preserves infinitary coproducts. For part (iii) we need to show that $x = y$ if and only if $\mathrm{Eq}_A(x, y)$ is inhabited. Assume that $x = y$. Then $x \equiv y$ by extensionality and we can derive $\mathrm{refl}(x) : \mathrm{Eq}_A(x, y)$. Conversely, assume that $\mathrm{Eq}_A(x, y)$ is inhabited. We have a diagram of the following form.

$$\begin{array}{ccccc} \mathrm{Eq}_A(x,y) & \longrightarrow & \mathbf{1} & \overset{x}{\underset{y}{\rightrightarrows}} & A \\ \uparrow & \nearrow & & & \\ \mathbf{1} & & & & \end{array}$$

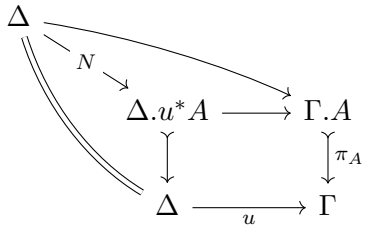
The diagram shows that $\mathrm{Eq}_A(x, y) \to \mathbf{1}$ is split epi. Since it is also monic it is an isomorphism and this implies that $x = y$. □

Let us next consider the internal composite I of K with the inclusion $\mathrm{SubS} \to \mathrm{Set}$. The functor I sends a subsingelton set A to $\exists x : A.\top$. The functor I also preserves binary products and sends $\mathrm{Eq}_A(x, y)$ to $x = y$, but it does not necessarily preserves infinitary coproducts because infinitary products in SubS are computed differently than in Set if they exist at all.



Even though the internal functor I does not necessarily preserve infinitary products, it is the only functor which can possibly preserve them and the terminal object. For if A is any subsingleton set, then A is the coproduct $A = \sum_{x:A} \mathbf{1} = [\sum_{x:A} \mathbf{1}]$ of its points in SubS and I must hence send A to $\exists_{x:A} \top$ in Prop when it preserves small coproducts and terminal objects.

It is interesting to see how the functor I acts from the external perspective. Let $\Gamma \vdash A : \mathrm{Set}_0$ be a subsingleton set. This means externally that the display map $\pi_A : \Gamma.A \to \Gamma$ of A is a monomorphism. The local sieve $\Gamma \vdash IA : \mathrm{Prop}_0$ on Γ contains precisely those $u : \Delta \to \Gamma$ which factor through π_A. To see this, assume first that $u : \Delta \to \Gamma$ factors through π_A. Then we have a section $\Delta \vdash N : u^*A$ induced by the following diagram.



This means that $\Delta \Vdash \exists x : u^*A.\top$ and we hence see that u is in the sieve $\Gamma \vdash \exists x : A.\top : \mathrm{Prop}_0$. Conversely, assume that $u : \Delta \to \Gamma$ is in the sieve $\Gamma \vdash \exists x : A.\top : \mathrm{Prop}_0$. This means that $\Delta \Vdash \exists x : u^*A.\top$ is true. It then follows that $\Delta \Vdash \exists! x : u^*A.\top$ is true because A is a subsingleton sets. We can use definite description to obtain a term $\Delta \vdash \imath x.\top : u^*A$. But having such a term means that we can factor u through the display map of A as claimed.

The functor I is automatically internally faithful since SubS is a preorder. It is also full as the following lemma shows.

Lemma 4.2 (internal)**.** *The functor* $I : \mathrm{SubS} \to \mathrm{Prop}$ *is full.*

Proof. This is an informal internal proof. Assume A and B are two subsingleton sets and that there is a morphism $P : \mathrm{Prop}_1(IA, IB)$. This means that IA implies IB. Given any point $x : A$, we know that A and hence B is inhabited, and can thus send $x : A$ to the unique point $\imath y.\top : B$ of B. Using ν-abstraction we get a function $A \to B$ in SubS, and it is clear that I sends that function to $IA \to IB$ since Prop is a preorder. □

The question which (co)limits are preserved and created by I can thus be replaced by the question under which universal construction its essential image is closed in Prop.

Example 4.3. Say for example that it is internally true that the essential image of I is closed under taking small coproducts in Prop. This can be formalised as follows:

$$\begin{aligned}\forall A : \mathrm{Set}_0.\forall \phi : (\textstyle\prod_{x:A}\mathrm{Prop})_0.\\ (\forall x : A.\exists B : \mathrm{SubS}_0\,.\ulcorner IB \textit{ and } \phi_x \textit{ are isomorphic}\urcorner)\\ \to (\exists B : \mathrm{SubS}_0\,.\ulcorner IB \textit{ and } \exists x : A.\phi_x \textit{ are isomorphic}\urcorner)\end{aligned} \tag{4.1}$$

Then we can use the fully faithfulness of I to derive internally that SubS has small coproducts and that I preserves them. Since SubS satisfies 1-definite choice one can even choose small coproducts in SubS and get a constructor $\exists : \mathrm{Fam}(\mathrm{SubS}) \to \mathrm{SubS}$ for them. ♢

The important question is thus under which operations the essential image of I is closed. We will now formulate a range of internal axioms for which express such closure conditions.

Definition 4.4. We say that a model can represent lex formulas if the following are internally true.

(i) "The essential image of I is closed under binary products in Prop and contains the terminal object."

(ii) "Whenever $x, y : \mathbf{1} \rightrightarrows A$ are two points of an internal set then $x = y$ lies in the essential image of I."

(iii) "Whenever ϕ_x is a family of propositions indexed by an internal set A such that each ϕ_x lies in the essential image of I and additionally $\forall x, y : A.\ ((\phi_x \wedge \phi_y) \to x = y)$ holds, then $\exists x : A.\phi_x$ lies also in the essential image of I."

Proposition 4.5. *Every model* $(\mathcal{S}, W)$ *can represent lex formulas*[16].

Proof. The informal internal proof will use an instance of 1-definite choice in SubS which we clarify below. The first two closure conditions (i) and (ii) follow from proposition 4.1 and the fact that limits in SubS are computed as in Set. We only have to show (iii). Assume we have a $(x : A)$-indexed family of propositions ϕ_x which lie in the essential image of I. This means that for each $x : A$ there is some subsingleton set B and an isomorphism $\alpha : IB \to \phi_x$. The (B, α) are determined up to unique isomorphism because I is fully faithful. The category SubS satisfies 1-definite choice. It follows that we can choose subsingleton sets $(B_x)_{x:A}$ and isomorphisms $\alpha_x : IB_x \to \phi_x$ for all $x : A$ at once. The assumption $\forall x, y : A.((\phi_x \to \phi_y) \to x = y)$ tells us that given any $x, y : A$ and points $p : B_x$ and $q : B_y$ it must be the case that $x = y$. This means that the set $\sum_{x:A} B_x$ is a subsingleton set. We can now use the fact that K preserves coproducts to make the following computation in Prop.

$$\begin{aligned} I(\textstyle\sum_{x:A} B_x) &= K(\textstyle\sum_{x:A} B_x)\\ &\cong \exists x : A.\exists y : B_x.\top\\ &\cong \exists x : A.\phi_x \end{aligned} \tag{4.2}$$

[16] Lex formulas are called "cartesian" in the elephant [21].

This shows that $\exists x : A.\phi_x$ lies in the essential image of I, which is what we wanted to show. □

We have informally appealed to 1-definite choice in our internal argument. It is still good to have an official formal statement which explains exactly what we did in that proof step. The following lemma does that.

Lemma 4.6. *The following is true in every model.*

$$\begin{aligned}\forall A : \mathrm{Set}_0.\forall \phi : (\textstyle\prod_{x:A}\mathrm{Prop})_0.& \\ (\forall x : A.\exists B : \mathrm{SubS}_0\,.(\phi_x \leftrightarrow \exists_{y:B}\top))& \\ \to (\exists B : (\textstyle\prod_{x:A}\mathrm{Prop})_0.\forall x : A.(\phi_x \leftrightarrow \exists_{y:B_x}\top))&\end{aligned} \tag{4.3}$$

Proof. One method to check this is to translate the statement into an external one via the forcing relation. One can then use that SubS is a stack to see that it is true. We can also use proposition 2.16 and our category constructors to write down an internal proof. It goes as follows: Assume we are given a set A and a family $(\phi_x)_{x:A}$ of propositions such that each ϕ_x lies in the essential image of I. For each x we may form the comma-category I/ϕ_x and cut out of it the full subcategory on those objects (B, α) for which the structure morphism $\alpha : IB \to \phi_x$ is an isomorphism. Denote that category by C_x. All the constructors which we used to build the categories C_x are from (1.29), and it is shown in the appendix that they all turn stacks into stacks. This is the only external input which we need. It is merely true that all the categories C_x are contractible, and we can use proposition 2.16 to conclude that $\prod_{x:A} C_x$ merely is contractible. But this means in particular that there merely is an object in $\prod_{x:A} C_x$, and this gives us the result that we wanted to show. □

Definition 4.7. A model can represent Δ_0-formulas [42, Convention 3.14] when it is internally true that the essential image of I is closed under small coproducts and products, under binary products and coproducts, under implication, contains $\top$ and $\bot$ and contains $x = y$ when $x, y : \mathbf{1} \rightrightarrows A$.

Definition 4.8. A model is logically complete when it is internally true that I is essentially surjective. One can then immediately use 1-definite choice to obtain an inverse equivalence Prop $\to$ SubS of I.

Remark 4.9. A different question is the following: Given some set A and a family of propositions $(\phi)_{x:A}$, can we internally construct a monomorphism $m : S \to A$ such that a point $x : A$ factors through m if and only if ϕ_x holds? If such a subset $m : S \to A$ exist then it is typically denoted by $\{x{:}A \mid \phi_x\}$. One says that the model satisfies separation for the predicate ϕ_x on A. When a model can represent a certain class of formulas via subsingleton sets, then it also satisfies separation for that class of formulas. This is not immediate and requires an application of 1-definite choice in SubS. The step from representation to separation uses lemma 4.6. Given some family $(\phi_x)_{x:A}$ of propositions which lie in the essential image of I, choose subsingleton sets B_x which represent the propositions ϕ_x and form the set $\sum_{x:A} B_x$. The projection $\sum_{x:A} B_x \to A$ is the monomorphism we are looking for. ♢

Remark 4.10. The paper [42] also considers a condition called *autological*. It sits in strength somewhere between Δ_0-separation and logically completeness. A model is autological when the essential image of I is closed under all proposition-constructors and not just the Δ_0-constructors. This is hard to express as a single internal axiom. If we for example want to express that the essential image is closed under existential quantification over the type of objects of a category, then the internal statement should start as follows.

$$\forall C : \mathrm{Cat} \,.\forall \phi : \textstyle\prod_{x:C_0} \mathrm{Prop}... \tag{4.4}$$

We don't have a category $\prod_{x:C_0}$ Prop though. But autologicalness can, as mentioned in [42], also be expressed as an axiom scheme. One rule of that scheme could for example look as follows.

$$\frac{\Xi, x : C_0 \vdash \phi_x : \mathrm{Prop}_0 \qquad \Xi, x : C_0 \Vdash \exists B : \mathrm{SubS}_0 \,.\ulcorner IB \textit{ is isomorphic to } \phi_x \urcorner}{\Xi \Vdash \exists B : \mathrm{SubS} \,.\ulcorner IB \textit{ is isomorphic to } \exists x : C_0.\phi_x \urcorner} \tag{4.5}$$

A model is *autological* when the axiom scheme indicated above is sound. Every logically complete model is in particular autological. ♢

When a model can represent Δ_0-formulas then this has a lot of consequences for its internal category of sets. It will for example be internally true that "Set is regular", that "Set is coherent" and even that "Set is a Heyting-category". All the necessary internal universal constructions in Set can be produced by using the internal (co)completeness properties of SubS and strong Σ-types.

The various internal closure axioms on the essential image of I also have many implications for the topology W. We need a little bit of preparation before we can show that.

Lemma 4.11. [29, p. 141] *Let $S = \langle \pi_i : \Delta_i \to \Gamma \rangle$ be a sieve on Γ generated by some family π_i of morphisms. There is a smallest W-local sieve T on Γ which contains S. Explicitly, $u \in T$ if and only if there is a cover v_j of the domain of u such that each uv_j factors through some π_i.*

Proof. Let T be the sieve described in the last half sentence. It is clear that T contains S and that any local sieve which contains S must also contain T. So we only need to show that T is local. Assume that $u : \Delta \to \Gamma$ is a map and there is a cover v_j of Δ such that each uv_j is in T. By definition of T there are covers w_{jk} such that each uv_jw_{jk} factors through one of the maps π_i. The maps v_jw_{jk} are a cover of Δ, and we hence see that u itself is contained in T. □

Lemma 4.12. *A sieve S is W-covering if and only if the smallest W-local sieve which contains S is the maximal sieve.*

Proof. Let T be the smallest W-local sieve containing S. We know explicitly how T looks like. Assume that S is a cover. Then $\mathrm{id}_\Gamma : \Gamma \to \Gamma$ is contained in T because S is a cover of Γ and each $\mathrm{id}_\Gamma f$ factors through some of the maps of S, namely f itself. Conversely, assume that T is the maximal sieve. Then id_Γ is in T, which means that S contains a covering family of morphisms by definition of T and hence must itself be a covering sieve. □

Proposition 4.13. *Let $(\mathcal{S}, W)$ be a model which can represent Δ_0-formulas. Let $\pi_{A_i} : \Gamma.A_i \to \Gamma$ be a finite family of maps. The following are equivalent.*

(i) The sieve S generated by the π_{A_i} is W-covering.

(ii) The categorical join of the categorical images of the π_{A_i} is the maximal subobject of Γ.

Proof. We have denoted the finite family of morphisms in the suggestive way $\pi_{A_i} : \Gamma.A_i \to \Gamma$, because we can equip each of them with extra reindexing data so that the π_{A_i} become the display maps of internal sets $\Gamma \vdash A_i : \mathrm{Set}_0$. Let us look at the local sieve $\Gamma \vdash \bigvee_{i=1}^n \exists_{x:A_i} \top : \mathrm{Prop}_0$ and denote it by T. We can compute its interpretation via the forcing relation: The sieve T contains precisely those maps $u : \Delta \to \Gamma$ for which there is a cover $v_j : \Theta_j \to \Delta$ such that each uv_j factors through one of the display maps π_{A_i}. In other words, the sieve T is exactly the smallest local sieve which contains the sieve $S = \langle \pi_{A_i} : \Gamma.A_i \to \Gamma \rangle$ generated by the display maps π_{A_i}. It is internally true that there is a subsingleton set B such that IB is isomorphic to the proposition $\bigvee_{i=1}^n \exists x : A_i.\top$. The subsingleton set B must be isomorphic to $B \cong \bigvee_{i=1}^n [A_i]$.

This is all the preparation we need. For the first direction assume that S is a cover. Then T is the maximal sieve and it follows that $\Gamma \vdash B : \mathrm{SubS}_0$ is inhabited. This means that the display map π_B of B is an isomorphism. The categorical interpretation of that display map is exactly the join of the images of the display maps π_{A_i}. This completes the first direction. For the other direction assume that $\Gamma \vdash \bigvee_{i=1}^n [A_i] : \mathrm{SubS}$ is inhabited. Since it is internally true that $I : \mathrm{SubS} \to \mathrm{Prop}$ preserves small and finite colimits and limits we have that $\Gamma \Vdash \bigvee_{i=1}^n \exists x : A_i.\top$ holds. This means that T is maximal, and in consequence that the sieve $S = \langle \pi_{A_i} : \Gamma.A_i \to \Gamma \rangle$ is covering. □

Condition (ii) is a condition which only depends on the base category $\mathcal{S}$ and not on the topology W. So we see that the "Δ_0-separation" condition on a model $(\mathcal{S}, W)$ determines if a sieve S is covering or not whenever the sieve S is generated by finitely many morphisms. It does not do that for sieves which do not have that property. When $\mathcal{S}$ is a Grothendieck topos then both $\mathcal{S}$ equipped with the coherent topology and $\mathcal{S}$ equipped with the canonical topology will are models which satisfy Δ_0-separation.

When a model $(\mathcal{S}, W)$ is logically complete then the topology W is completely determined by the category $\mathcal{S}$. This is because a sieve S is covering if and only if the smallest local sieve T which contains S is maximal. But when the model $(\mathcal{S}, W)$ is logically complete, then any local sieve on Γ is represented by a monomorphism $\pi_A : \Gamma.A \to \Gamma$ and the lattice of local sieves on Γ is in isomorphic to the lattice of subobjects of Γ in $\mathcal{S}$. Hence a sieve S is covering if and only if the smallest subobject of Γ through which all of the maps of the sieve factor is the maximal subobject. This is a condition which depends purely on the category $\mathcal{S}$. We see that logical completeness is a property of the underlying base category $\mathcal{S}$ and not of the site $(\mathcal{S}, W)$. There is at most one topology W on a category $\mathcal{S}$ equipped with which that category can become a logically complete model $(\mathcal{S}, W)$.

Lemma 4.14. *Grothendieck topoi* $\mathrm{Sh}_\lambda(C,J)$ *are logically complete.*

Proof. Here λ denotes an inaccessible cardinal and (C,J) is an internal site in Set_λ. The category $\mathrm{Sh}_\lambda(C,J)$ is a full subcategory of $(\mathrm{Set}_\lambda)^{C^{op}}$. The unique topology W with which $\mathrm{Sh}_\lambda(C,J)$ is a logically complete category is the canonical topology. The category $E = \mathrm{Sh}_\lambda(C,J)$ is locally Set_λ-small and the canonical topology is subcanonical. It is well-known that the Yoneda embedding

$$y : E \to \mathrm{Sh}_\lambda(E, can) \tag{4.6}$$

is an equivalence. This means that the subobject lattice of an object $X \in E$ is in canonical bijection with the subobject lattice of yX in $\mathrm{Sh}_\lambda(E,can)$. But the subobject lattice of yX is the lattice of *can*-local sieves on X, and hence we see that every *can*-local sieve on X can be represented by an actual subobject of X. This means that the internal functor $I : \mathrm{SubS} \to \mathrm{Prop}$ is an equivalence, and it follows in particular that I is internally essentially surjective on objects. □

5 Application: The Fibered Adjoint Functor Theorem

The aim of this section is to demonstrate that our language is expressive enough to internalise huge parts of naive category theory. We will prove the internal adjoint functor theorems in this section. Their proofs will be exact copies of the proofs from Emily Riehl's book "Category Theory in Context". We just write them down and make sure at each step that the reasoning is constructive and that we do not use more set theory than what is available in a general model $(\mathcal{S}, W)$. Remember that a model can be any lex site for which the self-indexing is a stack. This includes in particular lex categories with the codiscrete topology. We will externalise our result at the end of the chapter and in that way recover the fibered special adjoint functor theorem of fibered category theory [21, B2.4]. We take the adjoint functor theorems as an example because they are especially demanding on our language. Their proof relies on a subtle interplay between various smallness conditions The smallness conditions which we will need are local smallness, well-poweredness and having a small generating set of objects. We work propositionally truncated because this is the comfort zone of our language. But 1-definite choice will allow to escape mereness and extract an actual adjoint at the very end of our argument.

5.1 Smallness Conditions

Definition 5.1 (Internal)**.** A category C is *locally small* when for any two objects $x, y : C_0$ there is a set $H : \mathrm{Set}_0$ and a family $f_h : C_1(x,y)$ of morphisms indexed by $h : H$ such that for all morphisms $g : C_1(x,y)$ there is a unique $h : H$ for which $f_h = g$.

Remark 5.2. Definition 5.1 formulates local smallness as a mere property of C. It is also interesting to think about with which kind of extra data a category C should come equipped with to be considered a locally small category in the untruncated sense. Being actually locally small can of course never be a mere property of C in the internal language, because the hom-sets are extra data. Since we lack

the type constructors which would allow us build the very large type of locally small categories we have to formulate the definition of an "actually locally small category" half-externally. An actually locally small category in context Ξ is a category $\Xi \vdash C : \mathrm{Cat}$ together with the following data.

$$\begin{aligned} \Xi, x : C_0, y : C_0 &\vdash \mathrm{Hom}_C(x,y) : \mathrm{Set}_0 \\ \Xi, x : C_0, y : C_0, h : \mathrm{Hom}_C(x,y) &\vdash f_{xyh} : C_1(x,y) \\ \Xi &\vdash P : \forall x, y : C_0. \forall g : C_1(x,y). \\ &\quad \exists! h : \mathrm{Hom}_C(x,y). f_{xyh} = g \end{aligned} \tag{5.1}$$

The underlying category of an actually locally small category will in particular be internally locally small in the propositional truncated sense. When we have a category C which is actually locally small then we can do a lot of cool internal stuff with it. We can for example construct the internal Yoneda-embedding

$$y : C \to \mathrm{Pr}(C) \tag{5.2}$$

by using the dependent sets $x, y : C_0 \vdash \mathrm{Hom}_C(x,y) : \mathrm{Set}_0$. The index sets in the definition of a locally small category are determined up to unique compatible bijections. Since Set satisfies 1-definite choice it is reasonable to expect that there is no differences between being merely locally small and being actually locally small. This is indeed the case and we show it in appendix D. ♢

Definition 5.3 (Internal)**.** A category C has a small generating set of objects when there is a family of objects $(g_i)_{i:I}$ indexed by a set I such that for any two morphisms $f, h : x \rightrightarrows y$ for which $ft = ht$ holds for all $i : I$ and $t : g_i \to x$ it also holds that $f = h$.

Definition 5.4 (Internal)**.** A a category C is well-powered when for every object $x : C_0$ there is a family of monomorphisms $m_s : y_s \to x$ indexed by a set S such that for every monomorphism $n : z \to x$ there is a unique index s for which n and m_s are isomorphic as objects in C/x.

Remark 5.5. We can describe how an actually well-powered category looks like. The definition must again be half-externally. An actually well-powered category in the context Ξ is a category $\Xi \vdash C : \mathrm{Cat}$ which comes equipped with the following extra data.

$$\begin{aligned} \Xi, x : C_0 &\vdash \mathrm{Sub}_C(x) : \mathrm{Set}_0 \\ \Xi, x : C_0, s : \mathrm{Sub}_C(x) &\vdash y_{xs} : C_0 \\ \Xi, x : C_0, s : \mathrm{Sub}_C(x) &\vdash m_{xs} : C_1(y_{xs}, x) \end{aligned} \tag{5.3}$$

Additionally there should be a witness of the mere internal statement that each m_{xs} is monic and that every other monomorphism $n : z \to x$ is isomorphic to a unique one of the m_{xs} as a subobject of x. The data is unique up to unique isomorphisms, but this time also a family of objects y_{xs} in C is involved. One should thus expect that a merely well-powered internal category C is actually well-powered when C is a semicoflexible stack[17]. ♢

[17] I have not checked it carefully though because we do not need it, but I would be very surprised if it turned out to be false.

Definition 5.6 (internal)**.** A category C is locally small in families when for any two indexed families $(c_x)_{x:A}$ and $(d_y)_{y:B}$ of objects in C there are a family $H_{xy} : \mathrm{Set}_0$ of sets indexed by $x : A, y : B$ together with a family $f_{xyh} : C_1(c_x, d_y)$ of morphisms indexed by $x : A, y : B, h : H_{xy}$ such that for all $x : A, y : B$ and $g : C_1(c_x, d_y)$ there is a unique $h : H_{xy}$ such that $g = f_{xyh}$.

Proposition 5.7 (internal)**.** *Every category which is locally small is also locally small in families.*

Proof. From the external perspective the statement is clear because every merely locally small category can be equipped with a choice of hom-sets and hence is actually locally small in the untruncated sense. The underlying category of an actually locally small category is in particular locally small in families. A fully internal proof would go as follows: We form for each $p : A \times B$ a new category D_p of candidates for $\mathrm{Hom}_C(c_{p_1}, d_{p_2})$. One can internally show that all the categories D_p are contractible. We then use proposition 2.16 to to conclude that $\prod_{p:A\times B} C_p$ is contractible, and this implies that we can choose hom-sets for all $x : A, y : B$ at once. To make this a rigorous internal argument we would have to extend our language with category constructors which allow us to form the categories D_p of hom-set candidates, and we would also have to check the constructors which we use to build the D_p turn semicoflexible stacks into semicoflexible stacks. □

5.2 The Internal Adjoint Functor Theorems

We will now internally prove the adjoint functor theorems. We follow the treatment in Emily Riehl's book "Category Theory in Context" [40]. That this section is, for the most part, just a copy of a section from the category theory text book [40] is something that we should be very pleased with. This was after all the whole point of writing down the formal language and its interpretation. The take-away is that fibered category theory need not be more difficult than (constructive) naive category theory once the right translation apparatus is in place.

Lemma 5.8 (Internal)**.** [40, lemma 4.6.2] *For any functor $U : A \to S$ and any object $s : S_0$, the forgetful functor $\pi : s/U \to A$ creates all limits that A has and U preserves.*

Proof. The version of the lemma in Riehl's book says "strictly creates". We replace that by "creates" to be on the save side, since we do not have access to propositional equalities of objects. Let J be a category. Assume that A has limits of shape J and that U preserves them. We want to show that s/U has limits of shape J and that they are preserved by π. So let $D : J \to s/U$ be a functor. Take a limit $\lambda : \Delta L \Rightarrow \pi D$ of the diagram πD in the category A. Let us denote the object Dj in s/U by $\alpha_j : s \to U(\pi Dj)$. All the morphisms α_j form a cone over $U\pi D$, and there is hence an induced map $\beta : s \to UL$. The pair (β, L) is an object in the category s/U, and the λ_j give us morphisms $(\beta, L) \to Dj$. One can now check that the resulting cone in s/U is indeed a limit cone. By its definition the limit cone in s/U gets send to the cone λ in A via the functor $\pi : s/U \to A$, and we even get a

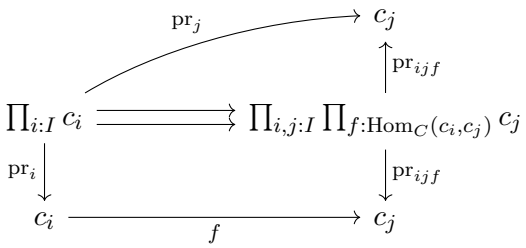
strict equalities $\pi(\beta, L) \equiv L$ and $\pi(\lambda_j) \equiv \lambda_j$ resulting from the computation rules of the involved category constructors. In particular π preserves limits of shape J. $\square$

Definition 5.9 (Internal)**.** A category C has a *weak initial set of objects* when there is a family of objects $(c_i)_{i:I}$ indexed by some small set I such that for all objects x in C there is an index i together with an arrow $f : c_i \to x$. A functor $U : A \to S$ satisfies the *solution set condition* when the category s/U has a weak initial set of objects for every $s : S_0$.

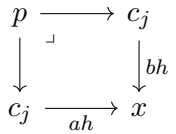
Definition 5.10 (Internal)**.** A category C is complete when it has binary products, a terminal object, equalizers and products indexed by small sets.

Lemma 5.11 (Internal)**.** [40, lemma 4.6.5] *If a category C is locally small, complete and has a weak initial set of objects, then C has an initial object.*

Proof. Take a weak initial set of objects $(c_i)_{i:I}$ and hom-sets $\mathrm{Hom}_C(c_i, c_j)$ indexed $i, j : I$. Here we use that C is locally small in families[18]. Look at the following pair of morphisms in C.



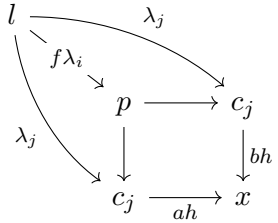
Let $\lambda : l \to \prod_{i:I} c_i$ be an equalizer of the two maps and denote the ith component of λ by λ_i. We have just computed the limit l of the small full subcategory of C generated by the c_i in elementary terms. We will now check that the object l is initial in C. Let x be any other object in C. We may find an index $i : I$ and a map $f : c_i \to x$. The composite $f \circ \lambda_i$ is a map $l \to x$, so we know that there is at least one map from l to x. This shows that l is weakly initial. Let $a : l \to x$ and $b : l \to x$ be any two maps which point from l into x. We can find an index j together with a map $h : c_j \to l$. The equation $\lambda_k h \lambda_j = \lambda_j$ holds for all k by definition of l, and since the λ_k are jointly monic we can conclude that $h\lambda_j = \mathrm{id}_l$. Let p be the pullback of ah and bh.



We may find a second index i and a morphism $f : c_i \to p$. The following diagram

[18] This is one of the places where we need to be a bit careful. The justification for why we are allowed to choose many hom-sets at once, even though we work constructively, is ultimately that Set satisfies 1-definite choice.

commutes by definition of l.



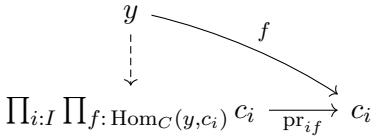
We can now compute that $a = a\,\mathrm{id}_l = ah\lambda_j = bh\lambda_j = b\,\mathrm{id}_l = b$ which finishes the proof. □

Theorem 5.12 (General Adjoint Functor Theorem, internal)**.** [40, theorem 4.6.3] *Let $U : A \to S$ be a functor. Assume that A is locally small and complete and that U satisfies the solution set condition and is continuous. Then each comma category s/U has an initial object.*

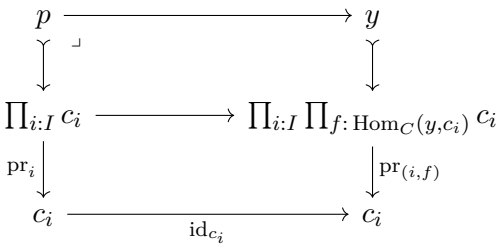
Proof. Fix some $s : S_0$. The category s/U is complete by lemma 5.8, and it has a weak initial set of objects by assumption. Lemma 5.11 then implies that the category s/U has an initial object. □

Lemma 5.13 (Internal)**.** [40, lemma 4.6.11] *If a category C is complete, well-powered, locally small and has a small coseparating family of objects, then it has an initial object.*

Proof. Take a small coseparating family $(c_i)_{i:I}$ of objects and hom-sets $\mathrm{Hom}_C(c_i, c_j)$. Here we are using again that C is locally small in families. We take a product $\prod_{i:I} c_i$ of all the objects c_i. Since C is well-powered we can find a set J together with monomorphisms $m_j : x_j \to \prod_{i:I} c_i$ indexed by $j : J$ such that every monomorphism into $\prod_{i:I} c_i$ is isomorphic to a unique one of the m_j as an object in the slice. Since C is complete we can form the intersection l of all of the monomorphisms m_j. We denote the structure maps by $\lambda_j : l \to x_j$. Our aim is to show that l is initial in C. We start by showing that l is weakly initial. Let y be any object in C. Take hom-sets $\mathrm{Hom}_C(y, c_i)$ and a product $\prod_{i:I} \prod_{f:\mathrm{Hom}_C(y,c_i)} c_i$. We construct the following map.



The fact that $(c_i)_{i:I}$ is a coseparating set of objects means that the morphism which we have constructed above must be monic. We take a pullback.



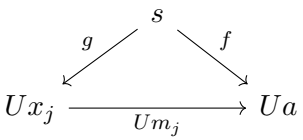
The subobject p is isomorphic to one of the canonical ones. Hence we get a morphism $l \to m_j \cong p$ and in consequence a morphism $l \to y$. To see that any two morphisms $f, g : l \to y$ must be equal, take the equalizer e of f and g. It is a subobject of l, but since l is minimal among all subobjects of $\prod_{i:I} c_i$ we see that e must be an isomorphism and hence that $f = g$. □

Lemma 5.14 (Internal)**.** [40, §4.6] *Assume that A is well-powered and complete and that S is locally small. Assume that $U : A \to S$ is a continuous functor. Then s/U is well-powered for any object $s : S_0$.*

Proof. We fix an object s in S. Lemma 5.8 tells us that s/U is complete and that the functor $\pi : s/U \to A$ creates limits. Since a morphism in a lex category is monic if and only if its kernel pair is the diagonal we see that the functor π preserves and reflects monicness in s/U. Take an object $f : s \to Ua$ in s/U. There is a set J and a family $m_j : x_j \to a$ of monomorphisms in A such that any monomorphism $n : z \to a$ is isomorphic to exactly one of the m_j as an object in A/a. We may also assume that we have hom-sets $\mathrm{Hom}_S(s, Ux_j)$ indexed by $j : J$, and also a hom-set $\mathrm{Hom}_C(s, Ua)$. We form the following set by using the type constructors of Set.

$$T := \textstyle\sum_{j:J} \sum_{g:\,\mathrm{Hom}_S(s,Ux_j)} \mathrm{Eq}(Um_j \circ g, f) \tag{5.4}$$

The morphism g in the second component is unique when it exists, which means that the canonical projection $T \to J$ is injective. We can associated to each point $(j, g, p) : T$ a monomorphism



in the category s/U. One can check that any monomorphism into $f : s \to Ua$ is isomorphic to a unique one of the monomorphisms associated to T which we just constructed. □

Remark 5.15. The hard part of the proof was to make sure that we can separate the set T out of the set J. Since we are working in a general model which might not satisfy strong separation principles we need to carefully make sure we can form those sets. Fortunately S is locally small and the morphism g is unique when it exists, which means that forming the set T is an instance of lex separation which is valid in every model. ♢

Lemma 5.16 (Internal)**.** [40, §4.6] *If $U : A \to S$ is a functor, A has a coseparating family and S is locally small, then s/U has a coseparating family for any object $s : S_0$.*

Proof. Fix some object $s : S_0$: Take a coseparating family $(a_i)_{i:I}$ for A and hom-sets $\mathrm{Hom}_S(s, Ua_i)$. Let J be the set $J := \sum_{i:I} \mathrm{Hom}_S(s, Ua_i)$. We can associate an object $f : s \to Ua_i$ in s/U to any point $(i, f) : J$ and it is not hard to check that this family of objects is a coseparating family of objects for s/U. □

Lemma 5.17 (internal)**.** *When $U : A \to S$ is a functor and both A and S are locally small and $s : S_0$ is an object in S, then s/U is locally small.*

Proof. Take two objects $g : s \to Ua$ and $f : s \to Ub$ in s/U. There are morphism sets $\mathrm{Hom}_A(a, b)$, $\mathrm{Hom}_S(s, Ua)$ and $\mathrm{Hom}_S(s, Ub)$. We can make the following definition.

$$\mathrm{Hom}_{s/U}((a, g), (b, f)) := \{h : \mathrm{Hom}_A(a, b) \mid Uh \circ g = f\} \tag{5.5}$$

This is an instance of lex separation which is valid in every model. One can check that the set above is the hom-set we are looking for. □

Theorem 5.18 (The Special Adjoint Functor Theorem, Internal)**.** [40, theorem 4.6.10]*Assume that both A and S are locally small, that A is complete and well-powered, that $U : A \to S$ is a continuous functor and that A has a small coseparating set. Then each comma-category s/U has an initial object.*

Proof. Fix an object $s : S_0$. The category s/U is complete by lemma 5.8, well-powered by lemma 5.14, locally small by lemma 5.17 and it has a small coseparating set of objects by lemma 5.16. Lemma 5.13 now implies that s/U has an initial object. □

Corollary 5.19 (Fibered Special Adjoint Functor Theorem)**.** [21, B2.4.6] *Let $\mathcal{S}$ be a lex category and assume that A and S are two Grothendieck fibrations above $\mathcal{S}$. If A and S are locally small, A is complete and well-powered, $U : A \to S$ is a continuous fibered functor and A has a small coseparating set (all in the fibered sense), then U has a left adjoint in the 2-category* $\mathrm{Fib}_{\mathcal{S}}$.

Proof. Equip $\mathcal{S}$ with the codiscrete topology to get a model $(\mathcal{S}, \mathit{codis})$ of our language. Apply the internal special adjoint functor theorem to $\mathrm{Sp}\, U : \mathrm{Sp}\, A \to \mathrm{Sp}\, S$ to get the following.

$$(\mathcal{S}, \mathit{codis}) \models \ulcorner \mathrm{Sp}\, U \textit{ merely pointwise has a left adjoint} \urcorner \tag{5.6}$$

We get an actual left adjoint of $\mathrm{Sp}\, U$ in $[\mathcal{S}^{op}, \mathrm{Cat}]$ because $\mathrm{Sp}\, A$ is a semicoflexible and a stack in the codiscrete topology. It follows that U has a left adjoint in $\mathrm{Fib}_{\mathcal{S}}$. The proof relies on the fact that all the internal categorical notions, like completeness and local smallness, correspond to the correct external fibered notions. This is checked in appendix D. □

We have proven the fibered adjoint functor theorem by following the proof laid out in the category theory textbook [40, section 4.6] word for word. The only work we had to do was checking that all arguments are constructive, and that they only use the limited amount of internal set-theory available in a general model $(\mathcal{S}, W)$. It is instructive to compare the proof in this section with the classical proofs that can be found in *Sketches of an Elephant* [21, B2.4] or the paper [22].

A 2-Categorical Background

We collect all the facts about 2-categories which we will need throughout the thesis in appendix A. The main aim of appendix A is to show that the classes of semicoflexible and n-separated indexed categories are closed under many of the constructions which we used in the main part of the thesis. To achieve that aim we will collect a range of results from different papers about 2-category theory and 2-monad theory and apply them to our situation.

We will work as strictly as possible, and only weaken our 2-categories and 2-functors in a very controlled way. We will thus follow the naming conventions of the papers [6, 24, 23]. All notions are assumed to be strict if not stated otherwise. For example a 2-functor is a Cat-enriched functor and a 2-adjunction is a Cat-enriched one. When K is a 2-category, then K_0 denotes the underlying 1-category of K and $|K|_1$ denotes its 1-truncation.

When K and L are 2-categories then we write $[K, L]$ to denote the 2-category of 2-functors, 2-transformations and modifications. The 2-category which additionally contains pseudo-transformations as 1-cells will be denoted by $\mathrm{Ps}(K, L)$, and the 2-category which additionally to that has all pseudo-functors as 0-cells will be denoted by $\mathrm{Hom}(K, L)$. The 2-category $\mathrm{Ps}(K, L)$ is a full sub-2-category of $\mathrm{Hom}(K, L)$. The inclusion $\mathrm{Ps}(K, L) \to \mathrm{Hom}(K, L)$ is a biequivalence when $L = \mathrm{Cat}$. There is also 2-category $\mathrm{Lax}(K, L)$ whose objects are strict 2-functors and whose 1- and 2-cells are lax transformations and modifications. Finally there is a 2-category $\mathrm{Oplax}(K, L)$ which has oplax transformations as 1-cells.

A.1 The Homotopy Theory of Strictification

A 2-monad is a monad in the Cat-enriched sense. Given a 2-monad $T : K \to K$ on a 2-category K, there are several associated 2-categories of algebras which one may consider. We denote the 2-category of strict algebra, strict algebra morphisms and transformations between them by $T\,\mathrm{Alg}_s$. One can also look at the 2-category $T\,\mathrm{Alg}$ where the algebras are still strict but the morphisms are allowed to preserve the algebra structure weakly up to invertible comparison cells [6]. There are also 2-categories $T\mathrm{Alg}_l$ and $T\mathrm{Alg}_{ol}$ where the algebras are still strict but the morphisms have directed coherence cells (lax and oplax). Finally, there is a 2-category $\mathrm{Ps}\,T\,\mathrm{Alg}$ consisting of pseudo-algebras and pseudo-morphisms between them. See [6] for precise definitions of all those 2-categories.

Example A.1. There is a 2-monad $T : \mathrm{Cat} \to \mathrm{Cat}$ such that $T\,\mathrm{Alg}_s$, is up to Cat-enriched equivalence, the 2-category of small categories with chosen terminal objects, products and equalizers [6, section 6.4]. The 1-cells are the functors which preserve the data on the nose and the 2-cells are transformations. The 2-category $T\,\mathrm{Alg}$ has as 1-cells functors which preserve the limits in the usual weak sense up to invertible coherence cells. This example shows that the 2-category $T\,\mathrm{Alg}$ is often the more important one. ♢

Example A.2. Let K be a 2-(co)complete 2-category and assume that $\mathcal{J}$ is a small 2-category. There is a 2-monad T on $\prod_{\mathrm{ob}\mathcal{J}} K$ such that the 2-categories $T\,\mathrm{Alg}_s$, $T\,\mathrm{Alg}$, $\mathrm{Ps}\,T\,\mathrm{Alg}$, $T\mathrm{Alg}_l$ and $T\mathrm{Alg}_{ol}$ are up to Cat-enriched equivalence precisely the

categories $[\mathcal{J}, K]$, $\mathrm{Ps}(\mathcal{J}, K)$, $\mathrm{Hom}(\mathcal{J}, K)$, $\mathrm{Lax}(\mathcal{J}, K)$ and $\mathrm{Oplax}(\mathcal{J}, K)$ respectively [6, remark 6.6]. The 2-monad is described in detail in remark 6.6 of [6]. The forgetful 2-functor $U_s : [\mathcal{J}, K] \to \prod_{\mathrm{ob}\,\mathcal{J}} K$ has both a left and a right 2-adjoint. ♢

When the underlying category K of a 2-monad $T : K \to K$ is 2-(co)complete and T has rank, then $T\,\mathrm{Alg}_s$ will be 2-(co)complete and the inclusion functor $J : T\,\mathrm{Alg}_s \to T\,\mathrm{Alg}$ has a left 2-adjoint which we denote by L [6, theorem 3.13].

Proposition A.3. [25, theorem 3.2] *Let $\mathcal{J}$ be a small 2-category and let K be a 2-(co)complete 2-category. Then the inclusion 2-functor $J : [\mathcal{J}, K] \to \mathrm{Hom}(\mathcal{J}, K)$ has both a left and a right Cat-enriched adjoint.*

Proof. The existence of the Cat-enriched left adjoint follows from [25, theorem 3.2]. The hypothesis that T preserves codescent objects is satisfied since $T = U_s F_s$ is a left 2-adjoint. The existence of the right adjoint follows from the dual of [25, theorem 3.2] together with the fact that $[\mathcal{J}, K]$ is comonadic above $\prod_{\mathrm{ob}\,\mathcal{J}} K$ and that the 2-comonad preserves descent objects because F_s has a further left 2-adjoint. □

The 2-functors L and R are very important and we will use them a lot. In the type theory literature they are known as the local universe construction [28] and the right adjoint splitting of a fibration [43, p. 12].

In the literature on 2-monads the left adjoint is often denoted by $(-)'$, and we will use that notation from time to time, especially when we talk about 2-(co)limits. There is also a left 2-adjoint $(-)^\dagger$ of the forgetful 2-functor $J_l : T\,\mathrm{Alg} \to T\,\mathrm{Alg}_l$ [24, p. 20], provided that the 2-monad T satisfy some finiteness conditions. The left 2-adjoint makes it possible to reduce lax limits and colimits to Cat-enriched ones.

Lack shows in [26] that every sufficiently (co)complete 2-category K has a canonical Cat-enriched model structure whose weak equivalences are exactly the equivalences in K. When T is a finitary 2-monad on K then one can lift that model structure along the adjunction $F_s \dashv U_s : T\,\mathrm{Alg}_s \to K$ to define the projective model structure on $(T\,\mathrm{Alg}_s)_0$. The resulting model structure is not the canonical one, its weak equivalences are precisely the 1-cells which become equivalences after applying the forgetful 2-functor $J : T\,\mathrm{Alg}_s \to T\,\mathrm{Alg}$. The underlying functor $(LJ)_0$ of the 2-comonad LJ together with its counit $LJ \Rightarrow 1$ is a cofibrant replacement for the projective model structure on $(T\,\mathrm{Alg}_s)_0$. The cofibrant objects in the projective model structure are called flexible algebras [26, theorem 4.12], and one can show that an algebra A is flexible if and only if the counit $LJA \to A$ has a section in the 1-categorical sense. All objects are fibrant in the projective model structure.

Example A.4. There is a model category structure on $(\mathrm{Cat})_0$ which presents the homotopy theory of all homotopy types [35], but that model structure is not compatible with the Cat-enrichment of Cat. This makes sense, since a Cat-enrichment compatible model structure should encode the two-dimensional homotopy theory of its 2-category. The lemma below also demonstrates that a Cat-enriched model structure correctly encodes the 2-dimensional homotopical behaviour of its 2-category. ♢

Lemma A.5. [16, lemma 9.5.13] *Assume K is a 2-(co)complete* Cat*-enriched model structure in the sense of* [26, §2.2]*. Let I denote the walking isomorphism. If A is cofibrant, then the tensor $I * A$ is a cylinder object for A. If B is fibrant, then the power object B^I is a path object for B.*

Proof. A proof in the case of sSet-enriched model categories can be found here [16, lemma 9.5.13]. In our case a proof looks as follows[19]. Applying the pushout product axiom to the acyclic cofibration $\{0\} \to I$ of Cat and the cofibration $\varnothing \to A$ of K shows that the map $i_0 : A \to I * A$ is a weak equivalence. Applying 2-out-of-3 to the diagram below shows that $I * A \to A$ is a weak equivalence.

$$\begin{array}{ccc} A & \xrightarrow{i_0} & I * A \\ & \searrow^{1_A} & \downarrow \\ & & A \end{array}$$

Applying the pushout product axiom to the cofibration $\{0, 1\} \to I$ of Cat and the cofibration $\varnothing \to A$ of K shows that the map $A \sqcup A \to I * A$ induced by $i_0 : A \to I * A$ and $i_1 : A \to I * A$ is a cofibration. The proof that B^I is a path object when B is fibrant is similar. □

Corollary A.6. *Assume that K is a 2-(co)complete* Cat*-enriched model category. Two morphisms from a cofibrant object A to a fibrant object B induce the same map in the homotopy category* $\mathrm{Ho}(K_0)$ *if and only if they differ by an invertible 2-cell.*

Proof. The theory of model categories tells us that two such morphisms induce the same map in the homotopy category if and only if there is a left homotopy between them with respect to any cylinder object. Since A is cofibrant the object $I * A$ is a cylinder object. A homotopy $H : I * A \to A$ between the two 1-cells $f : A \to B$ and $g : A \to B$ is exactly the same thing as an invertible 2-cell between f and g. □

We can even compute the homotopy category of $(T\,\mathrm{Alg}_s)_0$ with its model structure in a neat way. It is related to some of the other 2-categories which we have seen before.

Proposition A.7. [26, §4.14] *The 1-functor*

$$(T\,\mathrm{Alg}_s)_0 \to |T\,\mathrm{Alg}_s\,|_1 \xrightarrow{|J|_1} |T\,\mathrm{Alg}\,|_1$$

is the localisation $(T\,\mathrm{Alg}_s)_0 \to (T\,\mathrm{Alg}_s)_0[W^{-1}]$ of $(T\,\mathrm{Alg}_s)_0$ at the class of weak equivalences.

Proof. The theory of model categories tells us that the homotopy category can be constructed by taking the projectively fibrant-cofibrant algebras as objects and equivalence classes of 1-cells as morphisms, where two 1-cells define the same morphism in the homotopy category if and only if they are related by a left or right

[19] Reid Barton helped me with the proof on the Category-Theory Zulip server. If you ever read this Reid Barton, thank you!

homotopy (in the model category theoretic sense). Since the model structure on $(T\,\mathrm{Alg}_s)_0$ is compatible with the Cat-enrichment of $T\,\mathrm{Alg}_s$, it is the case that to 1-cells between fibrant-cofibrant algebras are homotopic in the model-theoretic sense if and only if they differ by an invertible 2-cell. This tells us that the homotopy category is the 1-truncation of the full sub-2-category of $T\,\mathrm{Alg}_s$ generated by the flexible algebras. The 2-functors J and L induce a biequivalence between the 2-category $T\,\mathrm{Alg}$ and the full sub-2-category of flexible algebras in $T\,\mathrm{Alg}_s$. This means they induce an equivalence between the 1-truncations and this shows that the homotopy category is equivalent to $|T\,\mathrm{Alg}\,|_1$. □

The reason we care about model structures and cofibrant objects is that they are related to the problem of strictifying pseudo-morphisms.

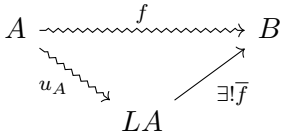
Proposition A.8. [26, proposition 4.13] *When an algebra A is projectively cofibrant then every pseudo-morhism $A \rightsquigarrow B$ can be replaced by an isomorphic strict one.*

Proof. Let B be a second algebra and assume that $f : A \rightsquigarrow B$ is a pseudo-morphism. The 1-cell $[f]$ induces a morphism in the homotopy category. The object A is cofibrant and the object B is fibrant because all objects are fibrant in the projective model structure of $T\,\mathrm{Alg}_s$. The theory of model categories tells us that

$$(T\,\mathrm{Alg}_s)_0(A, B) \to |T\,\mathrm{Alg}\,|_1(JA, JB)$$

is surjective in this case, and this directly implies that f can be replaced by an isomorphic strict 1-cell. □

Let us denote the counit of the adjunction by $q : LJ \Rightarrow 1$ and the unit by $u : 1 \Rightarrow JL$. Remember that the cofibrant objects of the model category are the flexible algebras [26]. Those are algebras A for which the counit $q_A : LJA \to A$ has a section. The components of q become equivalences after applying J. This can be shown directly without using the model structure on $T\,\mathrm{Alg}_s$ [6, theorem 4.2]. When A is an algebra then $u_A : A \rightsquigarrow LA$ is the universal pseudo-morphism out of A in the following sense: whenever B is a second algebra and $f : A \rightsquigarrow B$ is a pseudo-morphism then there is a unique strict map $\overline{f} : LA \to B$ such that $u_A \circ \overline{f} = f$. This follows directly from the fact that L is left 2-adjoint to J.



The universal pseudo-morphism u_A is a section of q_A in the 2-category $T\,\mathrm{Alg}$. This is one of the triangle identities $Jq \circ uJ = 1$. Since each entry of q is a weak equivalence we see that u_A is in fact a pseudo-inverse of q_A in $T\,\mathrm{Alg}$. The only thing that is preventing q_A from being an equivalence in the strict setting is that u_A might not be isomorphic to a strict algebra map.

Proposition A.9. [9, theorem 14] *Let A be an algebra. The following are equivalent:*

(i) The Ath component q_A of the counit has a pseudo-section in $T\,\mathrm{Alg}_s$. This means that there is some s_A such that $q_A \circ s_A \cong 1$.

(ii) The Ath component q_A of the counit is an equivalence in $T\,\mathrm{Alg}_s$.

(iii) The universal pseudomorphism $u_A : A \rightsquigarrow LJA$ is isomorphic to a strict morphism.

(iv) Every pseudo-morphism out of A is isomorphic to a strict one.

(v) The action $T\,\mathrm{Alg}_s(A, B) \to T\,\mathrm{Alg}(JA, JB)$ of J is an equivalence of categories for all other algebras B.

(vi) The 1-cell $q_A^* : T\,\mathrm{Alg}_s(A, -) \to T\,\mathrm{Alg}_s(LJA, -)$ has a pseudo-retraction in $\mathrm{Hom}(T\,\mathrm{Alg}_s, \mathrm{Cat})$.

Proof. Property (ii) implies (i), since every pseudo-inverse is automatically a pseudo-section. (i) is equivalent to (vi) by the bicategorical Yoneda lemma. Using the universal property of u_A it is not hard to see that (iii), (iv) and (v) are all equivalent to each other. When u_A is isomorphic to a strict map, then that strict map will be a pseudo-inverse of q_A in $T\,\mathrm{Alg}_s$. This shows that (iii) implies (ii). We are done when we can show that (i) implies (ii). So assume that s_A is a pseudo-section of q_A. Then Js_A is a pseudo-section of Jq_A in $T\,\mathrm{Alg}$. But since Jq_A is an equivalence it follows that Js_A is actually a pseudo-inverse of Jq_A, and because J is hom-cat-wise fully faithful we can conclude that s_A is also a pseudo-inverse of q_A in $T\,\mathrm{Alg}_s$. □

An algebra A which satisfies one and hence all of the conditions above is called semiflexible in the literature. The class of semiflexible algebras is precisely the closure of the class of flexible algebras under equivalence.

Proposition A.10. [6, theorem 4.17] *An algebra A is semiflexible if and only if it is equivalent to a flexible algebra in $T\,\mathrm{Alg}_s$. In particular all algebras of the form LJA and all flexible algebras are semiflexible.*

Proof. Let us first show that every flexible algebra is semiflexible. An algebra A is flexible if q_A is has a section. Clearly this implies that q_A has a pseudo-section and (i) of proposition A.9 applies. If A is equivalent to a flexible algebra, then it is in particular equivalent to a semiflexible algebra and hence semiflexible itself. Conversely assume that A is semiflexible. Then A is equivalent to LJA, so it is enough to check that LJA is flexible. To see that LJA is flexible, note that $qLJ \cong LuJ = 1_J$ by whiskering one of the triangle identities with J. Hence Lu_{JA} is a section of q_{LJA}. □

The most important 2-category for us is the 2-category of Cat-valued presheaves $[\mathcal{S}^{op}, \mathrm{Cat}]$ where $\mathcal{S}$ is a base 1-category. In that case we have that the forgetful 2-functor $U_s : [\mathcal{S}^{op}, \mathrm{Cat}] \to \prod_{\mathrm{ob}\,\mathcal{S}} \mathrm{Cat}$ has both a left and a right 2-adjoint, and in fact is both 2-comonadic and monadic. This suggests that the pointwise model structure on $\prod_{\mathrm{ob}\,\mathcal{S}} \mathrm{Cat}$ should also left-lift along U, so that there is a second model structure on $[\mathcal{S}^{op}, \mathrm{Cat}]$ with the same weak equivalences. This is indeed the case, and shown in detail in the paper [14] by Gambino. We call that model structure the injective

model structure. In the injective model structure all objects are cofibrant and the fibrant objects are precisely those objects A for which the unit $A \to RJA$ of the adjunction has a retract in the 1-categorical sense. Those objects are called coflexible in the literature. We denote the unit of the adjunction $J \dashv R$ by $r : 1 \Rightarrow RJ$ and the counit by $p : JR \Rightarrow 1$. The pair (R_0, r_0) is a functorial fibrant replacement for the injective model structure and $p_A : RA \rightsquigarrow A$ is the universal pseudo-map with codomain A. Dual results to those for the projective model structure hold. For example the following is true.

Proposition A.11. [6, theorem 4.7] *Let A be an indexed category. The following are equivalent.*

(i) The indexed category A is equivalent to a coflexible object.

(ii) The universal pseudo-morphism $p_A : RA \rightsquigarrow A$ is isomorphic to a strict one.

(iii) All pseudo-morphisms pointing into A are isomorphic to strict morphisms.

(iv) The 1-functor $T\,\mathrm{Alg}_s(A, B) \to T\,\mathrm{Alg}(JA, JB)$ is an equivalence for every other indexed category B.

(v) The 1-cell $(r_A)_*$ is an equivalence in $\mathrm{Hom}(T\,\mathrm{Alg}_s^{op}, \mathrm{Cat})$.

(vi) The 1-cell $(r_A)_*$ has a pseudo-retraction. This is equivalent to r_A having a pseudo-retract in $T\,\mathrm{Alg}_s$.

Proof. The proof is exactly the same as the one for the projective model structure. Here is an alternative proof. Look at the following diagram of functors.

$$\begin{array}{ccc} & T\,\mathrm{Alg}_s(B, RJA) & \\ {\scriptstyle (r_A)_*}\nearrow & & \searrow{\scriptstyle \cong} \\ T\,\mathrm{Alg}_s(B, A) & \xrightarrow[J]{} & T\,\mathrm{Alg}(JB, JA) \end{array}$$

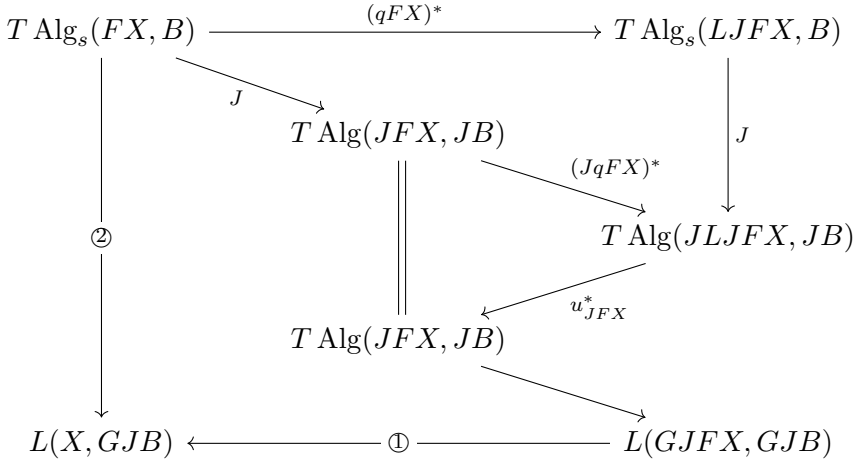
The isomorphism is that of the adjunction. Here we can directly see that J acts as an equivalence for each B if and only if $(r_A)_*$ is entry-wise an equivalence. Now using that J is already locally fully faithful we see that it is good enough when $(r_A)_*$ has a pseudo-retraction. We also see that being coflexible is a more strict special case of that. □

The class of semicoflexible objects is exactly the closure of the class of fibrant objects under equivalences in $[\mathcal{S}^{op}, \mathrm{Cat}]$. It is very useful for us to know when an indexed category is semicoflexible or flexible, and we thus want to show that the two classes are closed under many constructions. Here is a general result in that direction.

Lemma A.12. [6, theorem 5.1] *Assume that $G : T\,\mathrm{Alg} \to L$ is a 2-pseudo-functor such that GJ has a left biadjoint F. Then FX is semiflexible in $T\,\mathrm{Alg}_s$ for every 0-cell X in L.*

Proof. This is a slight modification of theorem 5.1 in [6], and our proof is also very similar. We start with a left biadjoint instead of a left 2-adjoint and only get asemiflexible object FX instead of a flexible one in return. Fix an object X in B.

We have the following diagram of functors which commutes at least up to natural isomorphisms.



The map ① is defined through precomposition and whiskering with the unit of the biadjunction $F \dashv GJ$. The map ② is an equivalence by definition of a biadjunction. We thus see that each component of $(qFX)^*$ has a pseudo-retract, and those retracts vary binaturally in B and thus assemble into a pseudo-retract in the 2-category $\mathrm{Hom}(T\,\mathrm{Alg}_s, \mathrm{Cat})$. By part (vi) of A.9 we can conclude that FX is semiflexible. □

By inspecting the proof of the lemma above one sees that when G is a 2-functor and F is a 2-adjoint, then we can produce a strict retraction of q^*_{FX} in $[T\,\mathrm{Alg}_s, \mathrm{Cat}]$ and hence conclude that FX is flexible. Similar results hold for the injective model structure and semicoflexible objects. Lemma A.12 is very useful. It allows us for example to reduce the construction of left biadjoints of bifunctors with domain $T\,\mathrm{Alg}$ to the construction of a left biadjoint of a bifunctor with domain $T\,\mathrm{Alg}_s$, which is often much easier.

Theorem A.13. [6, theorem 5.1] *Assume that $G : T\,\mathrm{Alg} \to L$ is a bifunctor and that $F : L \to T\,\mathrm{Alg}_s$ is a left biadjoint of GJ. Then JF is a left biadjoint of G.*

Proof. The proof is taken from the proof of theorem 5.1 in [6]. We spell it out since it is very short. We can compute that

$$\begin{aligned} T\,\mathrm{Alg}(JFX, JY) &\simeq T\,\mathrm{Alg}_s(FX, Y) \\ &\simeq L(X, GJY) \end{aligned}$$

for each object Y in $T\,\mathrm{Alg}_s$. We used in the first step that FX is almost flexible. Since each object in $T\,\mathrm{Alg}$ is of the form JY for some Y we are done. □

A.2 Homotopy (Co)limits

We will here summarise the most important facts about 2-limits and homotopy 2-limits. Following our general convention a weighted 2-limit is a Cat-enriched weighted limit. A good overview of weighted limits in the context of Cat-enriched categories can be found in the paper [23]. Cat-enriched limits are not preserved

by biequivalences and bi-right-adjoints, since they are not necessarily homotopical meaningful. In contrast homotopy 2-limits have biuniversal properties of the following form.

$$K(A, \mathrm{holim}_W D) \simeq \mathrm{Hom}(\mathcal{J}, \mathrm{Cat})(W, K(A, D-))$$

Here $\mathcal{J}$ is a bicategory and the weight $W : \mathcal{J} \to \mathrm{Cat}$ and the diagram $D : \mathcal{J} \to K$ can be bifunctors. When $\mathcal{J}$, W and D are strict and the equivalence is a 2-natural isomorphism, then we call the limit a pseudo-limit.

$$K(A, \mathrm{pslim}_W D) \cong \mathrm{Ps}(\mathcal{J}, \mathrm{Cat})(W, K(A, D-))$$

Pseudo-limits are thus a special case of homotopy limits. With help of the left 2-adjoint of $J : [\mathcal{J}, \mathrm{Cat}] \to \mathrm{Ps}(\mathcal{J}, \mathrm{Cat})$ we can do the following manipulation.

$$\begin{aligned} K(A, \mathrm{pslim}_W D) &\cong \mathrm{Ps}(\mathcal{J}, \mathrm{Cat})(W, K(A, D-)) \\ &\cong [\mathcal{J}, \mathrm{Cat}](LW, K(A, D-)) \\ &\cong K(A, \lim_{LW} D) \end{aligned}$$

Hence pseudolimits can be seen as a special case of strict weighted 2-limits, but with respect to a modified weight. In particular a 2-category which has all small 2-limits will in particular have all small 2-limits weighted by weights of the form LW and hence have all small homotopy 2-limits of strict diagrams and weights. A general result shows that it will then also have all homotopy limits indexed by non-strict 2-categories and weights [23]. The main point is that having all weighted 2-limits is actually a stronger condition than having all weighted homotopy 2-limits, since the homotopy 2-limits can be computed as strict 2-limits over modified weights.

Remember that any 2-category K comes with a canonical model structure which encodes its low-dimensional homotopical behaviour. There is also a notion of homotopy limit and colimit in the theory of model categories: The homotopy limit is the derived functor of the limit functor. The paper [14] explains how the model-category theoretic homotopy (co)limits relate to the 2-categorical homotopy limits defined above.

Both pseudolimits and 2-limits are only defined for strict diagrams and strict weights. They satisfy a strong universal property which allows us to turn them into 2-functors. If a 2-category K has all weighted pseudo-limits of a particular shape $\mathcal{J}$ then we may use the Cat-enriched Yoneda lemma to assemble choices of such limits into a 2-functor $\mathrm{pslim} : \mathrm{Ps}(\mathcal{J}, \mathrm{Cat}) \times \mathrm{Ps}(\mathcal{J}, K) \to K$. The homotopy limit is in contrast only well-defined up to equivalence, and the bicategorical Yoneda lemma only allows us to turn choices of homotopy limits into a bifunctor.

The literature on 2-limits and 2-colimits can be very confusing because there are two degrees of freedom along which the strictness of a (co)limit can vary. Table 1 below systematically lays out the different weakness/strictness combinations.

There are both omissions and redundancies in the table. For example weak lax limits and strong and weak oplax limits are missing. On the other hand a weak pseudo-limit is the same thing as a homotopy limit and a strong homotopy limit is a

Name	Universal Property
weighted 2-limit	$K(A, \lim_W D) \cong [\mathcal{J}, \mathrm{Cat}](W, K(A, D-))$
weak weighted 2-limit	$K(A, \lim^w_W D) \simeq [\mathcal{J}, \mathrm{Cat}](W, K(A, D-))$
pseudo-limit	$K(A, \mathrm{pslim}_W D) \cong \mathrm{Ps}(\mathcal{J}, \mathrm{Cat})(W, K(A, D-))$
weak pseudo-limit	$K(A, \mathrm{pslim}^w_W D) \simeq \mathrm{Ps}(\mathcal{J}, \mathrm{Cat})(W, K(A, D-))$
lax limit	$K(A, \mathrm{laxlim}_W D) \cong \mathrm{Lax}(\mathcal{J}, \mathrm{Cat})(W, K(A, D-))$
homotopy 2-limit	$K(A, \mathrm{holim}_W D) \simeq \mathrm{Hom}(\mathcal{J}, \mathrm{Cat})(W, K(A, D-))$
strong homotopy 2-limit	$K(A, \mathrm{holim}^s_W D) \cong \mathrm{Hom}(\mathcal{J}, \mathrm{Cat})(W, K(A, D-))$

Table 1: 2-Limits

pseudo-limit when both weight and diagram are strict. A second sort of redundancy and confusion comes from fact that many "strong weak[20] 2-limits", such as pseudo-limits or lax limits, can be expressed as strict 2-limits over modified weights LW and $W^\dagger$ respectively. A third kind of redundancy comes from the fact that when a weight is already flexible or semiflexible, then it makes up to equivalence no difference if we take a weighted 2-limit or a weighted homotopy-limit over it. In fact, this characterises the semiflexible weights according to [9, corollary 16].

Lemma A.14. [9, corollary 16] *If W is an semiflexible weight and K is a sufficiently 2-complete category, then $\lim_W(-) \simeq \mathrm{holim}_W(J-)$ as bifunctors into K.*

Proof. Sufficiently 2-complete just means that K has to admit the limits in question. One can directly check that both $\lim_W D$ and $\mathrm{holim}_W D$ birepresent the same 2-functor. Let us make the following calculation:

$$\begin{aligned} K(A, \lim_W D) &\cong [\mathcal{J}, \mathrm{Cat}](W, K(A, D-)) \\ &\simeq \mathrm{Ps}(\mathcal{J}, K)(W, K(A, D-)) \\ &\simeq K(A, \mathrm{holim}_W D) \end{aligned}$$

We used that the weight W is semiflexible in the second line. □

In 1-dimensional category theory limits and colimits in $\mathcal{C}$ can be described as right respectively left adjoints of the diagonal $\Delta : \mathcal{C} \to \mathcal{C}^{\mathcal{J}}$ which sends an object X to the diagram which constantly takes the value X. Something similar is possible for the various 2-(co)limit we described above. We start with the easiest case, namely weighted 2-limits in the Cat-enriched sense.

Lemma A.15. [1] *Assume that a 2-category K has tensors and weighted 2-limits of a fixed weight W. Then $\lim_W : [\mathcal{J}, K] \to K$ has a left 2-adjoint.*

[20] The first adjective refers to the strictness of the representation, and the second one to the strictness of the 2-functor that is being represented.

Proof. The proof is taken from here [1]. We can use the theory of enriched ends to produce the right 2-adjoint. Compute:

$$\begin{aligned} K(A, \lim_W D) &\cong [\mathcal{J}, \mathrm{Cat}](W, K(A, D-)) \\ &\cong \int_X \mathrm{Cat}(WX, K(A, DX)) \\ &\cong \int_X \mathrm{Cat}(WX * A, DX) \\ &\cong [\mathcal{J}, \mathrm{Cat}](W(-) * A, D) \end{aligned}$$

Hence the 2-functor $A \mapsto W(-) * A$ is a left 2-adjoint of $\lim_W$. □

Lemma A.16. *Assume that K has tensors and pseudo-limits of a fixed weight W. Then the 2-functor* $\mathrm{pslim}_W : \mathrm{Ps}(\mathcal{J}, K) \to K$ *has a left 2-adjoint.*

Proof. It is not enough that $\lim_{LW}$ has a left 2-adjoint, since $\lim_{LW}$ has a smaller domain than pslim_W. We can use a very similar proof though using the theory of pseudo-ends, which is developed in the very recent paper [15]. We compute:

$$\begin{aligned} K(A, \mathrm{pslim}_W D) &\cong \mathrm{Ps}(\mathcal{J}, \mathrm{Cat})(W, K(A, D-)) \\ &\cong \int_X^{ps} \mathrm{Cat}(WX, K(A, DX)) \\ &\cong \int_X^{ps} \mathrm{Cat}(WX * A, DX) \\ &\cong \mathrm{Ps}(\mathcal{J}, \mathrm{Cat})(W(-) * A, D) \end{aligned}$$

There is also an alternative proof which uses theorem A.13, but it only produces a left biadjoint and not a 2-adjoint. Here is how it goes: By theorem A.13 it is enough to produce a left biadjoint F of $\mathrm{pslim}_W(J-)$. But the functor $\mathrm{pslim}_W(J-)$ is isomorphic $\lim_{LW}(-)$, and we have already shown that $\lim_{LW}(-)$ has $A \mapsto LW(-) * A$ as a left 2-adjoint. Hence we see that $A \mapsto J(LW(-) * A)$ is a left biadjoint of $\mathrm{pslim}_W(-)$. □

Note that the left biadjoint which we get from theorem A.13 is equivalent to $A \mapsto W(-) * A$ but not isomorphic. We see that theorem A.13 in general only produces left biadjoints and not 2-adjoints, even when a 2-adjoint exists.

Lemma A.17. *Assume that K has weak tensors and homotopy limits of weight W. Assume additionally that* $\mathrm{Ps}(\mathcal{J}, K) \to \mathrm{Hom}(\mathcal{J}, K)$ *is a biequivalence*[21]*. Then* $\mathrm{holim}_W : \mathrm{Hom}(\mathcal{J}, K) \to K$ *has a left biadjoint.*

Proof. This time it would be wrong to ask for a left 2-adjoints, since homotopy limits are only well defined up to equivalence anyway. By assumption $\mathrm{Ps}(\mathcal{J}, K) \to \mathrm{Hom}(\mathcal{J}, K)$ is a biequivalence. This means it is enough to produce a left biadjoint of the restriction $\mathrm{holim}_W : \mathrm{Ps}(\mathcal{J}, K) \to K$, and this can be done using pseudo-ends and weak tensors exactly as in the proof of lemma A.16. □

Using lax ends [15] one can presumably also show that lax limit functors have left 2-adjoints. Corresponding results should of course also hold for weighted 2-colimits of all kinds.

[21] For example because [25, theorem 3.2] applies

A.3 Semicoflexible Indexed Categories

We will show in this section that the class of semicoflexible indexed categories is closed under semiflexible weighted 2-limits and 2-exponentials.

Proposition A.18. [9, proposition 23] *The class of semicoflexible indexed categories in* $[\mathcal{S}^{op}, \mathrm{Cat}]$ *is closed under 2-limits weighted by semiflexible weights.*

Proof. See also [9, proposition 23] for comparison. Assume that $\mathcal{J}$ is a 2-category and D is a diagram in $[\mathcal{S}^{op}, \mathrm{Cat}]$ such that all vertices Dj of D are semicoflexible. Assume that W is a semiflexible weight. We can compute:

$$\begin{aligned}
[\mathcal{S}^{op}, \mathrm{Cat}](A, \lim_W D) &\cong [\mathcal{J}, \mathrm{Cat}](W, [\mathcal{S}^{op}, \mathrm{Cat}](A, D-)) \\
&\simeq \mathrm{Ps}(\mathcal{J}, \mathrm{Cat})(JW, J[\mathcal{S}^{op}, \mathrm{Cat}](A, D-)) \\
&\simeq \mathrm{Ps}(\mathcal{J}, \mathrm{Cat})(JW, J\,\mathrm{Ps}(\mathcal{S}^{op}, \mathrm{Cat})(JA, JD-)) \\
&\simeq [\mathcal{J}, \mathrm{Cat}](W, \mathrm{Ps}(\mathcal{S}^{op}, \mathrm{Cat})(JA, JD-)) \\
&\cong \mathrm{Ps}(\mathcal{S}^{op}, \mathrm{Cat})(JA, \lim_W(JD)) \\
&\cong \mathrm{Ps}(\mathcal{S}^{op}, \mathrm{Cat})(JA, J(\lim_W D))
\end{aligned}$$

The first step uses the definition of a weighted limit, the second step uses that W is semiflexible. The third step follows from the fact that D takes values in semicoflexible objects so that the natural map $[\mathcal{S}^{op}, \mathrm{Cat}](A, D-) \to \mathrm{Ps}(\mathcal{S}^{op}, \mathrm{Cat})(JA, JD-)$ is entry-wise an equivalence and hence an equivalence in $\mathrm{Ps}(\mathcal{J}, \mathrm{Cat})$. The fourth step uses again that W is almost flexible, and the fifth step uses the definition of $\lim_W(JD)$. The final step uses that J is a right 2-adjoint and hence preserves 2-limits. One can now check that the chain of equivalence composes up to isomorphism to the action of J and this finishes the proof. □

The class of semiflexible weighted 2-limits is huge and includes in particular all (PIE)-weighted limits and all pseudo-limits. It does not include 2-equalizers and strict 2-pullbacks. Intuitively all those limit-type category constructors are safe which do not speak about equalities of objects.

The 2-category $[\mathcal{S}^{op}, \mathrm{Cat}]$ of indexed categories over $\mathcal{S}$ is 2-cartesian closed. There are 2-natural varying isomorphisms of the following form.

$$[\mathcal{S}^{op}, \mathrm{Cat}](A \times B, C) \cong [\mathcal{S}^{op}, \mathrm{Cat}](A, C^B)$$

We can explicitly compute how the indexed category C^B looks like by using the 2-Yoneda lemma. When Γ is an object of the base category $\mathcal{S}$, then we can make the following computation.

$$\begin{aligned}
(C^B)_\Gamma &\cong [\mathcal{S}^{op}, \mathrm{Cat}](y\Gamma, C^B) \\
&\cong [\mathcal{S}^{op}, \mathrm{Cat}](y\Gamma \times B, C)
\end{aligned}$$

The formula is completely analogues to the well-know formula for exponentials in presheaf 1-categories. The 2-categories $\mathrm{Ps}(\mathcal{S}^{op}, \mathrm{Cat})$ and $\mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})$ are bicartesian closed. This means that there are objects C^B such that the following weaker 2-universal property holds.

$$\mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})(A \times B, C) \simeq \mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})(A, C^B)$$

Using the biYoneda lemma one can compute that the exponentials in $\mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})$ can be defined through the analogous formula $(C^B)_\Gamma = \mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})(y\Gamma \times B, C)$. The forgetful 2-functor $J : [\mathcal{S}^{op}, \mathrm{Cat}] \to \mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})$ need not preserve 2-exponentials. It weakly preserve those 2-exponentials where the second entry is semicoflexible though.

Lemma A.19. *Assume that C is semicoflexible. Then the canonical map $J(C^B) \to JC^{JB}$ is an equivalence in* $\mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})$.

Proof. We can make the following computation:

$$\begin{aligned}
\mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})(A, J(C^B)) &\simeq \mathrm{Ps}(\mathcal{S}^{op}, \mathrm{Cat})(JLA, J(C^B)) \\
&\simeq [\mathcal{S}^{op}, \mathrm{Cat}](LA, C^B) \\
&\cong [\mathcal{S}^{op}, \mathrm{Cat}](LA \times B, C) \\
&\simeq \mathrm{Ps}(\mathcal{S}^{op}, \mathrm{Cat})(J(LA \times B), JC) \\
&\cong \mathrm{Ps}(\mathcal{S}^{op}, \mathrm{Cat})(JLA \times JB, JC) \\
&\simeq \mathrm{Ps}(\mathcal{S}^{op}, \mathrm{Cat})(JLA, JC^{JB}) \\
&\simeq \mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})(A, JC^{JB})
\end{aligned}$$

The first line uses that $JLA \to A$ is an equivalence and the second line uses that LA is flexible. The fourth line uses that C is semicoflexible. The fifth line follows from the fact that J is a right 2-adjoint and hence preserves products. The last line uses again that $JLA \to A$ is an equivalence. The equivalence of hom-categories varies binaturally in A, and hence we see that JC^{JB} and $J(C^B)$ satisfy the same biuniversal property. This means that the canonical comparison map must be an equivalence. □

Proposition A.20. *If C is a semicoflexible indexed category, then C^B is semicoflexible for any B.*

Proof. We have just shown that $J(C^B)$ is equivalent to JC^{JB}. Using this we can make the following computation:

$$\begin{aligned}
\mathrm{Ps}(\mathcal{S}^{op}, \mathrm{Cat})(JA, J(C^B)) &\simeq \mathrm{Ps}(\mathcal{S}^{op}, \mathrm{Cat})(JA, JC^{JB}) \\
&\simeq \mathrm{Ps}(\mathcal{S}^{op}, \mathrm{Cat})(J(A \times B), JC) \\
&\simeq [\mathcal{S}^{op}, \mathrm{Cat}](A \times B, C) \\
&\cong [\mathcal{S}^{op}, \mathrm{Cat}](A, C^B)
\end{aligned}$$

The proof is done once one has checked that the chain of equivalence is up to isomorphism the action of J. We skip over that step. □

A.4 n-Monic Morphisms

We will prepare ourselves to talk about stacks and prestacks in this section. For that we will need the definition of an n-monic map in a 2-category. Motivation for the definition of an n-monic map in the context of higher categories can be found for example in [3]. The authors of [3] also explain how the definition of an n-monic map is related to lifting conditions on cells of varying dimension.

Definition A.21. Let n be 0, 1 or 2. A 0-cell X in a 2-category K perceives a 1-cell $f : A \to B$ as n-monic when $K(X, f) : K(X, A) \to K(X, B)$ is faithful in case $n = 0$, fully faithful in case $n = 1$ or an equivalence in case $n = 2$. The object X perceives $f : A \to B$ as n-opmonic when the functor $K(f, X)$ is faithful, fully faithful or an equivalence respectively. A 1-cell in K is n-(op)monic when it is perceived as such by all 0-cells of K.

Proposition A.22. *Right biadjoints send n-monic maps to n-monic maps.*

Proof. Let $L : M \to K$ be a left biadjoint of $R : K \to M$ and assume $f : A \to B$ is n-monic in K. Given any X in M, we need to check that the functor $M(X, Rf) : M(X, RA) \to M(X, RB)$ is faithful, fully faithful or an equivalence respectively. By biadjunction there is a diagram

$$\begin{array}{ccc} L(X, GA) & \xrightarrow{(Rf)_*} & L(X, GB) \\ \downarrow\simeq & & \downarrow\simeq \\ K(FX, A) & \xrightarrow{f_*} & K(FX, B) \end{array}$$

in Cat which commutes up to an invertible natural transformation. It follows that $(Rf)_*$ is faithful, fully faithful or an equivalence respectively if and only if f_* is. □

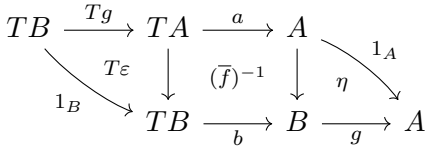
Lemma A.23. *A morphism $f : A \to B$ in a 2-category K is 2-monic if and only if it is an equivalence.*

Proof. The morphism f is 2-monic if and only if all functors $K(X, f)$ are equivalences. This is the case if and only if the transformation $f_* : K(-, A) \to K(-, B)$ is an equivalence in $\mathrm{Hom}(K^{op}, \mathrm{Cat})$. The biyoneda lemma implies that this holds if and only if f itself is an equivalence in K. □

Theorem A.24. [26, p. 16] *Let T be a 2-monad on a 2-category K. The 2-functor $U : T\,\mathrm{Alg} \to K$ preserves and reflects n-monicness. The 2-functor $U_s : T\,\mathrm{Alg}_s \to K$ preserves and reflects* 0*-monicness,* 1*-monicness and isomorphisms, but does not in general reflect equivalences.*

Proof. We start by showing that $U : T\,\mathrm{Alg} \to K$ preserves and reflects n-monicness. The forgetful 2-functor $UJ = U_s$ has a left 2-adjoint F_s. It follows from theorem A.13 that JF_s is a left biadjoint of U. This means that $U : T\,\mathrm{Alg} \to K$ is a right biadjoint and hence sends n-monic maps to n-monic maps. For the other direction, let us start with the case $n = 2$. We need to show that the 2-functor $U : T\,\mathrm{Alg} \to K$ reflects equivalences. This is a "routine (but important) exercise" according to Lack [26, p. 16] but a direct and elementary proof is in fact tedious. Here is a sketch of the argument. Assume that $(f, \overline{f}) : (A, a) \to (B, b)$ is a pseudo-morphism of algebras such that f is an equivalence in K. We may choose a pseudo-inverse g of f in K which is at the same time a right adjoint. Let us denote the invertible unit and counit by $\eta : 1_A \to gf$ and $\varepsilon : fg \to 1_B$ respectively. We like to turn g into a

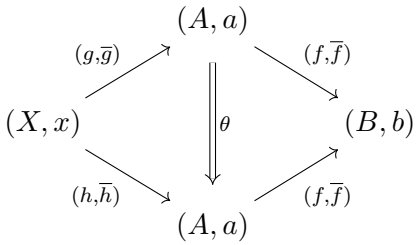
pseudo-morphism $(g, \overline{g}) : (B, b) \to (A, a)$ of algebras. Let $\overline{g}$ be the pasting of the 2-cells drawn below.



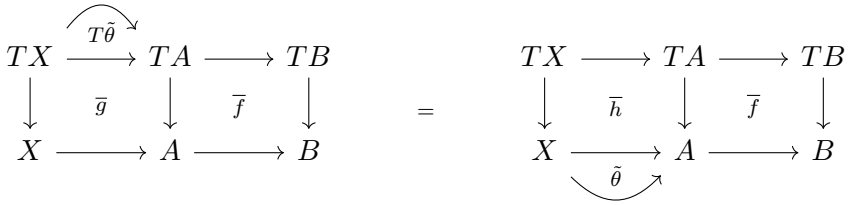
One has to check that $(g, \overline{g})$ satisfies the two coherence conditions of a pseudo-morphism of algebras, which takes a lot of space. Next one checks that the 2-cells $\eta : 1 \to gf$ and $\varepsilon : fg \to 1$ lift to invertible 2-cells in $T\,\mathrm{Alg}$. This is again a fairly long calculation with pasting diagrams. Once this is done we see that $(g, \overline{g})$ is a pseudo-inverse of $(f, \overline{f})$ so that $(f, \overline{f})$ is an equivalence as claimed. Next, let us do the case $n = 0$. Assume that $f : A \to B$ is a pseudo-morphism and assume that Uf is 0-monic. Given any algebra X and 1-cells $g, h : X \to A$ we have the following commutative diagram of sets.

$$\begin{array}{ccc} T\,\mathrm{Alg}(X, A)(g, h) & \xrightarrow{\quad f_* \quad} & T\,\mathrm{Alg}(X, B)(fg, fh) \\ \downarrow & & \downarrow \\ K(UX, UA)(Ug, Uh) & \xrightarrow[(Uf)_*]{} & K(UX, UB)(U(fg), U(fh)) \end{array}$$

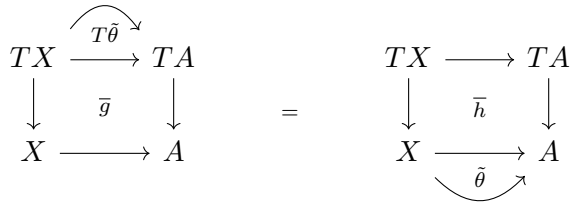
Since $(Uf)_*$ acts faithfully on 2-cells we see that the same is true for f_*. Finally let's do the case $n = 1$. Assume that $(f, \overline{f}) : (A, a) \to (B, b)$ is an algebra map and f is 1-monic. We already know that $(f, \overline{f})$ is 0-monic, so we only need to check that it acts surjectively on 2-cells. Say (X, x) is a third algebra, $(g, \overline{g})$ and $(h, \overline{h})$ are two pseudo-morphisms $(X, x) \to (A, a)$, and θ is a 2-cell of the following shape.



We need to produce a 2-cell $\tilde{\theta} : (g, \overline{g}) \Rightarrow (h, \overline{h})$ such that $f\tilde{\theta} = \theta$. By assumption there is a unique such cell downstairs in K, so we only need to check that this cell lifts to a 2-cell in $T\,\mathrm{Alg}$. This means we need to check that $\tilde{\theta}$ satisfies the coherence condition of a 2-cell between pseudo-algebra morphisms. A short calculation shows the following identity.



Using that $\overline{f}$ is invertible and that whiskering with f in K is faithful we see that this implies the following identity.

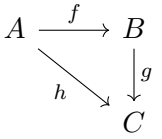


This is exactly the coherence condition for $\tilde{\theta}$ and so we are done. I will skip over the proof that U_s reflects 0-monicness, 1-monicness and isomorphisms, since it is very similar to the one we did for U, only that all the coherence cells are replaced by identifications. □

Proposition A.25. [37, §2.2] *An arrow in* Cat *is n-monic if and only if it is faithful, fully faithful or an equivalence in the usual sense respectively.*

Proof. Lemma A.23 already covers the case $n = 2$. Let us do the case $n = 0$ first. Assume that $f : A \to B$ is a faithful functor. Let C be any test category. Let $\alpha, \beta : g \to h$ be any two 2-cells between functors $g, h : C \to D$ and assume that $f\alpha = f\beta$. Then $f\alpha_c = g\beta_c$ for all objects $c \in C_0$, and hence $\alpha_c = \beta_c$ since f is faithful. It follows that $\alpha = \beta$, and we conclude that f is 0-monic. Conversely assume that $f : A \to B$ is 0-monic. Then $f_* : \mathrm{Cat}(\mathrm{pt}, A) \to \mathrm{Cat}(\mathrm{pt}, B)$ is faithful, but this functor is, up to isomorphisms, exactly f. Hence f is faithful. Let us do the case $n = 1$ next. Assume that $f : A \to B$ is a fully faithful functor and let C be a test category. We already established that $f_* : \mathrm{Cat}(C, A) \to \mathrm{Cat}(C, B)$ is faithful. Assume that we have two 1-cells $g, h : C \to A$ and a 2-cell $\theta : fh \Rightarrow gh$. For each object $c \in C_0$ one can find an arrow $\alpha_c : hc \to gc$ such that $f\alpha_c = \theta_c$. The faithfulness of f implies that the α_c are unique and one can check that they assemble into a natural transformation α such that $f\alpha = \theta$. For the other direction set again $C = \mathrm{pt}$. □

Lemma A.26. [37, §2.3] *Assume we are in a 2-category K and have a diagram of 1-cells such as the one drawn below which commutes up to an invertible 2-cell.*



When f and g are n-monic then so is h. When h and g are n-monic then so is f.

Proof. Since the definition of an n-monic map is representable, it is enough to check the assertion in the case $K = \mathrm{Cat}$. For Cat it is a quick calculation which can be done by hand. For example assume that both h and g are faithful. When $\alpha, \beta : h \Rightarrow g$ are two parallel 2-cells between 1-cells $g, h : T \to A$ such $f(\alpha) = f(\beta)$, then also $g(f(\alpha)) = g(f(\beta))$ and by faithfulness of $h = g \circ f$ we have that $\alpha = \beta$. Hence f is faithful. The other cases are left to the reader. □

A.5 Separation Conditions

Definition A.27. Let K be a 2-category and let W be a class of 1-cells in K. We say that a 0-cell C in K is n-W-separated when C perceives each morphism $w : A \to B$ from the class W as n-opmonic.

When W is a Grothendieck topology on a base category $\mathcal{S}$ then we can view each covering sieve of the Grothendieck topology as a 1-cell $S \to \Gamma$ in the 2-category $[\mathcal{S}^{op}, \mathrm{Cat}]$. Hence it makes sense to speak of n-W-separated indexed categories. A 2-$J(W)$-separated object in $\mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})$ is exactly a stack in the usual sense.

Lemma A.28. *The representable indexed categories $y\Gamma$ are semiflexible in $[\mathcal{S}^{op}, \mathrm{Cat}]$.*

Proof. We must show that the functor $[\mathcal{S}^{op}, \mathrm{Cat}](\Gamma, C) \to \mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})(J\Gamma, JC)$ is an equivalence for each indexed category C. Evaluating at the generic object of $y\Gamma$ gives us the following commutative diagram of functors.

$$\begin{array}{ccc} [\mathcal{S}^{op}, \mathrm{Cat}](\Gamma, C) & \longrightarrow & \mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})(J\Gamma, JC) \\ \downarrow & & \downarrow \\ C_\Gamma & = & C_\Gamma \end{array}$$

The 2-Yoneda lemma and the biYoneda lemma tell us that the two vertical functors are an isomorphism and an equivalence respectively. Hence we see that the upper horizontal map must be an equivalence. □

Proposition A.29. *Let $(\mathcal{S}, W)$ be a site, let C be a 0-cell in $[\mathcal{S}^{op}, \mathrm{Cat}]$ and assume that $n = 0$ or $n = 1$. Then JC is n-$J(W)$-separated in $\mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})$ if and only if C is n-W-separated in $[\mathcal{S}^{op}, \mathrm{Cat}]$.*

Proof. Given any covering sieve inclusion $S \to \Gamma$ we have a commutative diagram of functors which looks as follows.

$$\begin{array}{ccc} [\mathcal{S}^{op}, \mathrm{Cat}](\Gamma, C) & \longrightarrow & [\mathcal{S}^{op}, \mathrm{Cat}](S, C) \\ {\scriptstyle J}\downarrow & & \downarrow{\scriptstyle J} \\ \mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})(J\Gamma, JC) & \longrightarrow & \mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})(S, JC) \end{array}$$

The left vertical functor is an equivalence since Γ is semiflexible. The right vertical functor is 1-separated since J is a locally fully faithful 2-functor. One can now use lemma A.26 to get to the conclusion. □

Let us look again at the proof. When C is semicoflexible then the right vertical map is also an equivalence and JC will be 2-$J(W)$-separated if and only if C is 2-W-separated. But for non semicoflexible indexed categories there is a real difference between 2-separatedness in $[\mathcal{S}^{op}, \mathrm{Cat}]$ and in $\mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})$. The intuitive reason is that there are far less strict 1-cells $S \to C$ than non-strict ones if C is not semicoflexible. We will in general call an indexed category C such that JC is 2-$J(W)$-separated a stack.

Theorem A.30. *Let $(\mathcal{S}, W)$ be a site. The 2-functors J and R restrict to a biequivalence between the full sub-2-category of n-$J(W)$-separated 0-cells in $\mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})$ and the full sub-2-category of n-W-separated semicoflexible 0-cells in $[\mathcal{S}^{op}, \mathrm{Cat}]$.*

Proof. The 2-functors J and R restrict to a biequivalence between the sub-2-category cut out by the semicoflexible indexed categories and $\mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})$. This is because the components of the two transformations $1 \to RJ$ and $JR \to 1$ are equivalences when we restrict our attention to semicoflexible objects. The result is then a consequence of proposition A.29 and the paragraph below it. □

Proposition A.31. *Let $n = 0, 1$ and let M be a class of 1-cells in a sufficiently 2-cocomplete[22] 2-category K. Assume that $D : \mathcal{J} \to K$ is a diagram such that each object Dj is n-M-separated, and let $W : \mathcal{J} \to \mathrm{Cat}$ be any weight. Then the 2-limit $\lim_W D$ will be n-separated when it exists.*

Proof. We have to show that $\lim_W D$ perceives each of the 1-cells $m : A \to B$ in M as n-opmonic. There is a commuting diagram of functors which looks as follows.

$$\begin{array}{ccc} K(B, \lim_W D) & \xrightarrow{\quad m^* \quad} & K(A, \lim_W D) \\ \| \wr & & \| \wr \\ \lim_W K(B, D-) & \longrightarrow & \lim_W K(A, D-) \end{array}$$

The lower horizontal functor is the result of applying $\lim_W$ to a strict transformation of 2-functors $K(A, D-) \to K(A, D-)$ which is by assumption n-opmonic in each component. Since $n = 0, 1$ we can use theorem A.24, or more specifically the fact that the 2-functor $[\mathcal{J}, \mathrm{Cat}] \to \prod_{\mathrm{op}\,\mathcal{J}} K$ preserves and reflects 0- and 1-monicness, to deduce that the transformation $K(A, D-) \to K(B, D-)$ is itself n-monic. The 2-functor $\lim_W$ is a right 2-adjoint and hence preserves n-monicness. This shows that the lower horizontal functor is n-monic. In consequence so is the upper one. □

The proof does not work in the case $n = 2$ because the 2-functor $[\mathcal{J}, K] \to \prod_{\mathrm{ob}\,\mathcal{J}} K$ does not reflect equivalences. The result stays true though when the weight W is semiflexible. To show this let us first deal with homotopy limits.

Proposition A.32. *Assume that M is a class of morphisms in a 2-category K and that $D : \mathcal{J} \to K$ is a pseudo-functor such that each Dj is 2-M-separated. Then any weighted homotopy limit $\mathrm{holim}_W D$ of D will also be 2-M-separated.*

Proof. The proof is very similar to the proof of theorem A.31. We take a 1-cell $m : A \to B$ from M. We look at the following diagram of functors which commutes up to an invertible 2-cell.

$$\begin{array}{ccc} K(B, \mathrm{holim}_W D) & \xrightarrow{\quad m^* \quad} & K(A, \mathrm{holim}_W D) \\ \wr \| & & \wr \| \\ \mathrm{holim}_W K(B, D-) & \longrightarrow & \mathrm{holim}_W K(A, D-) \end{array}$$

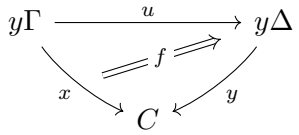
[22] K should be 2-cocomplete enough so that the 2-functor $U_s : [\mathcal{J}, K] \to \prod_{\mathrm{ob}\,\mathcal{J}} K$ has a right 2-adjoint. This applies in particular to $K = [\mathcal{S}^{op}, \mathrm{Cat}]$.

The lower horizontal map is obtained by applying holim_W to a pseudo-transformation which is entry-wise an equivalence. Hence the lower horizontal functor is an equivalence and it follows that m^* is an equivalence. □

Corollary A.33. *A semiflexible weighted limit of 2-M-separated objects is 2-M-separated.*

Proof. When the weight W is semiflexible, then $\lim_W D \simeq \mathrm{holim}_W D$ by lemma A.14. The result then follows from proposition A.32. □

Next we show that taking exponentials preserves n-separatedness. We have to do some preparation first. Assume that C is an object in $[\mathcal{S}^{op}, \mathrm{Cat}]$. Then we may form the category $\int C$ of elements of C which comes with a canonical projection map $\pi_C : \int C \to \mathcal{S}$. This is the Grothendieck construction. An object in the category of elements is a pair (Γ, x) with Γ an object in the base and x an object in C_Γ. A morphism $(\Gamma, x) \to (\Delta, y)$ is a pair (u, f) consisting of a map $u : \Gamma \to \Delta$ in the base together with a morphism $f : x \to u^* y$ in the fiber C_Γ. The functor π_C projects out the first entry. There is a canonical lax cocone below the diagram $y\pi : \int C \to [\mathcal{S}^{op}, \mathrm{Cat}]$.



Lemma A.34. *The lax cococone above expresses the object C as a lax conical 2-colimit of representables. We have that*

$$[\mathcal{S}^{op}, \mathrm{Cat}](C, A) \cong \mathrm{Lax}((\textstyle\int C)^{op}, \mathrm{Cat})(\Delta 1, [\mathcal{S}^{op}, \mathrm{Cat}](y\pi_C -, A))$$

holds 2-naturally.

Proof. Omitted. One has to spell out the data which a lax transformation from $y\pi_C$ to the constant diagram at A contains, and observe that it is exactly the same data as that of a strict 1-cell $C \to A$. □

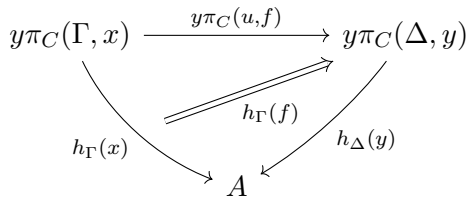
We can modify the weight to turn the lax 2-colimit into a strict one. We have that

$$C = \mathrm{colim}_{(\Delta 1)^\dagger}(y\pi_C) \tag{A.1}$$

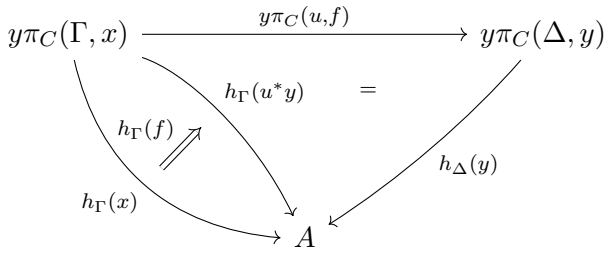
where $(-)^\dagger$ denotes the left 2-adjoint of $[(\int C)^{op}, \mathrm{Cat}] \to \mathrm{Lax}((\int C)^{op}, \mathrm{Cat})$. Lemma A.12 tells us that the modified weight $(\Delta 1)^\dagger$ is semiflexible. This is important because it means that the 2-colimit is, even though it is a strict 2-colimit, homotopical meaningful. In particular it makes no difference if we take the $(\Delta 1)^\dagger$-weighted homotopy colimit or the $(\Delta 1)^\dagger$-weighted strict colimit of $y\pi_C$.

$$\begin{aligned} C &\cong \mathrm{colim}_{(\Delta 1)^\dagger}(y\pi_C) \\ &\simeq \mathrm{hocolim}_{(\Delta 1)^\dagger}(y\pi_C) \end{aligned} \tag{A.2}$$

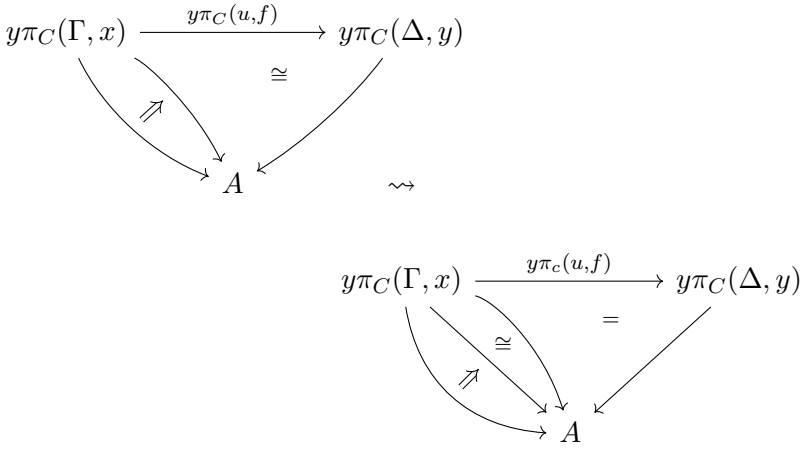
Remark A.35. Let me try to explain in more concrete terms what just happened. Lemma A.14 says that a strict 1-cell $h : C \to A$ in $[\mathcal{S}^{op}, \mathrm{Cat}]$ contains exactly the same data as the associated lax cocone below $y\pi_C$ pictured below.



In the second step we have replaced the non-trivial 2-cells between the legs by identities, but had to make the weight more complicated in return. One can picture the $(\Delta 1)^\dagger$-weighted cocone associated to $h : C \to A$ as follows.



We have moved the cells $h_\Gamma(f)$, which used to sit in between the legs of the cone, inside the legs. The object $\mathrm{hocolim}_{(\Delta 1)^\dagger}(y\pi_C)$ classifies cocones of a similar shape, only that the strict equalities are allowed to be coherently varying invertible 2-cells. The reason why the category of $(\Delta 1)^\dagger$-weighted pseudo-cones is equivalent to that of $(\Delta 1)^\dagger$-weighted strict cones is that the coherence isomorphisms of the pseudo-cones can be slurped into the 2-cells which sit inside the legs of the cocone. My internal picture looks roughly as follows:



The picture explains in intuitive terms why a weighted homotopy colimit over a semiflexible weight is equivalent to the strict 2-colimit over the same weight. ♢

Theorem A.36. [10, corollary 2.9] *Let $(\mathcal{S}, W)$ be a site and assume that C is n-W-separated in $[\mathcal{S}^{op}, \mathrm{Cat}]$. Then C^B will also be n-W-separated.*

Proof. Let us do the cases $n = 0, 1$ first. We have to show that the object C^B perceives every covering sieve inclusion $S \to y\Gamma$ as n-opmonic. The 1-cell $S \times y\Delta \to y\Gamma \times y\Delta$ is up to isomorphism also a covering sieve inclusion, since covering sieves in a Grothendieck topology are pullback stable. We will also use that the 2-functor $y\Gamma \times -$ commutes with weighted 2-colimits. This follows from the fact that it has $(-)^{y\Gamma}$ as a right 2-adjoint. Putting everything together we have the following strictly commuting diagram of functors.

$$\begin{array}{ccc}
[\mathcal{S}^{op}, \mathrm{Cat}](\Gamma, C^B) & \longrightarrow & [\mathcal{S}^{op}, \mathrm{Cat}](S, C^B) \\
\wr\| & & \wr\| \\
[\mathcal{S}^{op}, \mathrm{Cat}](\Gamma \times \mathrm{colim}_{(\Delta 1)^\dagger} y\pi_B, C) & \longrightarrow & [\mathcal{S}^{op}, \mathrm{Cat}](S \times \mathrm{colim}_{(\Delta 1)^\dagger} y\pi_B, C) \\
\wr\| & & \wr\| \\
\lim_{(\Delta 1)^\dagger} [\mathcal{S}^{op}, \mathrm{Cat}](\Gamma \times y\pi_B(-), C) & \longrightarrow & \lim_{(\Delta 1)^\dagger} [\mathcal{S}^{op}, \mathrm{Cat}](S \times y\pi_B(-), C)
\end{array}$$

The lower horizontal functor is the result of applying $\lim_{(\Delta 1)^\dagger}$ to a transformation of diagrams in $[\int B, \mathrm{Cat}]$ which is pointwise n-monic. Hence that transformation itself is n-monic and applying the 2-functor $\lim_{(\Delta 1)^\dagger}$ to it yields an n-monic cell in Cat. We can conclude that the upper horizontal functor is n-monic. The case $n = 2$ works in exactly the same way, but we have to use the homotopy colimit instead of the colimit. The isomorphisms become equivalences and the modified diagram commutes only up to invertible 2-cells. We are allowed to do that because the weight $(\Delta 1)^\dagger$ is semiflexible. □

Remark A.37. One should remember that C^B being 2-W-separated does not necessarily imply that $J(C^B)$ is 2-$J(W)$-separated in the 2-category $\mathrm{Hom}(\mathcal{S}^{op}, \mathrm{Cat})$. ♢

A.6 Reindexing Stability

We consider not a single 2-category but a system of 2-category $[(\mathcal{S}/\Gamma)^{op}, \mathrm{Cat}]$ connected by reindexing operations $u^* : [(\mathcal{S}/\Gamma)^{op}, \mathrm{Cat}] \to [(\mathcal{S}/\Delta)^{op}, \mathrm{Cat}]$ in the main part of the thesis. It is very important for us that the notions considered so far, namely 2-limits, semicoflexibility and n-separatedness, are stable under those reindexings. This is what we will check in this section. We fix a base site $(\mathcal{S}, W)$.

Definition A.38. Let $u : \Delta \to \Gamma$ be an object in $\mathcal{S}$. Reindexing along u for both indexed categories and weak indexed categories is defined via whiskering with the 1-cell $(\mathcal{S}/\Delta)^{op} \to (\mathcal{S}/\Gamma)^{op}$ induced by u. Explicitly this means that

$$(u^* C)(v) = C(uv) \tag{A.3}$$

and similar formulas hold for higher cells.

Proposition A.39. [21, B1.3.8] *The following diagram of 2-functors commutes at least up to an invertible pseudo-transformation.*

$$\begin{array}{ccc}
\mathrm{Fib}_{\mathcal{S}/\Gamma} & \xrightarrow{u^*} & \mathrm{Fib}_{\mathcal{S}/\Delta} \\
\int\uparrow & & \uparrow\int \\
\mathrm{Hom}((\mathcal{S}/\Gamma)^{op}, \mathrm{Cat}) & \xrightarrow[u^*]{} & \mathrm{Hom}((\mathcal{S}/\Delta)^{op}, \mathrm{Cat})
\end{array}$$

The upper 2-functor u^* is defined through (strict) 2-pullbacks along $\mathcal{S}/\Delta \to \mathcal{S}/\Gamma$ in Cat. *A similar result holds in the strict case.*

Proof. That change of base for pseudo-functors into Cat corresponds to pullbacks of fibrations is discussed in the elephant [21, B1.3.8]. The isomorphisms $u^* \int C \cong \int u^* C$ described in [21, B1.3.8] give the components of the isomorphism $\int u^* \cong u^* \int$, and we skip over writing down coherence cells and checking that everything is binatural. There is also a more abstract argument which looks as follows. According to [34] there is universal Cat-co-bundle $(\mathrm{Cat}_*, c)^{op} \to (\mathrm{Cat})^{op}$ such that the associated Grothendieck fibration to a pseudo-functor $C : \mathcal{S}/\Gamma \to \mathrm{Cat}^{op}$ can be obtained as a (strict) 2-pullback of the following form.

$$\begin{array}{ccc} \int C & \longrightarrow & (\mathrm{Cat}_*, c)^{op} \\ {\scriptstyle p}\downarrow & \lrcorner & \downarrow \\ \mathcal{S}/\Gamma & \longrightarrow & \mathrm{Cat}^{op} \end{array}$$

A detailed description of (Cat_*, c) can be found here [34]. The fact that precomposition with $\mathcal{S}/\Delta \to \mathcal{S}/\Gamma$ corresponds to (strict) pullbacks of Grothendieck fibrations is then a direct consequence of the fact that pullback squares patch. It is instructive to compare the second proof above to the standard proof of the fact that pulling back a subobject $m : S \hookrightarrow \Gamma$ along $u : \Delta \to \Gamma$ in a topos corresponds to precomposing the characteristic map $\phi_m : \Gamma \to \mathrm{Prop}$ of S with u. The idea is the same: pullbacks are replaced by composition when data "over" Γ is replaced by a map out of Γ into a universe which classifies that data. □

Even though the Grothendieck construction is compatible with base changes of all kinds, the same is not true for the two strictification operations L and R. When $F : \mathcal{E} \to \mathcal{S}$ is an arbitrary functor, then it will in general not be the case that $F^* RC \cong RF^* C$ as 0-cells in $[\mathcal{E}^{op}, \mathrm{Cat}]$. Similarly L does not commute in general with base change. Despite that, the good news is that R commutes with base change along functors $\mathcal{S}/\Delta \to \mathcal{S}/\Gamma$ induced by morphisms $u : \Delta \to \Gamma$ in the base.

Proposition A.40. *Let $u : \Delta \to \Gamma$ be a morphism in the base. Then the following diagram commutes at least up to an invertible 2-cell.*

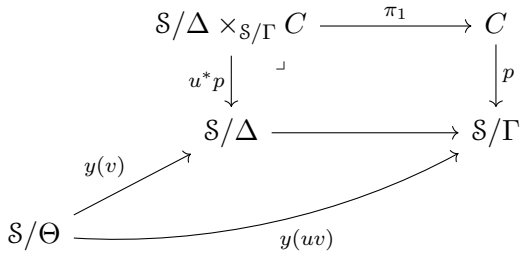
$$\begin{array}{ccc} \mathrm{Fib}_{\mathcal{S}/\Gamma} & \xrightarrow{u^*} & \mathrm{Fib}_{\mathcal{S}/\Delta} \\ {\scriptstyle \mathrm{Sp}_\Gamma}\downarrow & & \downarrow{\scriptstyle \mathrm{Sp}_\Delta} \\ [(\mathcal{S}/\Gamma)^{op}, \mathrm{Cat}] & \xrightarrow[u^*]{} & [(\mathcal{S}/\Delta)^{op}, \mathrm{Cat}] \end{array}$$

Proof. Let us fix a Grothendieck fibration $p : C \to \mathcal{S}/\Gamma$ and compare the fiberes of $u^* \mathrm{Sp}_\Gamma p$ with the fibers of $\mathrm{Sp}_\Delta u^* p$. Let $v : \Theta \to \Delta$ be any object in $\mathcal{S}/\Delta$.

$$\begin{aligned} (u^* \mathrm{Sp}_\Gamma p)(v) &= (\mathrm{Sp}_\Gamma p)(uv) \\ &= \mathrm{Fib}_{\mathcal{S}/\Gamma}(y(uv), p) \end{aligned} \tag{A.4}$$

In a similar way we can compute that $(\mathrm{Sp}_\Delta u^* p)(v) = \mathrm{Fib}_{\mathcal{S}/\Delta}(y(v), u^* p)$. It is not hard to compute that the total category of $y(v)$ is isomorphic to $\mathcal{S}/\Theta$, and that the

projection functor $\mathcal{S}/\Theta \to \mathcal{S}/\Delta$ is the functor induced by v. We get the following diagram in Cat.



The pullback is automatically a 2-pullback since Cat is 2-complete in the strict sense. This implies that post-composing with π_1 yields the isomorphism below.

$$\mathrm{Cat}_{/(\mathcal{S}/\Delta)}(y(v), u^*p) \cong \mathrm{Cat}_{/(\mathcal{S}/\Gamma)}(y(uv), p) \tag{A.5}$$

We just need to check that post-composing with π_1 sends fibered functors to fibered functors and hits all fibered functors in the target. Assume we have some functor $f : \mathcal{S}/\Theta \to \mathcal{S}/\Delta \times_{\mathcal{S}/\Gamma} C$ which sends $y(v)$-cartesian arrows to u^*p-cartesian arrows. Assume we have an $y(uv)$-cartesian arrow φ in $\mathcal{S}/\Theta$. Then [19, lemma 1.5.5] tells us that φ is in particular $y(v)$ cartesian and hence $f(\varphi)$ is u^*p-cartesian. It is also the case by [19, lemma 1.5.5] that $y(v)(\varphi)$ is $y(u)$-cartesian. Hence $f(\varphi)$ is $p\pi_1$-cartesian and in particular π_1-cartesian. This shows that post-composing with π_1 sends fibered functors to fibered functors. Conversely, assume that f is such that $\pi_1 f$ sends $y(uv)$-cartesian arrows to p-cartesian arrows. Let φ be a $y(u)$-cartesian arrow in $\mathcal{S}/\Theta$. Then $y(v)(\varphi)$ is also $y(uv)$-cartesian because $y(uv)$ is a discrete fibration. We get that $\pi_1 f(\varphi)$ is a p-cartesian arrow in C. From the characterisation of cartesian arrows in a pullback fibration [43, p. 16] we see that this already implies that $f(\varphi)$ is u^*p-cartesian in $\mathcal{S}/\Delta \times_{\mathcal{S}/\Gamma} C$ and we are done. □

Corollary A.41. *The strictification operations*

$$R_\Gamma : \mathrm{Hom}((\mathcal{S}/\Gamma)^{op}, \mathrm{Cat}) \to [(\mathcal{S}/\Gamma)^{op}, \mathrm{Cat}] \tag{A.6}$$

are compatible with reindexings along maps $u : \Delta \to \Gamma$ in the base.

Proof. We can write R_Γ as the composite $R_\Gamma = \mathrm{Sp}_\Gamma \int_\Gamma$, and we have already shown that both the Grothendieck construction and the 2-functors Sp_Γ commute with reindexings along maps $u : \Delta \to \Gamma$ in the base. □

Corollary A.42. *If $\Gamma \vdash C : \mathrm{Cat}$ is a semicoflexible category and $u : \Delta \to \Gamma$ is a morphism in the base, then $\Delta \vdash u^*C : \mathrm{Cat}$ is also semicoflexible.*

Proof. When C is semicoflexible then C is equivalent to $R_\Gamma JC$. It follows that u^*C is equivalent to $u^*R_\Delta JC \cong R_\Delta Ju^*C$, and this means that u^*C is semicoflexible. □

Proposition A.43. *The reindexing operations*

$$u^* : [(\mathcal{S}/\Gamma)^{op}, \mathrm{Cat}] \to [(\mathcal{S}/\Delta)^{op}, \mathrm{Cat}] \tag{A.7}$$

preserve 2-limits.

Proof. One way to see this is to use that 2-limits in $[(\mathcal{S}/\Gamma)^{op}, \mathrm{Cat}]$ and in the other 2-category are constructed pointwise. A second way to see this is to switch to split fibrations. Here it is easier to see that reindexing along u has left 2-adjoint Σ_u given by postcomposition with $\mathcal{S}/\Delta \to \mathcal{S}/\Gamma$.

$$\mathrm{Fib}^{sp}_{\mathcal{S}/\Gamma} \underset{u^*}{\overset{\Sigma_u}{\underset{\longrightarrow}{\overset{\longleftarrow}{\bot}}}} \mathrm{Fib}^{sp}_{\mathcal{S}/\Delta}$$

The picture above differs from our situation only by the 2-equivalence $\int_\Gamma$ and $\int_\Delta$ which commute with reindexings. It hence follows that $u^* : [(\mathcal{S}/\Gamma)^{op}, \mathrm{Cat}] \to [(\mathcal{S}/\Delta)^{op}, \mathrm{Cat}]$ also has a left 2-adjoint and hence preserves 2-limits. □

Proposition A.44. *Reindexing along maps $u : \Delta \to \Gamma$ preserves n-separatedness of indexed categories.*

Proof. Remember that we work with a fixed site $(\mathcal{S}, W)$. Assume that $\Gamma \vdash C : \mathrm{Cat}$ is n-separated, let $v : \Theta \to \Delta$ be an object $\mathcal{S}/\Delta$ and let $w_i : vw_i \to v$ be a cover.

$$\begin{array}{ccc} \Xi_i & \xrightarrow{w_i} & \Theta \\ & \searrow^{vw_i} & \downarrow{\scriptstyle v} \\ & & \Delta \end{array}$$

By definition of the induced topology on $\mathcal{S}/\Delta$ this means exactly that the $w_i : \Xi_i \to \Theta$ are a cover of Θ. One can now see we have the following commuting diagram of functors.

$$\begin{array}{ccc} (u^*C)(v) & \longrightarrow & \mathrm{Desc}(\{w_i : vw_i \to v\}_i, u^*C) \\ \| & & \| \wr \\ C(uv) & \longrightarrow & \mathrm{Desc}(\{w_i : uvw_i \to uv\}_i, C) \end{array}$$

The result we want follows from that diagram. □

B The Missing Rules

The aim of this appendix is to collect the missing rules of the category constructors listed in (1.29) and to explain their interpretation. Every category must come with object and morphism introduction and elimination rules, and with computation rules and uniqueness principles. Additionally there must be equations which explain how composition and the identities in the new category are defined.

B.1 Binary Products of Categories

$$\frac{\Xi \vdash X : C_0 \qquad \Xi \vdash Y : D_0}{\Xi \vdash (X, Y) : (C \times D)_0} \qquad \frac{\Xi \vdash N : C_1(X, Z) \qquad \Xi \vdash M : D_1(Y, W)}{\Xi \vdash (N, M) : (C \times D)_1((X, Y), (Z, W))}$$

$$\frac{\Xi \vdash P : (C \times D)_0}{\Xi \vdash P_1 : C_0} \qquad \frac{\Xi \vdash P : (C \times D)_0}{\Xi \vdash P_2 : D_0}$$

$$\frac{\Xi \vdash N : (C \times D)_1(X, Y)}{\Xi \vdash N_1 : C_0(X_1, Y_1)} \qquad \frac{\Xi \vdash N : (C \times D)_1(X, Y)}{\Xi \vdash N_2 : D_1(X_2, Y_2)} \tag{B.1}$$

The computation rules for both morphisms and objects are $(A, B)_1 \equiv A$ and $(A, B)_2 \equiv B$ and the uniqueness principle is $(A_1, A_2) \equiv A$. Additionally we have that $\mathrm{id}_{(X,Y)} \equiv (\mathrm{id}_X, \mathrm{id}_Y)$ and $(N, M) \circ (L, K) \equiv (M \circ L, M \circ K)$. The interpretation of $C \times D$ looks as follows:

$$[\![C \times D : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma) = [\![C : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma) \times [\![D : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma)$$

The other fibers are determined by the condition that interpretation and reindexings commute strictly. The indexed category $[\![C \times D : \mathrm{Cat}]\!]_\Gamma$ is a 2-product of $[\![C : \mathrm{Cat}]\!]_\Gamma$ and $[\![D : \mathrm{Cat}]\!]_\Gamma$ in the 2-category $[(\mathcal{S}/\Gamma)^{op}, \mathrm{Cat}]$. It is a prestack, a stack and semicoflexible respectively when both components are. This is because products are semiflexible 2-limits.

B.2 Slice Categories

The rules for the slice-category constructor are listed below. The computation rule and the uniqueness principle for morphisms are implicit in the notation.

$$\frac{\Xi \vdash Y : C_0 \qquad \Xi \vdash M : C_1(Y, X)}{\Xi \vdash (Y, M) : C/X}$$

$$\frac{\Xi \vdash L : C/X}{\Xi \vdash L_1 : C_0} \qquad \frac{\Xi \vdash L : C/X}{\Xi \vdash L_2 : C_1(L_1, X)}$$

$$\frac{\begin{array}{l}\Xi \vdash M : C_1(A, X) \\ \Xi \vdash N : C_1(B, X) \\ \Xi \vdash L : C_1(A, B) \qquad \Xi \vdash N \circ L \equiv M : C_1(A, X)\end{array}}{\Xi \vdash L : (C/X)_1((A, M), (B, N))}$$

$$\frac{\Xi \vdash L : (C/X)_1(A, B)}{\Xi \vdash L : C_1(A_1, B_1)} \qquad \frac{\Xi \vdash L : (C/X)_1(A, B)}{\Xi \vdash B_2 \circ L \equiv A_2 : C_1(A_1, X)} \tag{B.2}$$

The computation rules for objects are $(Y, M)_1 \equiv Y$ and $(Y, M)_2 \equiv Y$. The uniqueness principle for objects is $(A_1, A_2) \equiv A$. The identities are $\mathrm{id}_{(Y,M)} = \mathrm{id}_Y$. The composition rule is implicit in the notation. The interpretation of the slice-category constructor looks as follows.

$$[\![C/X : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma) = [\![C : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma)/[\![X : C_0]\!]_\Gamma \tag{B.3}$$

The other fibers are determined by the command that reindexings shall commute strictly with interpretation. The indexed category $[\![C/X : \mathrm{Cat}]\!]$ is a comma-object

in $[(\mathcal{S}/\Gamma)^{op}, \mathrm{Cat}]$.

$$\begin{array}{ccc} [\![C/X : \mathrm{Cat}]\!]_\Gamma & \longrightarrow & [\![C : \mathrm{Cat}]\!]_\Gamma \\ \downarrow & \swarrow & \| \\ \mathrm{pt} & \xrightarrow[[\![X:C_0]\!]_\Gamma]{} & [\![C : \mathrm{Cat}]\!]_\Gamma \end{array}$$

Since comma objects are flexible 2-limits we see that $[\![C/X : \mathrm{Cat}]\!]_\Gamma$ inherits n-separatedness and semicoflexibility from $[\![C : \mathrm{Cat}]\!]_\Gamma$.

B.3 Comma-Categories

$$\frac{\Xi \vdash A, B, C : \mathrm{Cat} \qquad \Xi \vdash F : (C^A)_0 \qquad \Xi \vdash G : (C^B)_0}{\Xi \vdash F/G : \mathrm{Cat}}$$

$$\frac{\Xi \vdash X : (F/G)_0}{\Xi \vdash X_1 : A_0} \qquad \frac{\Xi \vdash X : (F/G)_0}{\Xi \vdash X_2 : B_0}$$

$$\frac{\Xi \vdash L : (F/G)_1(X, Y)}{\Xi \vdash Y_3 \circ L_1 \equiv L_2 \circ X_3 : C_1(F_0X_1, G_0Y_2)} \qquad \frac{\Xi \vdash X : (F/G)_0}{\Xi \vdash X_3 : C_1(F_0X_1, G_0X_2)}$$

$$\frac{\Xi \vdash X : A_0 \qquad \Xi \vdash Y : B_0 \qquad \Xi \vdash N : C_1(FX, GY)}{\Xi \vdash (X, Y, N) : (F/G)_0}$$

$$\frac{\begin{array}{l}\Xi \vdash X, Y : (F/G)_0 \\ \Xi \vdash N : A_1(X_1, Y_1) \\ \Xi \vdash M : B_1(X_2, Y_2) \qquad \Xi \vdash Y_3 \circ F_1N \equiv G_1M \circ X_3 : C_1(F_0X_1, G_0Y_2)\end{array}}{\Xi \vdash (N, M) : (F/G)_1(X, Y)}$$

$$\frac{\Xi \vdash L : (F/G)_1(X, Y)}{\Xi \vdash L_1 : A_1(X_1, Y_1)} \qquad \frac{\Xi \vdash L : (F/G)_1(X, Y)}{\Xi \vdash L_2 : B_1(X_2, Y_2)} \tag{B.4}$$

The computation rules for the objects are $(X, Y, N)_1 \equiv X$, $(X, Y, N)_2 \equiv Y$ and $(X, Y, N)_3 \equiv N$. The uniqueness principle is $(X_1, X_2, X_3) \equiv X$. The computation rules for the morphisms are $(N, M)_1 \equiv N$ and $(N, M)_2 \equiv M$ and the uniqueness principle is $(N_1, N_2) \equiv N$. The identities are $\mathrm{id}_{(X,Y,N)} \equiv (\mathrm{id}_X, \mathrm{id}_Y)$ and the composition is componentwise. The interpretation of the internal comma-category constructor looks as follows.

$$[\![F/G : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma) = [\![F : C^A]\!]_\Gamma(\mathrm{id}_\Gamma)/[\![G : C^B]\!]_\Gamma(\mathrm{id}_\Gamma) \tag{B.5}$$

The second "/" denotes the construction of comma-objects in Cat. The prestack $[\![F/G : \mathrm{Cat}]\!]_\Gamma$ is a comma-object in the 2-categorical sense. Comma-objects are semiflexible 2-limits and we thus see that $[\![F/G : \mathrm{Cat}]\!]_\Gamma$ inherits n-separatedness and semicoflexibility from the indexed categories $[\![C : \mathrm{Cat}]\!]_\Gamma$, $[\![B : \mathrm{Cat}]\!]_\Gamma$ and $[\![A : \mathrm{Cat}]\!]_\Gamma$.

B.4 Small Products of Categories

$$\frac{\Xi \vdash A : \mathrm{Set} \qquad \Xi, x : A \vdash C : \mathrm{Cat}}{\Xi \vdash \prod_{x:A} C : \mathrm{Cat}} \qquad \frac{\Xi, x : A \vdash X : C}{\Xi \vdash (X)_{x:A} : (\prod_{x:A} C)_0}$$

$$\frac{\Xi \vdash X : (\prod_{x:A} C)_0 \qquad \Xi \vdash N : A}{\Xi \vdash X_N : C_0[N/x]} \qquad \frac{\Xi \vdash X, Y : (\prod_{x:A} C)_0 \qquad \Xi, x : A \vdash N : C_1(X_x, Y_x)}{\Xi \vdash (N)_{x:A} : (\prod_{x:A})_1(X, Y)}$$

$$\frac{\Xi \vdash L : (\prod_{x:A} C)_1(X, Y) \qquad \Xi \vdash N : A}{\Xi \vdash L_N : C_1(X_N, Y_N)} \tag{B.6}$$

The computation rule for both morphisms and objects is $((X)_{x:A})_N \equiv X[N/x]$ and the uniqueness principle is $(X_x)_{x:A} = X$. Composition and identities are defined pointwise. We make the following definition.

$$[\![\prod_{x:A} C : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma) = [\![C : \mathrm{Cat}]\!]_{\Gamma.A}(\mathrm{id}_{\Gamma.A}) \tag{B.7}$$

The other fibers are determined by the condition that reindexings commute strictly with the interpretation functions. We leave the interpretation of the object and morphism introduction and elimination rules to the reader. Even though we have not shown yet that $\prod_{x:A} C_x$ is always a prestack when C_x is one, we are allowed to already use the rules of $\prod_{x:A} C_x$ and its judgemental equations in our attempt to prove that fact. We just have to refrain from using (=)-extensionality.

Lemma B.1. *Sending* $\Gamma.A \vdash C : \mathrm{Cat}$ *to* $\Gamma \vdash \prod_A C$ *defines a right 2-adjoint of the 2-functor*

$$\pi_A^* : [(\mathcal{S}/\Gamma)^{op}, \mathrm{Cat}] \to [(\mathcal{S}/\Gamma.A)^{op}, \mathrm{Cat}]$$

which reindexes along the display map of A*.*

Proof. It is enough to check that $C \mapsto \prod_A C$ is a right adjoint of π_A^* on the 1-categorical level. We use the internal language to construct the unit and the counit of the adjunction. The $(\Gamma \vdash D : \mathrm{Cat})$th component of the unit is the following functor.

$$\Gamma \vdash I^0_{\mathrm{func}}(d.(d)_{x:A}, d.e.f.(f)_{x:A}) : \left((\prod_{x:A} D)^D\right)_0$$

The $(\Gamma, y : A \vdash C_y : \mathrm{Cat})$th component of the counit is the following functor.

$$\Gamma, y : A \vdash I^0_{\mathrm{func}}(c.c_y, c.d.f.f_y) : \left(C_y^{\prod_{x:A} C_x}\right)_0$$

One can check that the triangle identities hold by using the judgemental equations of the constructor $\prod_A(-)$. □

Proposition B.2. *If* $\Gamma.A \vdash C : \mathrm{Cat}$ *is semicoflexible, then* $\Gamma \vdash \prod_A C : \mathrm{Cat}$ *is also semicoflexible.*

Proof. Compute:

$$\begin{aligned}
\mathrm{Hom}((\mathcal{S}/\Gamma)^{op}, \mathrm{Cat})(JB, J\textstyle\prod_A C) &\cong [(\mathcal{S}/\Gamma)^{op}, \mathrm{Cat}](LJB, \textstyle\prod_A C) \\
&\cong [(\mathcal{S}/\Gamma.A)^{op}, \mathrm{Cat}](\pi_A^* LJB, C) \\
&\simeq \mathrm{Hom}((\mathcal{S}/\Gamma.A)^{op}, \mathrm{Cat})(J\pi_A^* LJB, JC) \\
&\cong \mathrm{Hom}((\mathcal{S}/\Gamma.A)^{op}, \mathrm{Cat})(\pi_A^* JLJB, JC) \\
&\simeq \mathrm{Hom}((\mathcal{S}/\Gamma.A)^{op}, \mathrm{Cat})(\pi_A^* JB, JC) \\
&\cong \mathrm{Hom}((\mathcal{S}/\Gamma.A)^{op}, \mathrm{Cat})(J\pi_A^* B, JC) \\
&\simeq [(\mathcal{S}/\Gamma.A)^{op}, \mathrm{Cat}](\pi_A^* B, C) \\
&\cong [(\mathcal{S}/\Gamma)^{op}, \mathrm{Cat}](B, \textstyle\prod_A C) \qquad \text{(B.8)}
\end{aligned}$$

Unfortunately reindexing does not commute with L, and this made the computation a little bit longer than it would be otherwise. We used several times that C is semicoflexible and once that $JL \simeq 1$. In principle one should check that the equivalence constructed above is up to an invertible 2-cell the action of J, but we skip over that. □

Proposition B.3. *If* $\Gamma.A \vdash C : \mathrm{Cat}$ *is n-separated, then so is* $\Gamma \vdash \prod_A C : \mathrm{Cat}$.

Proof. The indexed category $\prod_A C$ perceives a covering sieve inclusion $w : S \to y(u)$ as n-opmonic if and only if C perceives $\pi_A^* w : \pi_A^* S \to \pi_A^* y(u)$ as n-opmonic. But $\pi_A^* w$ is up to isomorphism a covering sieve inclusion of $\pi_A^* y(u) \cong y(\pi_A^* u)$ in $[(\mathcal{S}/\Gamma.A)^{op}, \mathrm{Cat}]$ and hence the result follows. □

B.5 The Category of Small Categories

$$\frac{\Xi \vdash C : \mathrm{sCat}_0}{\Xi \vdash \mathrm{Ob}_C : \mathrm{Set}_0} \qquad \frac{\Xi \vdash C : \mathrm{sCat}_0 \qquad \Xi \vdash X, Y : \mathrm{Ob}_C}{\Xi \vdash \mathrm{Mor}_C(X, Y) : \mathrm{Set}_0}$$

$$\frac{\Xi \vdash C : \mathrm{sCat}_0 \qquad \Xi \vdash X : \mathrm{Ob}_C}{\Xi \vdash \mathrm{id}_X : \mathrm{Mor}_C(X, X)} \qquad \frac{\Xi \vdash X, Y : \mathrm{Ob}_C \qquad \Xi \vdash N : \mathrm{Mor}_C(X, Y)}{\Xi \vdash \mathrm{id}_Y \circ N \equiv N : \mathrm{Mor}_C(X, Y)}$$

$$\frac{\Xi \vdash C : \mathrm{sCat}_0 \qquad \Xi \vdash X, Y, Z : \mathrm{Ob}_C \qquad \begin{array}{l} \Xi \vdash N : \mathrm{Mor}_C(X, Y) \\ \Xi \vdash M : \mathrm{Mor}_C(Y, Z) \end{array}}{\Xi \vdash M \circ_C N : \mathrm{Mor}_C(X, Z)}$$

$$\frac{\textit{similar assumptions}}{\Xi \vdash (M \circ_C N) \circ_C L \equiv M \circ_C (N \circ_C L) : ...}$$

$$\frac{\Xi \vdash X, Y : \mathrm{Ob}_C \qquad \Xi \vdash N : \mathrm{Mor}_C(X, Y)}{\Xi \vdash N \circ \mathrm{id}_X \equiv N : \mathrm{Mor}_C(X, Y)}$$

$$\frac{\begin{array}{ll}\Xi \vdash O : \mathrm{Set} & \\ \Xi, x, y : O \vdash M : \mathrm{Set} & \Xi, x, y, z : O, f : M, g : M[y/x, z/y] \vdash C : M[z/y] \\ \Xi, x : O \vdash I : M[x/y] & \ulcorner \textit{the data satisfies the equations of a category} \urcorner\end{array}}{\Xi \vdash (O, x.y.M, x.I, x.y.z.f.g.C) : \mathrm{sCat}_0} \tag{B.9}$$

These rules above are just the object introduction and elimination rules. Writing down the morphism introduction and elimination rules, the two sets of computation rules and uniqueness principles and the rules which explain composition and identities in sCat is left to the reader. The category

$$[\![\mathrm{sCat} : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma) \tag{B.10}$$

is the category of type-theoretic internal categories in $\mathcal{S}/\Gamma$. What kind of data such an internal category contains is well-explained by the object introduction rule above. We call them "type-theoretic internal categories" because the types of objects and morphisms are equipped with extra reindexing data. The category $[\![\mathrm{sCat} : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma)$ is equivalent to the category of internal categories in $\mathcal{S}/\Gamma$ in the categorical sense. One can use the fact that $[\![\mathrm{Set} : \mathrm{Cat}]\!]_\Gamma$ is 1-separated to derive that the same is true for $[\![\mathrm{sCat} : \mathrm{Cat}]\!]_\Gamma$. We will not do this here to safe space.

B.6 The Enlargement of a Small Category

$$\frac{\Xi \vdash C : \mathrm{sCat}_0}{\Xi \vdash [C] : \mathrm{Cat}} \qquad \frac{\Xi \vdash X : \mathrm{Ob}_C}{\Xi \vdash X : [C]_0}$$

$$\frac{\Xi \vdash X : [C]_0}{\Xi \vdash X : \mathrm{Ob}_C} \qquad \frac{\Xi \vdash N : \mathrm{Mor}_C(X, Y)}{\Xi \vdash N : [C]_1(X, Y)}$$

$$\frac{\Xi \vdash N : [C]_1(X, Y)}{\Xi \vdash N : \mathrm{Mor}_C(X, Y)} \tag{B.11}$$

The computation rules, the uniqueness principle, the identities and composition are all implicit in the notation. One can check that $[C]$ is prestack by using that Set is one. From the external perspective $[C]$ corresponds to the indexed category which is typically associated to an internal category in $\mathcal{S}$. The construction $[C]$ is called as externalisation in fibered category theory [19, Definition 7.3.1], but the name is not really fitting in our context, since we also consider indexed categories above $\mathcal{S}$ to be large internal categories in the world of $\mathcal{S}$.

B.7 The Category of Families

$$\frac{\Xi \vdash C : \mathrm{Cat}}{\Xi \vdash \mathrm{Fam}(C) : \mathrm{Cat}} \qquad \frac{\Xi \vdash A : \mathrm{Set}_0 \qquad \Xi, x : A \vdash X : C_0}{\Xi \vdash (X)_{x:A} : \mathrm{Fam}(C)_0}$$

$$\frac{\Xi \vdash X : \mathrm{Fam}(C)_0}{\Xi \vdash X_1 : \mathrm{Set}_0} \qquad \frac{\Xi \vdash X : \mathrm{Fam}(C)_0 \qquad \Xi \vdash N : X_1}{\Xi \vdash X_N : C_0}$$

$$\frac{\Xi \vdash X, Y : \mathrm{Fam}(C)_0 \qquad \Xi \vdash u : \mathrm{Set}_1(X_1, Y_1) \qquad \Xi, x : X_1 \vdash N : C_1(X_x, Y_{u(x)})}{\Xi \vdash (u, (N)_{x:X_1}) : \mathrm{Fam}(C)_1(X, Y)}$$

$$\frac{\Xi \vdash L : \mathrm{Fam}(C)_1(X, Y)}{\Xi \vdash L_1 : \mathrm{Set}_1(X_1, Y_1)} \quad \frac{\Xi \vdash L : \mathrm{Fam}(C)_1(X, Y) \qquad \Xi \vdash N : X_1}{\Xi \vdash L_N : C_1(X_N, Y_{L_1 N})} \tag{B.12}$$

The object computation rule is $((X)_{x:A})_N \equiv X[N/x]$ and the uniqueness principle is $(X_x)_{x:X_1} \equiv X$. The morphism computation rules are $(u, (N)_{x:A})_1(M) \equiv uM$ and $(u, (N)_{x:A})_M \equiv N[M/x]$. The uniqueness principle is $(L_1, (L_x)_{x:X_1}) \equiv L$ for a morphism L which has $X : \mathrm{Fam}(C)$ as domain. The identities are $\mathrm{id}_{(X)_{x:A}} \equiv (\mathrm{id}_A, (\mathrm{id}_X)_{x:A})$ and the composition is defined through the following equation.

$$(v, (M)_{y:B}) \circ (u, (L)_{x:A}) \equiv (v \circ u, (M[u(x)/y] \circ L)_{x:A}) \tag{B.13}$$

The category $[\![\mathrm{Fam}(C) : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma)$ has as objects pairs (A, X) where A is an internal set at stage Γ and X is an object of $[\![\pi_A^* C : \mathrm{Cat}]\!]_{\Gamma.A}(\mathrm{id}_{\Gamma.A})$. A morphism in $[\![\mathrm{Fam}(C) : \mathrm{Cat}]\!]_\Gamma$ is a pair $(u, N) : (A, X) \to (B, Y)$ where u is a morphism $u : \pi_A \to \pi_B$ in the slice $\mathcal{S}/\Gamma$ and N is a morphism of the following type.

$$N : [\![X : \pi_A^* C_0]\!]_{\Gamma.A} \to u^* [\![Y : \pi_B^* C_0]\!]_{\Gamma.B} \tag{B.14}$$

One can check that $[\![\mathrm{Fam}(C) : \mathrm{Cat}]\!]_\Gamma$ is a prestack when $[\![C : \mathrm{Cat}]\!]_\Gamma$ is a prestack. The internal construction $\mathrm{Fam}(-)$ is almost the same construction as the construction of the family fibration from fibered category theory [43, Definition 6.2]. If p is a Grothendieck fibration, then there is a natural weak equivalence $\mathrm{Fam}(\mathrm{Sp}\, p) \to \mathrm{Sp}(\mathrm{Fam}\, p)$ which forgets a little bit of reindexing information. The second Fam is the construction of fibered category theory [43, Definition 6.2].

B.8 Finite Categories

The rules of finite categories need some extra explanation. There is a category constructor for every external finite category J. Let us fix some such finite meta-category J. There are object and morphism constructors in our language for every actual object X and every actual morphism $N : X \to Y$ in the meta-category J.

$$\frac{}{X : J}\ (X \in J_0) \qquad\qquad \frac{}{N : J_1(X, Y)}\ (N \in J_1(X, Y)) \tag{B.15}$$

We use the "$\in$" symbol in the name of the rules to indicate that those are expressions of our meta-language. When J has 235 objects and 34672 morphisms, then $J : \mathrm{Cat}$ will have 34907 object and morphism introduction rules in total. There are no direct object and morphism elimination rules, instead there are rules which allow us to define functors and transformations pointing out of J. The number of assumptions which a particular elimination rule needs also depends on the total number of objects and morphisms in the actual category J.

$$\frac{\begin{array}{l}\Xi \vdash N_x : C_0 \quad (x \in J_0) \\ \Xi \vdash M_f : C_1(N_x, N_y) \quad (f \in J_1(x, y)) \\ \Xi \vdash C : \mathrm{Cat} \qquad \Xi \Vdash \ulcorner \textit{the data satisfies the equations of a functor} \urcorner\end{array}}{\Xi \vdash E_J^0((N_x)_{x \in J_0}, (M_f)_{f \in J_1}) : (C^J)_0} \tag{B.16}$$

The indices are not part of the language. Their purpose is just to indicate that the elimination rule eats as many terms as there are morphisms and objects in J. The rules (B.16) are in praxis only ever useful for reasonably small finite categories J. Writing down the missing elimination rule, appropriate computation rules, the uniqueness principles and rules which explain composition and identities in $J : \mathrm{Cat}$ is left to the reader.

The objects in the category $[\![J : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma)$ are the objects of the finite category J. A morphism $X \to Y$ in $[\![J : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma)$ is an equivalence class represented by a cover $u_i : \Delta_i \to \Gamma$ together with a morphism $f_i : X \to Y$ from J for each i. Two such families (u_i, f_i) and (v_j, g_j) represent the same morphism when for every i, j and w such that $u_i w = v_j w$ we have that either $f_i = g_j$ in J or the domain of Θ is covered by the empty cover. All other fibers of $[\![J : \mathrm{Cat}]\!]_\Gamma$ are determined by the condition that reindexing commutes strictly with interpretation. The indexed category $[\![J : \mathrm{Cat}]\!]_\Gamma$ has the following universal property with respect to prestacks C.

$$\begin{array}{ccc} \underline{J} & \longrightarrow & [\![J : \mathrm{Cat}]\!]_\Gamma \\ & \searrow & \downarrow \exists! \\ & & C \end{array}$$

Here $\underline{J}$ denotes the constant indexed category associated to J. This is because we have secretly applied the 1-step sheafification to the presheaf $u \mapsto \underline{J}(u^*X, u^*Y)$ to define the morphisms in $[\![J : \mathrm{Cat}]\!]_\Gamma$, which means that $[\![J : \mathrm{Cat}]\!]_\Gamma$ is the prestackification of $\underline{J}$. The strict 2-universal property above allows us to interpret the object and morphism elimination rules of $J : \mathrm{Cat}$, and the interpretation of the introduction rules should also be clear.

As a word of warning: If our base $\mathcal{S}$ has a geometric morphism $\gamma : \mathcal{S} \to \mathrm{Fin}$ to the external category of finite sets in our meta-language, then we can view J as an internal category in Fin an apply γ^* to it to get an internal category in $\mathcal{S}$. If we equip $\gamma^* J$ with extra reindexing data, then we get an internal category $\vdash \gamma^* J : \mathrm{sCat}_0$ in the empty context and hence also an indexed category $\vdash [\gamma^* J] : \mathrm{Cat}$. That indexed category need not be the same indexed category as the $\vdash J : \mathrm{Cat}$ we defined in this section. In some situations $\vdash [\gamma^* J] : \mathrm{Cat}$ will be a stack, but $\vdash J : \mathrm{Cat}$ is almost never a stack. The difference is that the objects $\Gamma \vdash X : [\gamma^* J]_0$ can vary over Γ, while the objects $\Gamma \vdash X : J_0$ are constant. There is an internal comparison functor $\vdash F : [\gamma^* J]^J$ which will, for reasonable models $(\mathcal{S}, W)$, be fully faithful and essentially surjective on objects from the internal perspective, but it will not be an equivalence in general. This also demonstrates that $J : \mathrm{Cat}$ does not typically satisfy 1-definite choice.

B.9 Full Subcategories

$$\frac{\Xi \vdash C : \mathrm{Cat} \qquad \Xi, x : C_0 \vdash \phi : \mathrm{Prop}_0}{\Xi \vdash \mathrm{Full}(C, x.\phi) : \mathrm{Cat}} \qquad \frac{\Xi \vdash X : C_0 \qquad \Xi \Vdash \phi[X/x]}{\Xi \vdash X : \mathrm{Full}(C, x.\phi)_0}$$

$$\frac{\Xi \vdash X : \mathrm{Full}(X, x.\phi)_0}{\Xi \vdash X : C_0} \qquad \frac{\Xi \vdash X : \mathrm{Full}(C, x.\phi)_0}{\Xi \Vdash \phi[X/x]}$$

$$\frac{\Xi \vdash X, Y : \mathrm{Full}(C, x.\phi)_0 \qquad \Xi \vdash N : C_1(X, Y)}{\Xi \vdash N : \mathrm{Full}(C, x.\phi)_1(X, Y)} \qquad \frac{\Xi \vdash N : \mathrm{Full}(C, x.\phi)_1(X, Y)}{\Xi \vdash N : C_1(X, Y)} \tag{B.17}$$

All judgemental equations are implicit in the notation. The category $[\![\mathrm{Full}(C, x.\phi) : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma)$ is the full subcategory of $[\![C : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma)$ cut out by those objects X for which $\Gamma \Vdash \phi[X/x]$ holds. The indexed category $[\![\mathrm{Full}(C, x.\phi) : \mathrm{Cat}]\!]_\Gamma$ is 0- or 1-separated when $[\![C : \mathrm{Cat}]\!]_\Gamma$ is.

Proposition B.4. *The category* $\mathrm{Full}(C, x.\phi)$ *is a stack if* C *is a stack.*

Proof. Without loss of generality assume we are at the global stage. Say we have descent data (X_i, θ_{ij}) in $\mathrm{Full}(C, x.\phi)$ for a cover $u_i : \Delta_i \to \Gamma$ that we like to glue. We can glue it in C. To show that the resulting object X lies in $\mathrm{Full}(C, x.\phi)$ we just have to show that $\Gamma \Vdash \phi[X/x]$ holds. We have that $\Delta_i \Vdash u_i^*\phi[X_i/x]$ holds for each i. The X_i are isomorphic to the u_i^*X and truth in the internal language is invariant under isomorphisms. This is something that we gain from not having a propositional equality of objects. We thus see that $\Delta_i \Vdash u_i^*(\phi[X/x])$ and $\Gamma \Vdash \phi[X/x]$ now follows from the locality of the forcing relation. □

Proposition B.5. *The indexed category* $\mathrm{Full}(C, x.\phi)$ *is semicoflexible if* C *is semicoflexible.*

Proof. If we want to strictify a pseudo-morphism $A \rightsquigarrow \mathrm{Full}(C, x.\phi)$ then we can start by strictifying the composite $A \rightsquigarrow \mathrm{Full}(C, x.\phi) \to C$. The resulting arrow factors through $\mathrm{Full}(C, x.\phi)$ because it differs from the original one only by an invertible 2-cell. □

B.10 The Opposite of a Category

$$\frac{\Xi \vdash C : \mathrm{Cat}}{\Xi \vdash C^{op} : \mathrm{Cat}} \qquad \frac{\Xi \vdash X : C_0}{\Xi \vdash X : C_0^{op}} \qquad \frac{\Xi \vdash N : C_1(X, Y)}{\Xi \vdash N : C_1^{op}(Y, X)} \tag{B.18}$$

All judgemental equations are implicit in the notation. The interpretation is straightforward.

$$[\![C^{op} : \mathrm{Cat}]\!](\mathrm{id}_\Gamma) = ([\![C : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma))^{op} \tag{B.19}$$

If $[\![C : \mathrm{Cat}]\!]_\Gamma$ is n-separated, then so is $[\![C^{op} : \mathrm{Cat}[\![_\Gamma$, and the same holds for semicoflexibility. The opposite of an indexed category is directly related to the corresponding construction in fibered category theory. We have that $\mathrm{Sp}(p^{op}) \cong (\mathrm{Sp}\, p)^{op}$ when p is a Grothendieck fibration. The first $(-)^{op}$ is the construction on fibrations described in [43, section 5].

B.11 Categories of Algebras for an Endofunctor

$$\frac{\Xi \vdash C : \mathrm{Cat} \qquad \Xi \vdash F : (C^C)_0}{\Xi \vdash F\,\mathrm{alg} : \mathrm{Cat}} \qquad \frac{\Xi \vdash A : C_0 \qquad \Xi \vdash a : C_1(F_0 A, A)}{\Xi \vdash (A, a) : F\,\mathrm{alg}_0}$$

$$\frac{\Xi \vdash X : F\,\mathrm{alg}_0}{\Xi \vdash X_1 : C_0} \qquad \frac{\Xi \vdash X : F\,\mathrm{alg}_0}{\Xi \vdash X_2 : C_1(F_0 X_1, X_1)}$$

$$\frac{\begin{array}{l}\Xi \vdash X, Y : F\,\mathrm{alg}_0 \\ \Xi \vdash N : C_1(X_1, Y_1) \qquad \Xi \vdash Y_2 \circ F_1 N \equiv N \circ X_2 : C_1(F_0 X_1, Y_1)\end{array}}{\Xi \vdash N : (F\,\mathrm{alg})_1(X, Y)}$$

$$\frac{\Xi \vdash N : (F\,\mathrm{alg})_1(X, Y)}{\Xi \vdash C_1(X_1, Y_1)} \qquad \frac{\Xi \vdash N : (F\,\mathrm{alg})_1(X, Y)}{\Xi \vdash Y_2 \circ F_1 N \equiv N \circ X_2 : C_1(F_0 X_1, Y_1)} \tag{B.20}$$

The computation rules for the objects are $(A, a)_1 \equiv A$ and $(A, a)_2 \equiv a$. The uniqueness principle is $(X_1, X_2) \equiv X$. The computation rule and the uniqueness principle for morphisms are implicit in the notation. The identities are $\mathrm{id}_{(A,a)} \equiv \mathrm{id}_A$ and the composition rule is also implicit in the notation. The definition of the category $[\![F\,\mathrm{alg} : \mathrm{Cat}]\!]_\Gamma(\mathrm{id}_\Gamma)$ can be read of from the rules. Let $\mathcal{J}$ be category with an object an a single endomorphism. Let $D : \mathcal{J} \to [(\mathcal{S}/\Gamma)^{op}, \mathrm{Cat}]$ be the functor which chooses the endomorphism $[\![F : (C^C)_0]\!]_\Gamma$ of $[\![C : \mathrm{Cat}]\!]_\Gamma$. Then $[\![F\,\mathrm{alg} : \mathrm{Cat}]\!]_\Gamma$ is the lax 2-limit of D. Since lax conical 2-limits can also be expressed as 2-limits over a flexible weight we see that $[\![F\,\mathrm{alg} : \mathrm{Cat}]\!]_\Gamma$ inherits semicoflexibility and n-separatedness from $[\![C : \mathrm{Cat}]\!]_\Gamma$.

C Internal Characterisations of Various Set Constructors

Appendix C collects the missing internal characterisations of the type constructors from chapter 3. A model is as always a lex site $(\mathcal{S}, W)$ such that the self-indexing of $\mathcal{S}$ is a stack.

C.1 Binary Product Types

The rules of the binary product type can be found in [19, p. 139]. We follow the pattern laid out in chapter 3.

Proposition C.1. *If we add product set constructors to the language $L(Sign)$, then we can construct an internal right adjoint prod of the diagonal $\Delta : \mathrm{Set} \to \mathrm{Set} \times \mathrm{Set}$.*

Proof. This is an informal internal proof in the language $L(Sign)$ + product types. The right adjoint *prod* takes a pair of sets (X, Y) to $X \times Y$. It takes a pair of functions $(f, g) : (X, Y) \to (X', Y')$ to the function $f \times g$ which sends (x, y) to (fx, fy'). All that is internally definable using the rules of product types, the functor introduction rule and ν-abstraction. The (X, Y)th component of the counit

$\varepsilon : \Delta prod \Rightarrow 1$ is the pair (π_X, π_Y) where π_X and π_Y denote the functions $\nu p.p_1 : X \times Y \to X$ and $\nu p.p_2 : X \times Y \to Y$ respectively. To see that $\varepsilon_{(X,Y)}$ satisfies the correct universal property assume we are given a third set Z and two functions $f : Z \to X$ and $g : Z \to Y$. Then we can define a third function $\nu z.(fz, gz) : Z \to X \times Y$, and one can check that the composites of that function with π_X and π_Y are f and g respectively. To see that this is the only such function assume that h is any other such function. If $z : Z$ is a point then $hz = ((hz)_1, (hz)_2) = (\pi_X hz, \pi_Y hz) = (fz, gz)$, and $h = \nu z.(fz, gz)$ follows from function extensionality. □

Proposition C.2. *If we add the axiom "*Set *merely has binary products" to the language $L(Sign)$, then we can internally construct a right adjoint of Δ and make sense of the binary product type rules.*

Proof. An standard proof shows that "Set has binary products" is equivalent to "each comma category $\Delta/(X, Y)$ has a terminal object". We may apply the constructor $(\mathbf{R}, \varepsilon)$ to get an actual right adjoint. Using the actual adjoint we can make sense of the binary product type rules as follows. We interpret $X \times Y$ as $\mathbf{R}_\Delta(X, Y)$. When $p : X \times Y$ is a point then we interpret $p_1 : X$ and $p_2 : Y$ as the composites of p with the two projection morphisms of the categorical product. Given $x : X$ and $y : Y$ we let $(x, y) : X \times Y$ denote the unique point $\mathbf{1} \to X \times Y$ whose composites with the projections are x and y respectively. One can internally derive the computation rules and the uniqueness principle. □

Corollary C.3. *Let $(\mathcal{S}, W)$ be a model. The category with families $CwF(\mathcal{S})$ admits a binary product type constructor if and only if the following holds.*

$$(\mathcal{S}, W) \models \ulcorner \text{Set } \textit{has binary products} \urcorner \tag{C.1}$$

Proof. If $CwF(\mathcal{S})$ admits binary products, then we can us them to construct a right adjoint of the diagonal. Internally this means that "Set actually has binary products", and in particular this implies that Set has binary products. Conversely, when it is internally true that Set has binary products, then we can get actual binary products by using $(\mathbf{R}, \varepsilon)$. We have just seen how $(R_\Delta, \varepsilon_\Delta)$ can be used to produce binary product types for the category with families $CwF(\mathcal{S})$ associated with $\mathcal{S}$. □

C.2 Extensional Identity Types

The rules of extensional identity types can be found here [30, p. 11].

Proposition C.4. *When we add extensional identity types to the language $L(Sign)$ then we can internally show that "*Set *has equalizers of pairs of points".*

Proof. This is an informal internal proof in the language $L(Sign)$ + extensional identity types. Take a pair $x, y : \mathbf{1} \rightrightarrows A$ of points of an internal set A : Set. We form the set $\mathrm{Eq}_A(x, y)$ by using the extensional identity type introduction rule. There is a unique map $\mathrm{Eq}_A(x, y) \to \mathbf{1}$ and we want to show that this map is an equalizer of x and y. It is enough to check its universal property against $\mathbf{1}$ because

the singelton set can detect limits in Set. Assume there is $p : \mathbf{1} \to \mathbf{1}$ such that $xp = yp$. Then $x = y$ and hence $x \equiv y$ by extensionality. This allows us to derive $\mathrm{refl}(x) : \mathrm{Eq}_A(x, y)$ by using the intro rule of the extensional identity type. Hence p factors through $\mathrm{Eq}_A(x, y) \to \mathbf{1}$. Since the rules of the extensional identity type tell us that $\mathrm{Eq}_A(x, y)$ is a subsingleton set we see that $\mathrm{refl}(x)$ is the unique morphism which factors p. □

Proposition C.5. *If we add extensional identity types to the language $L(Sign)$ and also have access to strong Σ-types, then we can internally derive that "*Set *has equalizers".*

Proof. This is an informal internal proof. Given two functions $f, g : A \rightrightarrows B$ construct the set $\sum_{x:A} \mathrm{Eq}_B(fx, gx)$. We claim that $e := \nu(x, p).x : \sum_{x:A} \mathrm{Eq}_B(fx, gx) \to A$ is an equalizer of f and g. The function e is a monomorphism since each $\mathrm{Eq}_B(fx, gx)$ is a subsingleton set. If $h : C \to A$ is a function such that $fh = gh$, then we can factor h through e via the map $\nu c.(hc, \mathrm{refl}(fhc)) : C \to \sum_{x:A} \mathrm{Eq}_B(fx, gx)$. The uniqueness of the factorisation follows from the factor that e is monic. □

Proposition C.6. *If we add the axiom that "*Set *has equalizers" to the language $L(Sign)$, then we can derive the rules of extensional identity types from our language.*

Proof. We merely have equalizers, but we can choose some by applying the $(\mathbf{R}, \varepsilon)$ constructor to the diagonal $\Delta : \mathrm{Set} \to \mathrm{Set}^{\bullet \rightrightarrows \bullet}$. When $x : A$ and $y : A$ are two points, then we can make sense of the rules of the extensional identity type as follows. The object introduction rule gets interpreted as the chosen equalizer of x and y. If $x \equiv y$, then the equalizer is the maximal subobject of $\mathbf{1}$ and we can hence find a term $\mathrm{refl}(x) : \mathbf{1} \to \mathrm{Eq}_A(x, y)$. If we have a point $p : \mathbf{1} \to \mathrm{Eq}_A(x, y)$, then this means that $\mathrm{Eq}_A(x, y) \to \mathbf{1}$ has a section and is hence an isomorphism. This means that $x = y$ and in consequence that $x \equiv y$ by extensionality. This explains another of the rules of the extensional identity type. The final rule that $p \equiv q$ for all $p, q : \mathrm{Eq}_A(x, y)$ follows from the fact that every equalizer is a monomorphism and that $\mathrm{Eq}_A(x, y)$ is hence a subsingleton set. □

So, in the presence of strong Σ-types, adding extensional equality type constructors to the language $L(Sign)$ is exactly the same as adding the mere axiom that "Set has equalizers". Since every model admits strong Σ-types we get the following result.

Corollary C.7. *Let $(\mathcal{S}, W)$ be a model. The category with families $CwF(\mathcal{S})$ associated with $\mathcal{S}$ has extensional equality types if and only if the following is true.*

$$(\mathcal{S}, W) \models \ulcorner \text{Set } \textit{has equalizers} \urcorner \tag{C.2}$$

C.3 The Empty Type

We skip over the singleton type because nothing interesting happens there. We deal with the empty type next. We do not use the rules from [30, p. 13] but instead

slightly modified ones.

$$\frac{}{\varnothing : \mathrm{Set}_0} \qquad \frac{\Gamma \vdash A : \mathrm{Set}_0 \qquad \Gamma \vdash P : \varnothing}{\Gamma \vdash \mathrm{ind}_\varnothing(P) : A}$$

$$\frac{\Gamma \vdash A : \mathrm{Set}_0 \qquad \Gamma \vdash N, M : A \qquad \Gamma \vdash P : \varnothing}{\Gamma \vdash N \equiv M : A} \tag{C.3}$$

The first two rules appear in [30] and the third rule is derivable from the first two in the presence of extensional equality types.

Proposition C.8. *If we add the rules of the empty type (C.3) to the language $L(Sign)$, then we can internally derive that "*Set *has a strict initial object".*

Proof. This is an informal internal proof. We start by showing that the identity type is initial in Set. If A is any other set, then we can define a function $\nu x.\, \mathrm{ind}_\varnothing(x) : \varnothing \to A$. If $g, f : \varnothing \to A$ are any two functions and $x : \varnothing$ is a point of the domain, then we can conclude that $fx = gx$ by the third rule. This means that f and g agree on all points, and so they must be equal by function extensionality. Next let us check that $\varnothing$ is strict. Assume we have any map $f : A \to \varnothing$. Let $g : \varnothing \to A$ denote the unique map out of $\varnothing$ into A. It is clear that $fg = \mathrm{id}_\varnothing$, so we only need to check that $gf = \mathrm{id}_A$. Given a point $x : A$ we have that $fx : \varnothing$ and can hence conclude that $gfx = x$ by the third rule of the empty type. It follows that $gf = \mathrm{id}_A$ by function extensionality. □

Proposition C.9. *If we add the mere axiom that "*Set *has a strict initial object" to the language $L(Sign)$, then we can derive the rules (C.3) of the empty type in our language.*

Proof. We merely have an initial object in Set, but we can choose one by applying the $(\mathbf{L}, \eta)$-constructor to the functor Set $\to$ pt. We need to show that the initial object, which we denote by $\varnothing : \mathrm{Set}_0$, satisfies the rules listed in (C.3). If A is a second set and $x : \mathbf{1} \to \varnothing$ is a point, then we can interpret $\mathrm{ind}_\varnothing(x) : \mathbf{1} \to A$ to be the composite of x with the unique function $\varnothing \to A$. To check the third rule assume we have two points $x, y : \mathbf{1} \to A$ of an internal set and also a point $p : \mathbf{1} \to \varnothing$. The morphism p is invertible because the initial object is strict. We have that $xp^{-1} = yp^{-1}$ by the uniqueness clause in the definition of an initial object. Composing with p on both sides gives us $x = y$ and hence $x \equiv y$ by extensionality. □

Corollary C.10. *Let $(\mathcal{S}, W)$ be a model. The category with families $CwF(\mathcal{S})$ associated with $\mathcal{S}$ has an empty type if and only if the following holds.*

$$(\mathcal{S}, W) \models \ulcorner \text{Set } \textit{has a strict initial object} \urcorner \tag{C.4}$$

C.4 Weak Sum Types

$$\frac{\Gamma \vdash A : \mathrm{Set}_0 \qquad \Gamma \vdash B : \mathrm{Set}_0}{\Gamma \vdash A + B : \mathrm{Set}_0} \qquad \frac{\Gamma \vdash N : A}{\Gamma \vdash \kappa N : A + B} \qquad \frac{\Gamma \vdash M : B}{\Gamma \vdash \kappa' M : A + B}$$

$$\frac{\Gamma \vdash C : \mathrm{Set}_0 \qquad \Gamma, x : A \vdash Q : C \qquad \Gamma, y : B \vdash R : C}{\Gamma, z : A + B \vdash \mathrm{elim}_+(z, x.Q, y.R) : C}(weak) \tag{C.5}$$

Additionally there are computation rules and a uniqueness principle. Stronger elimination principles can be derived from the weak ones in the presence of strong Σ-types and Π-types.

Proposition C.11. *When we add the rules (C.5) to the language $L(Sign)$ then we can internally derive that "*Set *has binary coproducts".*

Proof. Given two sets A, B we can form the set $A+B$. We can use the introduction rules to get two functions $\nu a.\kappa a : A \to A + B$ and $\nu b.\kappa' b : B \to A + B$ which we denote by i_A and i_B respectively. If C is a third set and $f : A \to C$, $g : B \to C$ are two functions, then we can define a third function $\nu z.\, \mathrm{elim}_+(z, x.fx, y.gy) : A+B \to C$. Let us denote that function by h. One can check that $hi_A x = fx$ and $hi_B y = gy$ for all points $x : A$ and $y : B$ by using the computation rules of the sum type. By function extensionality it follows that $hi_A = hi_B$. When $h' : A + B \to C$ is any other function which satisfies that condition and $z : A + B$ is any point, then we can make the following computation.

$$\begin{aligned} h'z &= \mathrm{elim}_+(z, x.h'\kappa x, y.h'\kappa' y) \\ &= \mathrm{elim}_+(z, x.h'i_A x, y.h'i_B y) \\ &= \mathrm{elim}_+(z, x.fx, y.gy) \end{aligned} \tag{C.6}$$

This shows that h' and h are pointwise equal, and function extensionality implies that they are equal morphisms. □

Proposition C.12. *If we add the mere axiom that "*Set *has binary coproducts" to the axioms of $L(Sign)$, then we can derive the rules of the weak sum types from the rules of our language.*

Proof. We merely have binary coproducts, but we can choose some by applying the $(\mathbf{L}, \eta)$ constructor to the diagonal $\Delta : \mathrm{Set} \to \mathrm{Set} \times \mathrm{Set}$. We can write $A + B$ to denote the set $\mathbf{L}_\Delta(A, B)$. If $x : A$ is a point, then we can interpret $\kappa x : A + B$ to be the composition of $x : \mathbf{1} \to A$ with the coprojection $i_A : A \to A + B$. The second introduction rule can be interpreted in a similar way. If we have a third set C : Set, dependent families of points $q_x : C$ and $r_y : C$ indexed by $x : A$ and $y : B$ respectively and additionally a point $z : A + B$, then we can write $\mathrm{elim}_+(z, x.q_x, y.r_y) : C$ to denote the composite of z with the unique map $A + B \to C$ induced by $\nu x.q_x : A \to C$ and $\nu y.r_y : B \to C$. The computation rules and the uniqueness principle can be internally derived. □

Corollary C.13. *Let $(\mathcal{S}, W)$ be a model. The category with families $CwF(\mathcal{S})$ associated to $\mathcal{S}$ admits weak sum types if and only if the following is true.*

$$(\mathcal{S}, W) \models \ulcorner \text{Set } \textit{has binary coproducts} \urcorner \tag{C.7}$$

C.5 Π-Types

The rules of Martin-Löf's Π-types can be found here [30, p. 15].

Proposition C.14. *If we add* Π*-types to the rules of* $L(Sign)$*, then we can internally derive that "*Set *has small coproducts".*

Proof. Given a family of sets B_x indexed by a set $x : A$ we construct the set $\prod_{x:A} B_x$. The elimination rule of the Π-type allows us to construct a projection map $\nu p.p(x) : \prod_{x:A} B_x \to B_x$ for each $x : A$ which we denote by π_x. If C is any other set and $f_x : C \to B_x$ is an $(x : A)$-indexed family of functions, then we can construct a function $\nu c.(\lambda x.f_x c) : C \to \prod_{x:A} B_x$ by using the term-introduction rule of the Π-type. Let us denote that function by h. One can check that $\pi_x h = f_x$ for every $x : A$. If h' is any other function which satisfies that and $z : C$ is a point, then we can make the following computation.

$$\begin{aligned} h'z &= \lambda x.(h'z)(x) \\ &= \lambda x.\pi_x(h'z) \\ &= \lambda x.f_x z \end{aligned} \tag{C.8}$$

This shows that h and h' agree on points and hence are equal functions. □

Proposition C.15. *If we add the mere axiom "*Set *has small products" to the language* $L(Sign)$*, then we can derive the rules of the* Π*-type from our language.*

Proof. We apply the $(\mathbf{L}, \eta)$ constructor to the functor $\Delta : \mathrm{Set}^{op} \to \mathrm{Fam}(\mathrm{Set}^{op})$ to choose small coproducts in Set. We denote those coproducts by $\prod_{x:A} B_x$. If $p : \mathbf{1} \to \prod_{x:A} B_x$ is a point and $x : A$, then we can interpret $p(x)$ to be the composite of p with the projection $\pi_x : \prod_{x:A} B_x \to B_x$. This explains the term elimination rule. If we have a family $b_x : \mathbf{1} \to B_x$ of points indexed by $x : A$, then we can interpret $\lambda x.b_x : \mathbf{1} \to \prod_{x:A} B_x$ as the unique point p for which $\pi_x \circ p = b_x$ for all $x : A$. This explains the introduction rule, and the computation rule and the uniqueness principle can also be checked internally. □

Corollary C.16. *Let* $(\mathcal{S}, W)$ *be a model. The category with families* $CwF(\mathcal{S})$ *associated to* $\mathcal{S}$ *admits* Π*-types if and only if the following holds.*

$$(\mathcal{S}, W) \models \ulcorner \text{Set } \textit{as small products} \urcorner \tag{C.9}$$

C.6 Bracket Types

The rules of the bracket type constructor come from the paper [2]. We will from now on assume that strong Σ-types and finite limit constructors are already part of the language $L(Sign)$. As we have seen this is the same as adding the mere axioms that "Set is infinitary extensive" and "Set has finite limits" to the language. Both hold in every model.

Proposition C.17. *If we add the bracket type rules to the language* $L(Sign)$*, then we can internally derive that "*Set *has coequalizers of projections* $\pi_1, \pi_2 : A \times A \rightrightarrows A$*, and those coequalizers are subsingleton sets".*

Proof. This is an informal internal argument in the language $L(Sign)$ + (bracket types). Given a set A, form the set $[A]$ and define a function $\eta_A : A \to [A]$ which sends $x : A$ to $[x] : [A]$. We like to show that this function is the coequalizer of the two projection maps our of $A \times A$. So let B be a second set and let $f : A \to B$ be a function such that $f\pi_1 = f\pi_2$. When $x : A$ and $y : A$ are any two points of A, then $(x, y) : A \times A$ is a point and we have that $fx = f\pi_1(x, y) = f\pi_2(x, y) = fy$. Given a point $z : [A]$ we can use the elimination rule of the bracket type to produce a point $\mathrm{elim}_{[-]}(z, x.fx) : B$ of B. We get a function $\nu z.\,\mathrm{elim}_{[-]}(z, x.fx) : [A] \to B$ which we denote by h. The computation rule of the bracket type allows us to make the following computation for every $a : A$.

$$\begin{aligned} h\eta_A a &= \mathrm{elim}_{[-]}([a], x.fx) \\ &= fa \end{aligned} \tag{C.10}$$

We get $h\eta_A = f$ by function extensionality. To see that h is the only function with that property, assume h' is a second such function. We use the uniqueness principle of the bracket type to make the following computation for each point $z : [A]$.

$$\begin{aligned} h'z &= \mathrm{elim}_{[-]}(z, x.h'[x]) \\ &= \mathrm{elim}_{[-]}(z, x.h'\eta_A x) \\ &= \mathrm{elim}_{[-]}(z, x.fx) \end{aligned} \tag{C.11}$$

This shows that h and h' are pointwise equal, and hence they must be the same function. That the coequalizer $[A]$ is a subsingleton set is one of the rules of the bracket type. □

Proposition C.18. *If we add bracket types to the language $L(Sign)$, then we can internally derive that "*Set *is regular".*

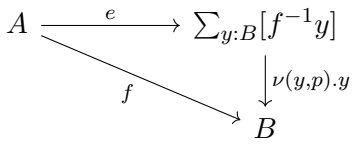
Proof. This is an informal internal argument in the language $L(Sign)$ + (bracket types). We need to show that we can factor every function in Set as an extremal epi followed by a monomorphism, and we also need to show that the factorisation is pullback stable. We can assume that we have already chosen pullbacks in Set for every span of functions. We will need those pullbacks in the construction of the image factorisation below. Start with some function $f : A \to B$. We form the set $\sum_{y:B}[f^{-1}y]$ where $f^{-1}y$ denotes the pullback indicated below.

$$\begin{array}{ccc} f^{-1}y & \xrightarrow{i_y} & A \\ \downarrow & \lrcorner & \downarrow f \\ \mathbf{1} & \xrightarrow[y]{} & B \end{array}$$

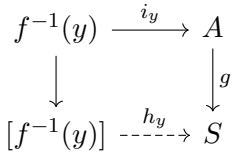
Since each $[f^{-1}y]$ is a subsingleton set we see that the projection

$$\nu(y, p).y : \textstyle\sum_{y:B}[f^{-1}y] \to B \tag{C.12}$$

is a monomorphism. Let $e : A \to \sum_{y:B}[f^{-1}y]$ be the function which sends a point $x : A$ to the pair $(fx, [p_x]))$ where p_x is the unique point $p_x : f^{-1}y$ for which $i_y \circ p_x = x$. We can factor f as follows.



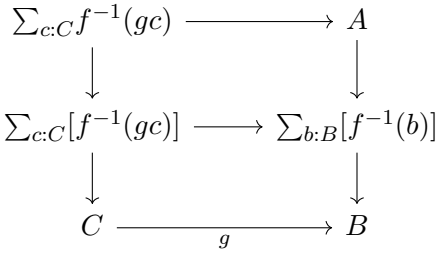
We need to check that e is an extremal epimorphism. Assume that $m : S \to \sum_{y:B}[f^{-1}y]$ is a monomorphism such that e factors as $e = m \circ g$ for some function g. Fix a point $y : B$. The map $g \circ i_y$ is constant since $m \circ g \circ i_y = e \circ i_y$ is constant. We get an induced morphism h_y as shown below.



The maps h_y together induce a function $h : \sum_{y:B}[f^{-1}(y)] \to S$. We can make the following computation:

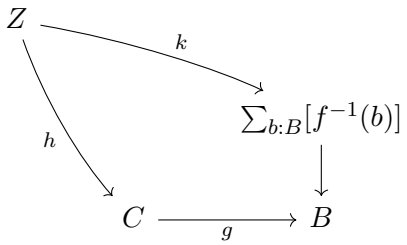
$$\begin{aligned} m \circ h_y \circ [a] &= m \circ \mathrm{elim}_{[\,]}([a], x.gi_yx) \\ &= m \circ g \circ i_y \circ a \\ &= e \circ i_y \circ a \\ &= (y, [a]) \end{aligned} \tag{C.13}$$

This shows that $m \circ h \circ (y, p) = (y, p)$ for all points $(y, p) : \sum_{y:B}[f^{-1}(y)]$ and hence that h is a section of m. A monomorphism with a section is an isomorphism. We have successfully shown that e is extremal. Next, let us check the pullback stability of the factorisation. Assume we have a second map $g : C \to B$. We can draw the following diagram.

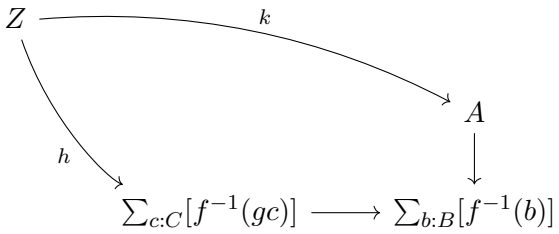


The upper left vertical map is again an extremal epimorphism, and the lower left vertical map is monic. So if we can show that both rectangles are pullbacks then we are done. Let us start with the lower rectangle. Say we are in the following

situation:



If $z : Z$ is a point of Z, then $h(z)$ is a point of C and $k(z)_2$ is a point of $[f^{-1}(ghz)]$. We thus get a point $(hz, (kz)_2) : \sum_{c:C}[f^{-1}(gc)]$. We can use ν-abstraction to define a function $Z \to \sum_{c:C}[f^{-1}(gc)]$ and one can check that this map is the one and only map which fits. Next we have to check that the upper rectangle is a pullback. Assume we are in the situation depicted below.



Given a point $z : Z$ we have $(hz)_1 : C$ and the fact that the diagram commutes means that $g((hz)_1) = fkz$. This means that kz factors through the monomorphism $f^{-1}(g((hz)_1)) \to A$. Let us denote the induced point by $\overline{kz}$. We get a point $((hz)_1, \overline{kz}) : \sum_{c:C} f^{-1}(gc)$. The ν-abstraction rule gives us a function $Z \to \sum_{c:C} f^{-1}(gc)$ and one can check that this function is the unique function which fits. This finishes the proof that Set is regular. □

Proposition C.19. *If we add the mere axiom that "*Set *has coequalizers of projections $\pi_1, \pi_2 : A \times A \rightrightarrows A$ and the coequalizers are subsingleton sets" to the language $L(Sign)$, then we can derive the bracket type rules.*

Proof. This is an informal internal argument. Let us denote the inclusion from subsingleton sets to sets by $i : \mathrm{SubS} \to \mathrm{Set}$. Let us start by showing that all of the comma categories A/i have an initial object. If A is some set then we can take the coequalizer $q : A \to B$ of the pair $\pi_1, \pi_2 : A \times A \rightrightarrows A$. The set B is a subsingleton set by assumption. When C is any other subsingleton set and $f : A \to C$ any function of sets, then $f\pi_1 = f\pi_2$ and we hence see that f factors uniquely through q. This shows that the pair (B, q) is initial in the comma category A/i. Using the $(\mathbf{L}, \eta)$ constructor we can choose a left adjoint $[-]$ of I together with a unit $\eta_A : A \to [A]$. We like to check that $[A]$ satisfies the rules of the bracket type. If $a : A$ is a point, then we can interpret $[a]$ to be the composite $\eta_A \circ a$. For the elimination rule assume we have a set B, a family of points $b_x : B$ indexed by $x : A$ such that $b_x = b_y$ for all $x, y : A$ and additionally a point $z : [A]$. It holds that $(\nu x.b_x)\pi_1 = (\nu x.b_x)\pi_2$ and hence we get an induced map $h : [A] \to B$. We can interpret $\mathrm{elim}_{[-]}(z, x.b_x)$ to be the composite of $z : \mathbf{1} \to [A]$ with $h : [A] \to B$. The rule $p = q : [A]$ for all $p, q : [A]$ follows directly from the fact that $[A] \to \mathbf{1}$ is monic, and the computation rule and the uniqueness principle can also be derived. □

Proposition C.20. *We can internally show that "*Set *is regular" implies "*Set *has coequalizers of projections* $\pi_1, \pi_2 : A \times A \rightrightarrows A$ *and the coequalizers are subsingleton sets".*

Proof. This is an informal internal proof. If Set is regular and A is a set, then we can factorise the unique map $t : A \to \mathbf{1}$ as $t = me$ where m is monic e is an extremal epi. General facts about regular categories, whose proofs are intuitionistic and can be carried out in our internal language, tell us that $e : A \to B$ is the effective coequalizer of the kernel pair $\pi_1, \pi_2 : A \times A \rightrightarrows A$ of $A \to \mathbf{1}$ in Set. The fact that m is monic means that B is automatically a subsingleton set. □

Corollary C.21. *Let* $(\mathcal{S}, W)$ *be a model. The following are equivalent.*

(i) The category with families $CwF(\mathcal{S})$ *associated to* $\mathcal{S}$ *admits bracket types.*

*(ii) The internal statement "*Set *has coequalizers of projections* $\pi_1, \pi_2 : A \times A \rightrightarrows A$ *and the coequalizers are subsingleton sets" holds in the model* $(\mathcal{S}, W)$*.*

*(iii) The internal statement "*Set *is regular" holds in the model* $(\mathcal{S}, W)$*.*

C.7 Effective Quotient Types

The rules of effective quotient types can be found here [30, p. 15]. We will at some point have to choose quotients in Set, and our only means of doing that is the $(\mathbf{L}, \eta)$ constructor. We thus need an internal large category at the global stage which classifies the input data of a categorical quotient, and we need an internal functor of which "constructing quotients" is an adjoint. The category in question is the category Setoid of setoids. We did not introduce it as an official category constructor yet, and I do not feel like going through all the necessary steps since we have already done that many times. I will informally describe how the objects and morphisms of Setoid look like from the internal perspective and leave it to the reader to provide the missing details and rules.

Definition C.22 (internal)**.** An object in Setoid is a set A together with a pair of maps $r_1, r_2 : R \rightrightarrows A$ such that the induced map $R \to A \times A$ is monic and the pair (r_1, r_2) satisfies the axioms of a categorical equivalence relation expressed in diagrammatic form. A morphism of setoids $f : (A, R) \to (B, S)$ is a map of sets $f : A \to B$ which respects the equivalence relations.

Definition C.23 (internal)**.** There is a functor $\Delta :$ Set $\to$ Setoid which equips a set A with the diagonal of $A \times A$.

Remark C.24. We could have also defined Setoid in a more type theoretic way. A setoid would be a set A together with a $((x, y) : A \times A)$-indexed family of subsingleton sets $R(x, y)$ which satisfy the three standard axioms of an equivalence relation. The resulting category of "type-theoretic" setoids is internally equivalent to the category of categorical setoids which we defined above. ♢

Proposition C.25. *If we add the effective quotient type rules to the language* $L(Sign)$*, then we can internally show that "*Set *has effective quotients of internal equivalence relations".*

Proof. This is an informal internal argument in the language $L(Sign)$ + (effective quotient types). We work with fixed choice of pullbacks in Set. Assume we have an internal equivalence relation $r_1, r_2 : R \rightrightarrows A$ on a set A. Given any two points $x : A$, $y : A$ we may form the following set.

$$\begin{array}{ccc} R(x,y) & \xrightarrow{i_{(x,y)}} & R \\ \downarrow & \lrcorner & \downarrow \\ \mathbf{1} & \xrightarrow[(x,y)]{} & A \times A \end{array}$$

The sets $R(x, y)$ are subsingleton sets because $R \to A \times A$ is a monomorphism. One can check that the $R(x, y)$ satisfies the rules of a type-theoretic equivalence relation. We can apply the quotient type introduction rule [30, p. 15] to form a new set A/R. The introduction rule allows us to define a map $\nu x.[x] : A \to A/R$ which we denote by η. The map η_A is a map of setoids $\eta : (A, R) \to (A/R, \Delta_{A/R})$ because of the eq-Q rule in [30, p. 15]. Let $f : A \to B$ be any map of sets for which $fr_1 = fr_2$. Given $x : A$, $y : A$ and $p : R(x, y)$ we can look at the following diagram to see that $(fx, fy) = (fr_1 i_{(x,y)} p, fr_2 i_{(x,y)} p)$ and hence that $fx = fy$.

$$\begin{array}{ccccccc} & & R(x,y) & \longrightarrow & R & & \\ & \overset{p}{\nearrow} & \downarrow & & \downarrow & & \\ \mathbf{1} & \xrightarrow[\mathrm{id}_{\mathbf{1}}]{} & \mathbf{1} & \xrightarrow[(x,y)]{} & A \times A & \xrightarrow[f \times f]{} & B \times B \end{array}$$

The (weak) elimination rule of the effective quotient types allows us to produce a function $\nu z.\,\mathrm{elim}(z, x.fx) : A/R \to B$ which we denote temporarily by h. One can check that h is the one and only function which satisfies $\eta h = f$. This shows that η is indeed the coequalizer of r_1 and r_2. To check effectiveness we need to show that the diagram below is a pullback.

$$\begin{array}{ccc} R & \xrightarrow{r_1} & A \\ {\scriptstyle r_2}\downarrow & & \downarrow{\scriptstyle \eta} \\ A & \xrightarrow[\eta]{} & A/R \end{array}$$

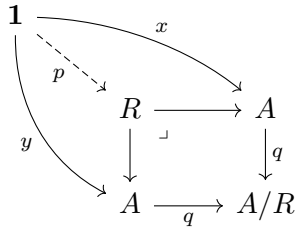
We have that $R \cong \sum_{(x,y):A\times A} R(x, y)$ and hence it doesn't hurt to replace the square with the following square.

$$\begin{array}{ccc} \sum_{(x,y):A\times A} R(x,y) & \xrightarrow{\nu(x,y,p).x} & A \\ {\scriptstyle \nu(x,y,p).y}\downarrow & & \downarrow{\scriptstyle \eta_A} \\ A & \xrightarrow[\eta_A]{} & A/R \end{array}$$

It is enough to check the universal property of that square against the test object $\mathbf{1}$, since $\mathbf{1}$ can detect limits in Set. But for $\mathbf{1}$ the universal property is a direct consequence of the effectiveness rule in [30, p. 15]. □

Proposition C.26. *If we add the mere axiom that "*Set *has effective quotients of internal equivalence relations" to the language $L(Sign)$, then we can derive the effective quotient type rules from our language.*

Proof. This is an informal internal argument. A standard argument shows that "Set has quotients of internal equivalence relations" if and only if "each comma category $(A, R)/\Delta$ has an initial object". Here Δ denotes the functor $\Delta : \mathrm{Set} \to \mathrm{Setoid}$ which equips a set with its diagonal. We can use the constructor $(\mathbf{L}, \eta)$ to choose quotients in Set. Let A be a set and let $R(x, y)$ be a family of subsingleton sets indexed by $x, y : A$. We form the set $R = \sum_{x:A} \sum_{y:A} R(x, y)$ and we get two natural projection maps $r_1, r_2 : R \rightrightarrows A$. The induced map $R \to A \times A$ is monic because all the $R(x, y)$ are subsingleton sets. If the family $R(x, y)$ satisfies the assumptions of the quotient type introduction rule, then $r_1, r_2 : R \rightrightarrows A$ will be an equivalence relation in the categorical sense. We can thus from the effective coequalizer $q : A \to A/R$ of r_1 and r_2 and make sense of the object introduction rule in [30, p. 15]. Given some $a : A$ we can understand $[a] : A/R$ as the composite of a with q. For the elimination rule let us start with the weak version where the target type does not depend on the input. Assume we have a set B and a family of points $b_x : B$ indexed by $x : A$ such that $b_x = b_y$ whenever $R(x, y)$ is inhabited. It is then clear that $(\nu x.b_x)r_1 = (\nu x.b_x)r_2$ and we hence get an induced map $A/R \to B$ which we denote by h. Given a point $z : A/R$ we can interpret $\mathrm{elim}(z, x.b_x) : B$ as the composite of $z : A/R$ with h. This is explains the weak elimination rule of the quotient type, and the strong elimination rule can be derived from the weak one in the presence of strong Σ-types. The computation rule and the uniqueness principle can also be checked internally. To check effectiveness assume we have two points $x, y : A$ such that $[x] = [y]$. Then we get an induced point as shown below, because q is an effective quotient.



This implies that the fiber $R(x, y)$ is inhabited. This explains the effectiveness rule from [30, p. 15] and we are done. □

Corollary C.27. *Let $(\mathcal{S}, W)$ be a model. The category with families $CwF(\mathcal{S})$ associated to $\mathcal{S}$ has effective quotient types if and only if the following is true.*

$$(\mathcal{S}, W) \models \ulcorner \mathrm{Set} \text{ has effective quotients of internal equivalence relations} \urcorner \quad \text{(C.14)}$$

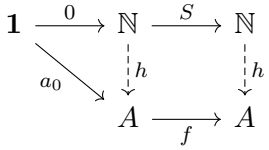
C.8 The Natural Number Type

The rules of the natural number type can also be found in [30, p. 12]. We will again be in the situation that we have to choose a particular natural number object in Set. This has to be done with the constructor $(\mathbf{L}, \eta)$. The most convenient way to do this is to assume that we already have access to weak sum types. Because then we can internally define the endofunctor $F_0 A = \mathbf{1} + A$ on Set and form the category $F\,\mathrm{alg}$ of F-algebras. The category $F\,\mathrm{alg}$ happens to be a semicoflexible stack because it is the lax limit of the endofunctor F, and the natural number object happens to be an initial F-algebra. We can thus apply $(\mathbf{L}, \eta)$ to the unique functor

$F\,\mathrm{alg} \to \mathrm{pt}$ and in that way choose a natural number object at the global stage provided it is merely true that there is one. This way we can avoid having to deal with the coherence problem by hand!

Proposition C.28. *If we add a natural number type to the language $L(Sign)$, then we can internally show that "*Set *has a natural number object".*

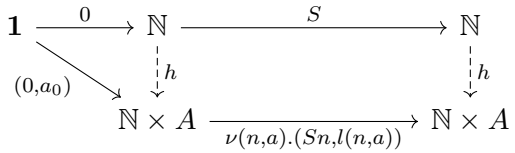
Proof. We have the set $\mathbb{N}$, we have a point $0 : \mathbf{1} \to \mathbb{N}$ and we have a successor function $\nu n.Sn : \mathbb{N} \to \mathbb{N}$ which we denote by S. Assume we are given a set A together with an element $a_0 : \mathbf{1} \to A$ and a function $f : A \to A$. Then we can form a function $\nu n.\,\mathrm{rec}_{\mathbb{N}}(n, a_0, m.x.fx) : \mathbb{N} \to A$ which we denote by h. One can check with help of the uniqueness principle of the natural number type that h is the one and only function which fits into the following diagram.



This shows that Set has a natural number object. □

Proposition C.29. *If we add the mere axiom that "*Set *has a natural number object" to the language $L(Sign)$[23], then we can internally make sense of the natural number type rules.*

Proof. We choose a natural number object $(\mathbb{N}, 0, S)$. We can immediately make sense of the two term introduction rules of the natural number type. For the elimination rule we will start with the weaker version where the target type does not depend on the input. The stronger version can then be derived from the weaker version with the help of strong Σ-types. Assume we are given a set A, a point $a_0 : A$ and a family of points $l(n, x) : A$ indexed by $n : \mathbb{N}, x : A$. We may look at the induced map h in the diagram below.



If $m : \mathbb{N}$ is a point, then we can interpret $\mathrm{rec}_{\mathbb{N}}(m, a_0, n.x.l(x, y)) : A$ as the composite $\pi_2 hm$. One can check that the computation rule and the uniqueness principle are satisfied. □

Corollary C.30. *Assume that $(\mathcal{S}, W)$ is a model which admits weak sum types. Then $CwF(\mathcal{S})$ admits a natural number type if and only if the following is true.*

$$(\mathcal{S}, W) \models \ulcorner \text{Set } \textit{has a natural number object} \urcorner \tag{C.15}$$

[23] We also add the mere axiom that "Set has coproducts" to the language to simplify the proof. This allows us to get around by-hand strictification using the trick spelled out in the first paragraph of the section. This is just avoid work, the proposition will most likely still be true in the absence of sum-types.

D A Small Lexicon between Fibered and Internal Concepts

The aim of this appendix D is to collect translations of some of the internal definitions which we used throughout the thesis. We will also relate them to the corresponding concepts in fibered category theory.

Definition D.1 (internal)**.** A large category C is complete if it has a terminal object, binary products, equalizers and products indexed by small sets.

Proposition D.2. *Assume that* $(\mathcal{S}, W)$ *is a model and that* $C : \mathrm{Cat}$ *is a stack. "*C *is complete" holds internally if and only if the fibration* $\int C$ *is complete in the fibered sense* [8, definition 8.5.5]*.*

Proof. This is an exercise in translating internal statements into external ones. We have already done binary products in detail, and the terminal object and equalizers are similar. The external meaning of "C has a terminal object, binary products and equalizers" is that C, seen as an indexed category, has finite limits in each fiber which are preserved by reindexings up to isomorphisms. So let us now translate the statement "C has small products". It looks as follows when written out.

$$\begin{aligned}\forall A : \mathrm{Set}_0.\forall X : (\textstyle\prod_{x:A} C)_0.\\ \exists P : C_0.\exists \varepsilon : (\textstyle\prod_{x:A} C)_1(\Delta P, X).\ulcorner (P,\varepsilon) \textit{ is terminal among such pairs}\urcorner\end{aligned} \tag{D.1}$$

We remove the first block of quantifiers and start with a stage Γ and an object $\Gamma \vdash X : C_0$ at that stage. The next block of quantifiers says that we may find a cover $u_i : \Delta_i \to \Gamma$ together with terms $\Delta_i \vdash P_i : (Cu_i)_0$ and $\Delta_i, x : Au_i \vdash \varepsilon_i(x) : C_1(X(u_i, x), P_i)$ such that "(P_i, ε_i) is initial among all such pairs" holds internally for all i at the correct stage. The stack condition tells us as usually that we can glue the pairs (P_i, ε_i) to a pair (P, ε) which satisfies the same internal universal property already at stage Γ. The term ε is of type $\Gamma, x : A \vdash \varepsilon(x) : C_1(P, X_x)$. The pair (P, ε) is terminal among all such pairs and the same stays true at lower stages. If we let X vary, then we see that this means exactly that the weakening functor π_A^* has a right adjoint. Since every morphism in the base can be a display map we see that all the u^* have right adjoints. The reindexing stability of the universal property of the pairs (P, ε) translates to the usual Beck-Chevalley condition. We thus recover the fibered definition of small products. □

Corollary D.3. *Let* $\mathcal{S}$ *be a category with finite limits and let* p *be a fibration above* $\mathcal{S}$*. Then* p *is complete in the fibered sense* [8, definition 8.5.5] *if and only if "*$\mathrm{Sp}(p)$ *is complete" holds in the model* $(\mathcal{S}, codis)$*.*

Proof. The only covers in the codiscrete topology are isomorphisms. This means that $\mathrm{Sp}(p)$ is automatically a stack and proposition D.2 applies. □

Proposition D.4. *Assume that* $(\mathcal{S}, W)$ *is a model and that the indexed category* $C : \mathrm{Cat}$ *is at least a prestack. Then* $\int C$ *is locally small in the fibered sense* [43, definition 10.1] *if and only if "*C *is locally small" holds internally.*

Proof. We translate the internal statement "C is locally small" step by step via the forcing semantics. Here is the formal version of "C is locally small".

$$\forall x, y : C_0.\exists H : \mathrm{Set}_0.\exists f : (\textstyle\prod_{h:H} C)_1((x)_{h:H},(y)_{h:H}).\\ \forall g : C_1(x,y).\exists! h : H.f_h = g \qquad \text{(D.2)}$$

Let us remove the first block of quantifiers. We start with an arbitrary stage Γ and two objects $\Gamma \vdash x, y : C_0$ in the corresponding fiber. The next block of quantifiers tells us that we need to restrict to a cover, but since the data that we get on that cover is unique up to unique isomorphisms and Set is a stack we can skip over that step and stay at the current stage. We find terms $\Gamma \vdash H : \mathrm{Set}_0$ and $\Gamma.H \vdash f : C_1(\pi_H^* x, \pi_H^* y)$ such that the following holds internally.

$$\Gamma \Vdash \forall g : C_1(x,y).\exists! h : H.f_h = g \qquad \text{(D.3)}$$

The translation of (D.3) is that for each $g : x \to y$ in the fiber over Γ we may find a section h of π_H such that $h^* f = g$, and the same still works at all lower stages. Or put differently: For all $u : \Delta \to \Gamma$ and for all $g : u^*x \to u^*y$ there is a unique morphism $h : \mathrm{id}_\Delta \to \pi_{u^*H}$ in the slice $\mathcal{S}/\Delta$ such that $h^* u^* f = g$. A morphism $h : \mathrm{id}_\Delta \to \pi_{u^*H}$ in the slice category $\mathcal{S}/\Delta$ is the same thing as a map $h : u \to \pi_H$ in $\mathcal{S}/\Gamma$, and we hence see that π_H represents the following functor.

$$(\mathcal{S}/\Gamma)^{op} \to \mathrm{Set},\ u \mapsto C_{\partial_0 u}(u^*x, u^*y) \qquad \text{(D.4)}$$

But this is exactly the definition of a locally small fibration according to [19, Definition 9.5.1]. □

Proposition D.5. *Let $(\mathcal{S}, W)$ be any model and assume that $C : \mathrm{Cat}$ is at least a semicoflexible prestack. If the fibration $\int C$ associated to C is locally small in the fibered sense, then we can internally equip C with an actual choice of small hom-sets (see remark 5.2).*

Proof. We may without loss of generality assume that C is of the form $\mathrm{Sp}\, D$ for some locally small Grothendieck fibration D, since C is actually equivalent (internally and externally) to its own splitting when C is semicoflexible. Unfortunately this time we can not just point to the semicoflexibility of Set to strictify everything, because we not only need to produce a strict functor $\mathrm{Hom}_C : C^{op} \times C \to \mathrm{Set}$, we also need to prove that the terms of $\mathrm{Set}_1(\mathbf{1}, \mathrm{Hom}_C(x,y))$ correspond bijectively to those of $C_1(x,y)$ in a way which is strictly compatible with reindexings. We choose for each stage Γ and for each $x, y \in D(\Gamma)$ terminal spans as in the diagram shown below.

$$x \xleftarrow{cart.} \underline{h}(x,y) \longrightarrow y$$

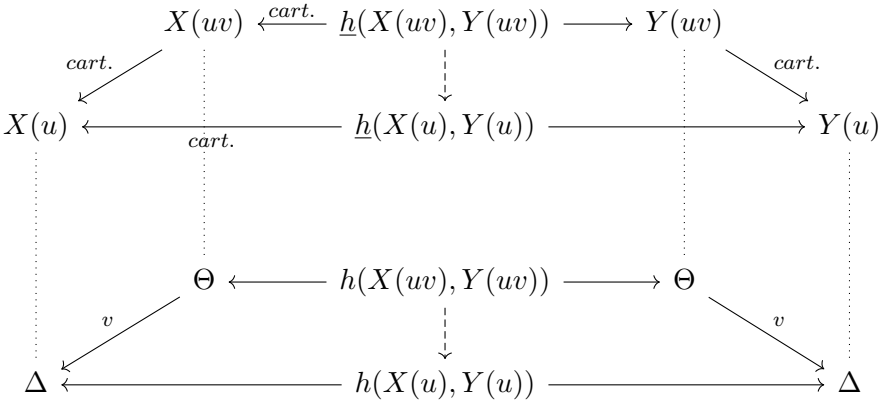
$$\Gamma \longleftarrow h(x,y) \longrightarrow \Gamma$$

This is possible by the definition of a locally small fibration. Now let us assume that we have two objects $\Gamma \vdash X, Y : C$ at stage Γ. Then X and Y are by definition of $C = \mathrm{Sp}(D)$ two fibered functors $X, Y : y\Gamma \to D$ in the 2-category $\mathrm{Fib}_{\mathcal{S}}$. We need to

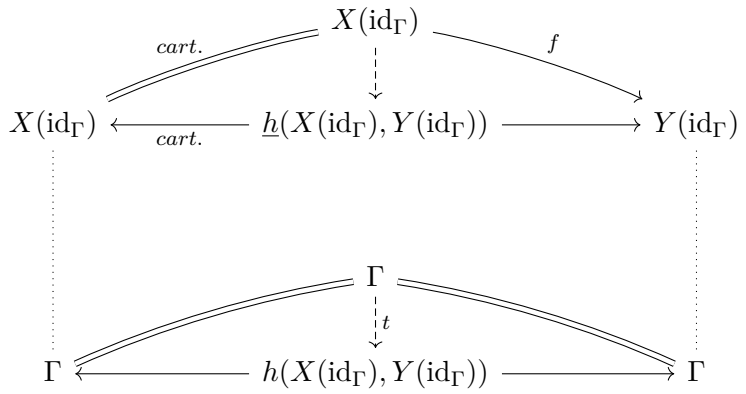
define a fibered functor $\mathrm{Hom}_C(X,Y) : y\Gamma \to \partial_1$ into the self-indexing of $\mathcal{S}$. For each arrow $u : \Delta \to \Gamma$ we let $\mathrm{Hom}_C(X,Y)(u)$ be the chosen map $h(X(u),Y(u)) \to \Delta$ in $\mathcal{S}/\Delta$. This makes sense since $X(u)$ and $Y(u)$ are both objects in D which lie above Δ. Given $v : \Theta \to \Delta$, we also need to specify the action of $\mathrm{Hom}_C(X,Y)$ on the cartesian morphism $uv \to u$ in $y\Gamma$ which indicates that uv is the composite of u and v. We need to construct a map $h(X(uv),Y(uv)) \to h(X(u),Y(u))$ in $\mathcal{S}$ such that the diagram below is a pullback diagram.

$$\begin{array}{ccc} h(X(uv),Y(uv)) & \longrightarrow & h(X(u),Y(u)) \\ \downarrow & & \downarrow \\ \Theta & \xrightarrow{\quad v \quad} & \Delta \end{array}$$

We construct this map by using the universal property of the spans $x \leftarrow \underline{h}(x,y) \to y$. There is a unique map in D which fits into the dashed place in the diagram below.



Next one has to check that those arrows indeed give us pullback diagrams in $\mathcal{S}$. One also has to check that each $\mathrm{Hom}_C(X,Y)$ is a functor and that the formation of the $\mathrm{Hom}_C(X,Y)$ commutes strictly with reindexings. We will skip over all those verifications. Let us instead describe the bijection between morphisms $\Gamma \vdash f : C(X,Y)$ and terms $\Gamma \vdash t : \mathrm{Hom}_C(X,Y)$. We remember that each such t is a section of the display map of $\mathrm{Hom}_C(X,Y)$. Given a morphism f, we let the associated term t be the map defined by the diagram below.



Conversely, given a section t of the display map, we let can take the cartesian lift of t which is up to a canonical isomorphism a morphism $X(\mathrm{id}_\Gamma) \to \underline{h}(X(\mathrm{id}_\Gamma), Y(\mathrm{id}_\Gamma))$ in D. Composing that map with the morphism into $Y(\mathrm{id}_\Gamma)$ gives us back f. We omit the tedious verification that the bijection between terms and morphisms is strictly compatible with reindexings. □

Corollary D.6. *Let $\mathcal{S}$ be a lex category. A fibration C above $\mathcal{S}$ is locally small in the fibered sense if and only if "$\mathrm{Sp}\, C$ is locally small" holds in the model $(\mathcal{S}, codis)$. If that is the case, then $\mathrm{Sp}\, C$ can even be equipped with a choice of internal hom-sets (see remark 5.2).*

Proof. Apply proposition D.4 and D.5 to the semicoflexible stack $\mathrm{Sp}(D)$. □

Proposition D.7. *Let $(\mathcal{S}, W)$ be a model and assume that $C : \mathrm{Cat}$ is a prestack. Let $(g)_{x:A} : \mathrm{Fam}(C)$ be a family of objects in C indexed by some internal set $A : \mathrm{Set}$. Then "$(g_x)_{x:A}$ is a generating family" holds internally if and only if $(\ ).A \vdash g : C_0$ is a generating family in the fibered sense* [43, definition 10.2].

Proof. The internal statement that $(g_x)_{x:A}$ is a generating family of objects looks as follows in its formal version.

$$\forall x, y : C_0.\forall f, h : C_1(x, y).((\forall a : A.\forall t : C_1(g_a, x).ft = ht) \to f = h) \tag{D.5}$$

When can remove the first block of quantifiers, the implication and the second block of quantifiers at once. The translation is: For every parallel pair of arrows $f, h : x \rightrightarrows y$ in C at some stage Γ, if for every $u : \Delta \to \Gamma$, $(!_\Delta, a) : \Delta \to (\ ).A$ and $\Delta \vdash t : C_1(g(!_\Delta, a), u^*x)$ it holds that $u^*f \circ t = u^*h \circ t$, then f and h are equal. But this is exactly the definition of a small generating family in [43, Definition 10.2]. □

Corollary D.8. *If C is a Grothendieck fibration over a lex category $\mathcal{S}$ which has a small generating family in the fibered sense, then it is internally true in the model $(\mathcal{S}, codis)$ that "$\mathrm{Sp}\, C$ merely has a small generating family".*

Proof. Apply proposition D.7 to the indexed category $\mathrm{Sp}(C)$ in the model $(\mathcal{S}, codis)$. □

Proposition D.9. *Let $(\mathcal{S}, W)$ be a model and assume that $C : \mathrm{Cat}$ is a stack. Assume additionally that vertical monomorphisms are preserved by the reindexing operations of C. Then "C is well-powered" holds in the model if and only if C is well-powered in the fibered sense* [43, definition 11.1].

Proof. The statement that "C is well-powered looks as follows when we write it down formally.

$$\begin{aligned}&\forall x : C_0.\exists S : \mathrm{Set}_0.\exists y : (\textstyle\prod_{s:S} C)_0.\exists m : (\textstyle\prod_{s:S})_1(y, \Delta x).\\ &\qquad \ulcorner \textit{every mono } n : z \to x \textit{ is isomorph to a unique } m_s : y_s \to x \urcorner\end{aligned} \tag{D.6}$$

We start with a stage Γ and an object $\Gamma \vdash x : C_0$. The forcing conditions for the existential quantifier tell us to switch to a cover, but since Set and C are stacks

and the data is unique up to unique isomorphisms we can avoid this. Hence we have some $\Gamma.S \vdash y : C_0$ together with a term $\Gamma.S \vdash m : C_1(y, \pi_S^* x)$ such that the following is true.

$$\Gamma \Vdash \forall n : C_1(z, x).(\ulcorner n \text{ is monic} \urcorner \to \exists! s : S.\ulcorner n \cong m_s \text{ as subobjects of } x \urcorner) \quad \text{(D.7)}$$

The translation of an internal statement "n is monic" is just that n and all of its reindexings are monic. Since reindexings preserve monomorphisms by assumption we see that the translation of "n is monic" is just that n is monic in its fiber. The property that $m : y \to \pi_S^* x$ satisfies is thus the following: For all arrows $u : \Delta \to \Gamma$ in the base and for every monomorphism $n : z \to u^* x$ in the fiber over Δ there is a unique section $\Delta \vdash s : u^* S$ such that $n : z \to u^* x$ and $m(u, s) : y(u, s) \to u^* x$ are isomorphic in C_Δ / x. This is a reformulation of the fibered definition that C is well-powered [43, definition 11.1]. □

Proposition D.10. *Assume that* $(\mathcal{S}, W)$ *is a model and that* $C : \mathrm{Cat}$ *is a semicoflexible stack. When the fibration* $\int C$ *associated to* C *is well-powered in the fibered sense, then we can internally equip* C *with an actual choice of subobject-sets* $\mathrm{Sub}_C(x) : \mathrm{Set}$.

Proof. The proof should look very similar to the proof of proposition D.5 and we thus skip over it. Since the sets $\mathrm{Sub}_C(x)$ with their universal families of monomorphism can not be characterised through an adjunction at the global stage in an obvious way, one has to solve the coherence problem by hand. □

References

[1] Mike Shulman (https://mathoverflow.net/users/49/mike-shulman). *Weighted (co)limits as adjunctions*. MathOverflow. https://mathoverflow.net/q/324986 (version: 2019-03-09). eprint: `https://mathoverflow.net/q/324986`. URL: `https://mathoverflow.net/q/324986`.

[2] Steven Awodey and Andrej Bauer. "Propositions as [types]". In: *Journal of logic and computation* 14.4 (2004), pp. 447–471.

[3] John C Baez and Michael Shulman. "Lectures on n-categories and cohomology". In: *Towards higher categories* (2010), pp. 1–68.

[4] Jean Bénabou. "Fibered categories and the foundations of naive category theory". In: *The Journal of Symbolic Logic* 50.1 (1985), pp. 10–37.

[5] GJ Bird et al. "Flexible limits for 2-categories". In: *Journal of Pure and Applied Algebra* 61.1 (1989), pp. 1–27.

[6] Robert Blackwell, Gregory M Kelly, and A John Power. "Two-dimensional monad theory". In: *Journal of pure and applied algebra* 59.1 (1989), pp. 1–41.

[7] Ingo Blechschmidt. "Exploring mathematical objects from custom-tailored mathematical universes". In: *Objects, Structures, and Logics: FilMat Studies in the Philosophy of Mathematics*. Springer, 2021, pp. 63–95.

[8] Francis Borceux. *Handbook of Categorical Algebra: Volume 2, Categories and Structures*. Vol. 2. Cambridge University Press, 1994.

[9] John Bourke and Richard Garner. "On semiflexible, flexible and pie algebras". In: *Journal of Pure and Applied Algebra* 217.2 (2013), pp. 293–321.

[10] Marta Bunge and Robert Paré. "Stacks and equivalence of indexed categories". In: *Cahiers de topologie et geometrie differentielle* 20.4 (1979), pp. 373–399.

[11] Aurelio Carboni, Stephen Lack, and Robert FC Walters. "Introduction to extensive and distributive categories". In: *Journal of Pure and Applied Algebra* 84.2 (1993), pp. 145–158.

[12] ME Descotte, EJ Dubuc, and M Szyld. "Model bicategories and their homotopy bicategories". In: *arXiv preprint arXiv:1805.07749* (2018).

[13] Peter Dybjer. "Internal type theory". In: *International Workshop on Types for Proofs and Programs*. Springer. 1995, pp. 120–134.

[14] Nicola Gambino. "Homotopy limits for 2-categories". In: *Mathematical Proceedings of the Cambridge Philosophical Society*. Vol. 145. 1. Cambridge University Press. 2008, pp. 43–63.

[15] Kengo Hirata. "Notes on Lax Ends". In: *arXiv preprint arXiv:2210.01522* (2022).

[16] Philip S Hirschhorn. *Model categories and their localizations*. 99. American Mathematical Soc., 2003.

[17] Martin Hofmann. "On the interpretation of type theory in locally cartesian closed categories". In: *International Workshop on Computer Science Logic*. Springer. 1994, pp. 427–441.

[18] Martin Hofmann and Martin Hofmann. "Syntax and semantics of dependent types". In: *Extensional Constructs in Intensional Type Theory* (1997), pp. 13–54.

[19] Bart Jacobs. *Categorical logic and type theory*. Elsevier, 1999.

[20] Niles Johnson and Donald Yau. "2-dimensional categories". In: *arXiv preprint arXiv:2002.06055* (2020).

[21] Peter T Johnstone. *Sketches of an Elephant: A Topos Theory Compendium: Volume 2*. Vol. 2. Oxford University Press, 2002.

[22] Peter T Johnstone et al. "Abstract families and the adjoint functor theorems". In: *Indexed categories and their applications*. Springer. 1978, pp. 1–125.

[23] Gregory Maxwell Kelly. "Elementary observations on 2-categorical limits". In: *Bulletin of the Australian Mathematical Society* 39.2 (1989), pp. 301–317.

[24] Stephen Lack. "A 2-categories companion". In: *Towards higher categories*. Springer, 2009, pp. 105–191.

[25] Stephen Lack. "Codescent objects and coherence". In: *Journal of Pure and Applied Algebra* 175.1-3 (2002), pp. 223–241.

[26] Stephen Lack. "Homotopy-theoretic aspects of 2-monads". In: *arXiv preprint math/0607646* (2006).

[27] F William Lawvere and Colin McLarty. "An elementary theory of the category of sets (long version) with commentary". In: *Reprints in Theory and Applications of Categories* 11 (2005), pp. 1–35.

[28] Peter LeFanu Lumsdaine and Michael A Warren. "The local universes model: an overlooked coherence construction for dependent type theories". In: *ACM Transactions on Computational Logic (TOCL)* 16.3 (2015), pp. 1–31.

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

[30] Maria Emilia Maietti. “Modular correspondence between dependent type theories and categories including pretopoi and topoi”. In: *Mathematical Structures in Computer Science* 15.6 (2005), pp. 1089–1149.
[31] Michael Makkai. “First order logic with dependent sorts, with applications to category theory”. In: *Preprint: http://www. math. mcgill. ca/makkai* (1995).
[32] Per Martin-Löf and Giovanni Sambin. *Intuitionistic type theory*. Vol. 9. Bibliopolis Naples, 1984.
[33] Lyne Moser. “Injective and projective model structures on enriched diagram categories”. In: *arXiv preprint arXiv:1710.11388* (2017).
[34] nLab authors. *Grothendieck construction*. `https://ncatlab.org/nlab/show/Grothendieck+construction`. Revision 94. Oct. 2023.
[35] nLab authors. *Thomason model structure*. `https://ncatlab.org/nlab/show/Thomason+model+structure`. Revision 35. July 2023.
[36] Bengt Nordström, Kent Petersson, and Jan M Smith. *Programming in Martin-Löf's type theory*. Vol. 200. Oxford University Press Oxford, 1990.
[37] Fernando Lucatelli Nunes and Lurdes Sousa. “On lax epimorphisms and the associated factorization”. In: *Journal of Pure and Applied Algebra* 226.12 (2022), p. 107126.
[38] Erik Palmgren. “Categories with families, folds and logic enriched type theory”. In: (2016).
[39] John Power and Edmund Robinson. “A characterization of pie limits”. In: *Mathematical Proceedings of the Cambridge Philosophical Society* 110 (July 1991), pp. 33–47. DOI: `10.1017/S0305004100070092`.
[40] Emily Riehl. *Category theory in context*. Courier Dover Publications, 2017.
[41] M Shulman. “Large categories and quantifiers in topos theory, 2021 slides”. In: *See http://home. sandiego. edu/˜ shulman/papers/cambridge-stacksem. pdf* ().
[42] Michael A Shulman. “Stack semantics and the comparison of material and structural set theories”. In: *arXiv preprint arXiv:1004.3802* (2010).
[43] Thomas Streicher. “Fibered categories à la Jean Benabou”. In: *arXiv preprint arXiv:1801.02927* (2018).

Statement of authorship

I hereby declare that the thesis I am submitting is my own work and that information which has been directly or indirectly taken from other sources has been noted as such.

Berlin, January 4, 2025 .....