# Type Theory à la Mode

Dependent Type Theory with Multiple Modes and Modalities

Daniel Gratzer
Aarhus University

Alex Kavvos*
Aarhus University

Andreas Nuyts†
imec-DistriNet
KU Leuven

Lars Birkedal
Aarhus University

Wednesday 29th April, 2020

*Supported in part by a research grant (12386, Guarded Homotopy Type Theory) from Villum Fonden.
†Holds a PhD Fellowship from the Research Foundation - Flanders (FWO).

# Contents

**Abstract**

We introduce MTT, a dependent type theory which supports multiple modalities. MTT is parameterized by a mode theory which specifies a collection of modes, modalities, and transformations between them. We show that different choices of mode theory allow us to use the same type theory to compute and reason in many modal situations, including guarded recursion, axiomatic cohesion, and parametric quantification. We reproduce examples from prior work in guarded recursion and axiomatic cohesion — demonstrating that MTT constitutes a simple and usable syntax whose instantiations intuitively correspond to previous handcrafted modal type theories. In some cases, instantiating MTT to a particular situation unearths a previously unknown type theory that improves upon prior systems. Finally, we investigate the metatheory of MTT. We prove the consistency of MTT and establish canonicity through an extension of recent type-theoretic gluing techniques. These results hold irrespective of the choice of mode theory, and thus apply to a wide variety of modal situations.

# 1 *The Syntax and Semantics of MTT*

Nothing is more arbitrary than a modal logic: "I am done with this logic, may I have another one ?" seems to be the motto of modal logicians.

Jean-Yves Girard
*The Blind Spot*

The objective of this work is to define and study a *multimode* and *multimodal* dependent type theory.

By 'multimode' we mean that each type and term of this type theory can be thought of as being in a particular *mode*. The modes of this type theory may share some common structure—for example, we may be able to form dependent sums in all modes—but some may have their own unique features. In semantic terms, *different modes correspond to different categories*.

Once we have established multiple modes, our type theory will allow us to relate them. In other words, the type theory will feature emergent *modal* behaviour. The various relations between different modes will be known as *modalities*. As we have already taken care to admit multiple modes in our theory, nothing will stop us from admitting multiple modalities between any two such modes. In that sense, this type theory will be not just multimode, but also *multimodal*. For the purposes of this work, we will limit ourselves to simple relations between modes, namely relations of a functional nature. In semantic terms, *our modalities will be functors* with specific properties.

In the past few years there has been considerable interest in and demand for such a type theory. A special workshop on Geometry in Modal Homotopy Type Theory took place in Pittsburgh during March 2019, and the HoTTEST seminar by Licata [Lic19] nicely surveys a number of candidate applications that would be enabled by the existence of such a type theory. There is already quite a bit of literature on extend type theory with modalities. For example, modalities have been used to express guarded recursive definitions [Bir+12; Biz+16; BGM17], to internalize parametricity arguments and quantification [NVD17; ND18], to capture proof irrelevance [Pfe01; AS12; ND18], and to define global operations on types and terms (cf. slice-wise) [Lic+18]. There has also been a concerted effort to construct a dependent type theory corresponding to Lawvere's *axiomatic cohesion* [Law07], which has many interesting applications [Sch13; SS12; Shu18; Gro+17; Kav19].

Despite this recent flurry of developments, a unifying account of modal dependent type theory has yet to emerge. Faced with a new modal situation, the type theorist must handweave a brand new system, and then prove the usual battery of metatheorems. This introduces formidable difficulties on two levels. First, an increasing number of these applications are *multimodal*: they involve multiple interacting modalities, which significantly complicates the design of the appropriate judgmental structure. Second, the technical development of each such system is entirely separate from the others, so it is impossible to share the burden of proof—even between very closely related systems. To take a recent example, there is no easy way to transfer the work done in the 80-page-long normalization proof for $\mathsf{MLTT}_{🔒}$ [GSB19] to a normalization proof for the modal dependent type theory of Birkedal et al. [Bir+20], even though these systems are only marginally different. Put simply, if one wished to prove that type-checking is decidable for the latter, then one would have to start afresh.

We intend to avoid such duplication in the future. Rather than designing a new dependent type theory for some preordained set of modalities, we will introduce a system that is *parametrized* by a *mode theory*, i.e. an algebraic specification of a modal situation. This system, which we call MTT, solves both problems at once. First, by instantiating it with different mode theories we will show that MTT can capture a wide class of situations. Some of these, e.g. the one for guarded recursion, lead to a previously unknown system that improves upon earlier work. Second, the predictable behavior of our rules allows us to prove metatheoretic results about large classes of instantiations of MTT at once. For example, our canonicity theorem applies irrespective of the chosen mode theory. As a result, we only need to prove such results *once*. Returning to the previous example, careful choices of mode theory yield two systems that closely resemble the calculi of Birkedal et al. [Bir+20] and MLTT$_{🔒}$ [GSB19] respectively, so that our proof of canonicity applies to both.

The work that is closest in spirit to ours is the dependently-typed extension of the Licata-Shulman-Riley (LSR) framework [LSR17]. As of April 2020, this work remains unpublished. Our approach is not as complex as that of *op. cit.* as we have consciously chosen to make our type theory cartesian (i.e. admitting weakening and contraction). This puts us at odds with the goals of the LSR framework, which is meant to encompass linear dependent type theories, with a view to many interesting applications in synthetic homotopy theory. Our decision to remain cartesian puts these examples out of reach. However, we remain practically-minded: we will see in §2 that most of our motivating examples—which largely arise from applications in programming languages—have an intuitive and elegant formulation in our setting.

## 1.1 Syntax of MTT

We now introduce the syntax of our multimode type theory, MTT, which is a type theory in the style of Martin-Löf. In the interest of conveying the intuitions without becoming overly formal, our description will be in terms of an informal, variable-based syntax, as is common for type theories in the style of Martin-Löf. In Section 1.2 we present the theory formally using *algebraic syntax*, which greatly simplifies the proof of a number of metatheoretic and semantic results.

### 1.1.1 Informal Syntax

The salient difference between MTT and ordinary Martin-Löf type theory is that the judgments of MTT (contexts, types, and terms) are parameterised by a *mode*. In semantic terms, each mode represents a distinct category. In order to capture the allowed modes we define our judgments parametrically in a small 2-category $\mathcal{M}$. This category is referred to as the *mode theory* [Ree09; LS16; LSR17], and it axiomatizes the modalities in scope as well as their interactions. We refer to the objects of $\mathcal{M}$ as modes, and to the morphisms between them as *modalities*. In §2 we show how we may obtain previously studied modal idioms through a careful choice of mode theory.

Returning to judgments, we will for example write

$$\Gamma \vdash M : A \mathrel{@} m$$

for a term $M$ of type $A$, in context $\Gamma$, at mode $m \in \mathcal{M}$. Many of the usual rules (e.g. those for $\Sigma$-types) will be parametric in $m$. Others, such as the rule for $\Pi$-types, will interact in a more intricate manner with the mode. Finally, the modal rules of the type theory will exactly describe the various ways of moving between the different modes.

The ways of moving between different modes are often called *modalities*. In our setting, modalities are specified by the morphisms of the 2-category $\mathcal{M}$, for which we will write $\mu : \mathrm{Hom}_{\mathcal{M}}(m, n)$ or $\mu : m \to n$. Each such morphism introduces a *contravariant functorial action* on contexts. That is: we are allowed to push a context backwards along a modality, which results in an 'image' of it in another mode. We notate this functorial action by a *lock*, which is introduced by the rule

$$\frac{\Gamma\ \mathsf{ctx} \mathrel{@} n \qquad \mu : \mathrm{Hom}_{\mathcal{M}}(m, n)}{\Gamma, \text{🔒}_{\mu}\ \mathsf{ctx} \mathrel{@} m}$$

The functoriality of this action will be enforced by equalities such as $\Gamma, \text{🔒}_{\nu}, \text{🔒}_{\mu} = \Gamma, \text{🔒}_{\nu \circ \mu}\ \mathsf{ctx} \mathrel{@} m$ whenever $\mu$ and $\nu$ are composable.

The first serious departure from the usual Martin-Löf style occurs in the context extension rule, which encapsulates the idea that *each assumption in the context comes under a modality*. We signify that by writing $x : (\mu \mid A)$ in place of the usual $x : A$. Thus, before extending a context $\Gamma\ \mathsf{ctx} \mathrel{@} n$ we need to make sure that the type $A$ by which it is to be extended exists under some modality $\mu : \mathrm{Hom}_{\mathcal{M}}(m, n)$. That is: $A$ must be a type in mode $m$. Yet, $A$ must also be a type in context $\Gamma$, which is in the wrong mode. To resolve this issue we use the image of $\Gamma$ under the lock, which lives in the correct mode. These considerations lead to the context extension rule

$$\frac{\Gamma\ \mathsf{ctx} \mathrel{@} n \qquad \mu : \mathrm{Hom}_{\mathcal{M}}(m, n) \qquad \Gamma, \text{🔒}_{\mu} \vdash A\ \mathsf{type} \mathrel{@} m}{\Gamma, x : (\mu \mid A)\ \mathsf{ctx} \mathrel{@} n}$$

This change in the structure of contexts necessitates a change in the *variable rule*. The variable rule is what allows us to use the assumptions found in a context. Recalling that our assumptions are now all modal, it is evident that the variable rule is the central device that generates modal behaviour in our system. Its usual form stipulates that, given a context $\Gamma_0, x : A, \Gamma_1$ ctx, we may 'project' the assumption $x : A$ to obtain the term $\Gamma_0, x : A, \Gamma_1 \vdash x : A$. This rule is certainly not valid in our system:

given $\Gamma_0, x : (\mu \mid A), \Gamma_1 \ \mathsf{ctx} @ n$, the assumption $x : (\mu \mid A)$ is available under the modality $\mu$, i.e. in the wrong mode, or—equivalently—in a different category.

In order to account for this modal structure, therefore, we give a more refined definition of the variable rule. For the sake of simplicity, suppose that $\Gamma_1$ is empty. If $\mu : \mathrm{Hom}_{\mathcal{M}}(m, n)$, we may construct the context $\Gamma_0, x : (\mu \mid A), \text{🔒}_{\mu} \ \mathsf{ctx} @ m$. This context is at mode $m$, and it is thus not unreasonable to ask that our variable rule should at the very least allow the inference

$$\frac{\Gamma_0, x : (\mu \mid A), \text{🔒}_{\mu} \ \mathsf{ctx} @ m}{\Gamma_0, x : (\mu \mid A), \text{🔒}_{\mu} \vdash x : A @ m} \tag{1.1}$$

Those familiar with type theories with a comonadic modality might notice that this has the flavour of a *counit* for the comonad generated by a 'left adjoint' $-, \text{🔒}_{\mu}$ and a 'right adjoint' $(\mu \mid -)$ (but not quite, as those two act on distinct sorts, viz. contexts and types respectively).

However, this is not the full extent of the expressive power we have at our disposal. Recall that the 2-category $\mathcal{M}$ is also equipped with 2-cells, i.e. transformations $\alpha : \mu \Rightarrow \nu$ between modalities. If each modality $\mu : \mathrm{Hom}_{\mathcal{M}}(m, n)$ introduces a (contravariant) functorial action $-, \text{🔒}_{\mu}$ from contexts at $n$ to contexts at $m$, then surely each 2-cell $\alpha : \mu \Rightarrow \nu$ should (contravariantly) generate a natural transformation from the action $-, \text{🔒}_{\nu}$ to the action $-, \text{🔒}_{\mu}$.[1] Thus, as long as a natural transformation $\alpha : \mu \Rightarrow \nu$ allows us to adjust a lock to be $-, \text{🔒}_{\nu}$ we should still be able to extract a variable that is available under $\mu$. To wit, our variable rule should allow the inference

$$\frac{\mu, \nu : \mathrm{Hom}_{\mathcal{M}}(m, n) \qquad \alpha : \mu \Rightarrow \nu}{\Gamma_0, x : (\mu \mid A), \text{🔒}_{\nu} \vdash x^{\alpha} : A^{\alpha} @ m} \tag{1.2}$$

Notice that $\alpha$ has now appeared as a superscript of both the variable $x^{\alpha}$ and the type $A^{\alpha}$. In the first case, $\alpha$ becomes part of the syntax: each variable should come annotated with a rule that indicates which natural transformation allowed us to extricate it from the context. The second case is slightly more complicated. Recall that $\Gamma_0, x : (\mu \mid A), \text{🔒}_{\mu} \ \mathsf{ctx} @ m$ presupposes that $\Gamma_0, \text{🔒}_{\mu} \vdash A \ \mathsf{type} @ m$, and hence that—modulo weakening under the lock—$A$ is indeed in the right context in (1.1). However, in (1.2) we must use the 2-cell $\alpha$ to somehow "act on $A$" in order to bring it to the correct context $\Gamma_0, \text{🔒}_{\nu}$, which we can then silently weaken to $\Gamma_0, x : (\mu \mid A), \text{🔒}_{\nu}$. As types depend on terms, we must define the action of $\alpha$ on a term as well. The good news is that—much like substitution—this metatheoretic action is admissible.

We have thus identified three principles that should be encapsulated by the variable rule:

1. The ability to use a variable depends on what appears *to the right of it* in the context, i.e. it depends on the presence of a suitable lock.

2. The 2-cells express in which way we may *strengthen* locks in order to make them exactly match the pattern $x : (\mu \mid A), \text{🔒}_{\mu}$, in which case we can use the variable.

3. Some additional context weakening needs to be built into the rule.

These guiding principles bring us to the final version. We first define a function that gathers the modalities from all the locks that appear in a telescope. This function is defined by the following clauses:

$$\begin{aligned} \mathsf{locks}(\cdot) &\triangleq 1 \\ \mathsf{locks}(\Gamma, x : (\mu \mid A)) &\triangleq \mathsf{locks}(\Gamma) \\ \mathsf{locks}(\Gamma, \text{🔒}_{\mu}) &\triangleq \mathsf{locks}(\Gamma) \circ \mu \end{aligned}$$

[1] The reason for this double ('coop') contravariance will become evident when we discuss the categorical semantics of this type theory in Section 1.4. For now, we just point out that it is in line with thinking of the lock as a left adjoint to the actual modality.

The variable rule then is

$$\frac{\Gamma_0, x : (\mu \mid A), \Gamma_1 \ \mathsf{ctx} \ @\, m \qquad \alpha : \mu \Rightarrow \mathsf{locks}(\Gamma_1)}{\Gamma_0, x : (\mu \mid A), \Gamma_1 \vdash x^{\alpha} : A^{\alpha} \ @\, m} \tag{1.3}$$

In short: we gather the modalities under which $\Gamma_1$ locks the context that precedes it, and we look for a 2-cell that witnesses the fact that $\mu$ is strong enough to slide past these locks.

In many examples it will be the case that $\mu = \mathsf{locks}(\Gamma_1)$, and we wish to pick $\alpha = 1$ in order to access the variable. In these cases we will elide the subscript entirely and simply write $x$. In particular, this means that when accessing a variable modified by 1 and behind no locks we can simply write $x$.

**Remark 1.1.1** (The operation $(-)^{\alpha}$)**.** Even though our formal study will be of the algebraic syntax of Section 1.2, we provide a brief description of how to define, for $\alpha : \mu \Rightarrow \nu$, the admissible operation

$$\Gamma_0, \text{🔒}_{\mu} \vdash A \ \mathsf{type} \ @\, m \ \longmapsto\ \Gamma_0, \text{🔒}_{\nu} \vdash A^{\alpha} \ \mathsf{type} \ @\, m$$

As types depend on terms, we also need a similar operation which maps a term $\Gamma_0, \text{🔒}_{\mu} \vdash M : A \ @\, m$ to a term $\Gamma_0, \text{🔒}_{\nu} \vdash M^{\alpha} : A^{\alpha} \ @\, m$. Intuitively, the operation $(-)^{\alpha}$ must whisker appropriately the 2-cell $\beta$ occurring as part of each variable occurence $x^{\beta}$ in $M$. However, the 2-cell by which it must be whiskered depends on the structure of the context $\Gamma_0$, and becomes more complicated as we recur down the typing derivation for $M$.

The clearest way to account for this is to define a finite map $\sigma(\Gamma_0, \alpha)$ from variables of $\Gamma_0$ to 2-cells by induction on $\Gamma_0$. This map keeps a record of which variable we must whisker by which 2-cell. It is given by the clauses

$$\begin{aligned} \sigma(\cdot, \alpha) &= \emptyset \\ \sigma(\Gamma, x : (\mu \mid A), \alpha) &= \sigma(\Gamma, \alpha) \cup \{x \mapsto \alpha\} \\ \sigma(\Gamma, \text{🔒}_{\mu}, \alpha) &= \sigma(\Gamma, 1_{\mu} \star \alpha) \end{aligned}$$

As we recur past locks in $\Gamma_0$ the 2-cell by which we whisker is adjusted, in order to ensure that it has the right boundary. We then define[2] this admissible operation as follows:

$$A^{\alpha} \triangleq A^{\sigma(\Gamma_0, \alpha)} \qquad\qquad M^{\alpha} \triangleq M^{\sigma(\Gamma_0, \alpha)}$$

It remains to define the action $A^{\sigma}$ and $M^{\sigma}$ of such a finite map $\sigma$ on a term $A$ and a type $M$ respectively. For now, we define this just on variables:

$$(x^{\beta})^{\sigma} \triangleq x^{\sigma(x) \circ \beta}$$

We will later extend this to all the type formers we introduce. We will take care so that this extension always satisfies that $\Gamma_0, \text{🔒}_{\mu} \vdash x^{\beta} : A^{\beta}$ implies $\Gamma_0, \text{🔒}_{\nu} \vdash x^{\sigma(\Gamma_0,\alpha)(x) \circ \beta} : A^{\sigma(\Gamma_0,\alpha)(x) \circ \beta}$.

This operation suffices to yield the full one of (1.3): we can perform a series of silent context conversion steps where we decompose the lock in $\Gamma_0, \text{🔒}_{\mathsf{locks}(\Gamma_1)}$ into the locks from which the composition $\mathsf{locks}(\Gamma_1)$ originated, followed by a number of silent weakenings under the appropriate locks. $\lhd$

Our type theory will also have $\Pi$-types. These will follow the structure of our context assumptions, which have been altered to include a modality. The formation rule reflects this fact:

$$\frac{\mu : \mathrm{Hom}_{\mathcal{M}}(n, m) \qquad \Gamma, \text{🔒}_{\mu} \vdash A \ \mathsf{type} \ @\, n \qquad \Gamma, x : (\mu \mid A) \vdash B \ \mathsf{type} \ @\, m}{\Gamma \vdash (x : (\mu \mid A)) \to B \ \mathsf{type} \ @\, m}$$

[2] Note that if $\rho \circ \mu = \mu$ and $\rho \circ \nu = \nu$, and $\Gamma, \text{🔒}_{\rho}, \text{🔒}_{\mu} \vdash A \ \mathsf{type} \ @\, m$ then $A^{\alpha}$ could be interpreted as either $A^{\sigma(\Gamma,\alpha)}$ or $A^{\sigma((\Gamma,\text{🔒}_{\rho}),\alpha)}$. These are not equivalent in general, and so it is necessary to specify $\Gamma_0$.

This rule encodes the idea that the variable $x : (\mu \mid A)$ in terms of which $B$ is given is abstracted along with the modality $\mu$. Thus our dependent function spaces are modal. Of course, the usual MLTT function space is recovered by taking $\mu = 1$.

The introduction rule is predictable: to introduce a function, we $\lambda$-abstract:

$$\frac{\mu : \mathrm{Hom}_{\mathcal{M}}(n,m) \qquad \Gamma, x : (\mu \mid A) \vdash B \ \mathsf{type} \mathbin{@} m \qquad \Gamma, x : (\mu \mid A) \vdash M : B \mathbin{@} m}{\Gamma \vdash \lambda x.\ M : (\mu \mid A) \to B \mathbin{@} m}$$

More crucially, we may only apply functions to arguments that are available in the right mode. The elimination rule is:

$$\frac{\begin{array}{c}\mu : \mathrm{Hom}_{\mathcal{M}}(n,m) \\ \Gamma, x : (\mu \mid A) \vdash B \ \mathsf{type} \mathbin{@} m \qquad \Gamma \vdash M : (x : (\mu \mid A)) \to B \mathbin{@} m \qquad \Gamma, \text{🔒}_{\mu} \vdash N : A \mathbin{@} n\end{array}}{\Gamma \vdash M(N) : B[N/x] \mathbin{@} m}$$

**Remark 1.1.2** (The operation $(-)^{\alpha}$ on $\Pi$-types)**.** We define

$$((x : (\mu \mid A)) \to B)^{\sigma} = (x : (\mu \mid A^{\sigma'})) \to B^{\sigma \cup \{x \mapsto 1\}} \quad \text{where } \sigma'(x) = \sigma(x) \star 1_{\mu} \qquad \triangleleft$$

Finally, we reify the modalities as operators on types. The introduction rule turns a lock $-, \text{🔒}_{\mu}$ into a unary operator on types, which we write as $\langle \mu \mid - \rangle$:

$$\frac{\mu : \mathrm{Hom}_{\mathcal{M}}(n,m) \qquad \Gamma, \text{🔒}_{\mu} \vdash M : A \mathbin{@} n}{\Gamma \vdash \mathsf{mod}_{\mu}(M) : \langle \mu \mid A \rangle \mathbin{@} m}$$

It is thus evident that $\langle \mu \mid - \rangle$ demonstrates behaviour akin to that of a 'right adjoint' to $-, \text{🔒}_{\mu}$.

The elimination rule is somewhat more complicated. The fundamental intuition is the following: if we have a term $M : \langle \nu \mid A \rangle$, we should be able to substitute it for an assumption of the form $v : (\nu \mid A)$. That is, we should be able to open $\langle \nu \mid A \rangle$ into a variable available under the modality $\nu$. Hence, our elimination rule should have a *positive* flavour, and should moreover admit the inference

$$\frac{\begin{array}{c}\nu : \mathrm{Hom}_{\mathcal{M}}(o,n) \qquad \Gamma \vdash M_0 : \langle \nu \mid A \rangle \mathbin{@} n \\ \Gamma, x : (1 \mid \langle \nu \mid A \rangle) \vdash B \ \mathsf{type} \mathbin{@} n \qquad \Gamma, v : (\nu \mid A) \vdash M_1 : B[\mathsf{mod}_{\nu}(v)/x] \mathbin{@} n\end{array}}{\Gamma \vdash \mathsf{let}\ \mathsf{mod}_{\nu}(v) \leftarrow M_0 \ \mathsf{in}\ M_1 : B[M_0/x] \mathbin{@} n}$$

Notice how the variable $x$ in the motive $B$ is replaced by the modal version of $v$. This rule encapsulates a form of *modal induction*, viz. that every variable $x : (1 \mid \langle \nu \mid A \rangle)$ can be assumed to be of the form $\mathsf{mod}_{\nu}(v)$ for some $v : (\nu \mid A)$.

From this point, we only require a small generalisation to reach the final version of the rule. This generalisation is needed to deal with the case where the variable $x$ in the motive $B$ type is not under the identity modality $1 : \mathrm{Hom}_{\mathcal{M}}(n,n)$, but under a further level of indirection, i.e. a general modality $\mu : \mathrm{Hom}_{\mathcal{M}}(n,m)$. In that case, we may absorb the additional modality by using composition in the 2-category $\mathcal{M}$:

$$\frac{\begin{array}{c}\nu : \mathrm{Hom}_{\mathcal{M}}(o,n) \qquad \mu : \mathrm{Hom}_{\mathcal{M}}(n,m) \qquad \Gamma, \text{🔒}_{\mu} \vdash M_0 : \langle \nu \mid A \rangle \mathbin{@} n \\ \Gamma, x : (\mu \mid \langle \nu \mid A \rangle) \vdash B \ \mathsf{type} \mathbin{@} m \qquad \Gamma, v : (\mu \circ \nu \mid A) \vdash M_1 : B[\mathsf{mod}_{\nu}(v)/x] \mathbin{@} m\end{array}}{\Gamma \vdash \mathsf{let}_{\mu}\ \mathsf{mod}_{\nu}(x) \leftarrow M_0 \ \mathsf{in}\ M_1 : B[M_0/x] \mathbin{@} m}$$

**Remark 1.1.3** (The operation $(-)^{\alpha}$ on modal types)**.** We define

$$\langle \mu \mid A \rangle^{\sigma} = \langle \mu \mid A^{\sigma'} \rangle \quad \text{where } \sigma'(x) = \sigma(x) \star 1_{\mu} \qquad \triangleleft$$

The type theory also includes $\Sigma$-types (in negative/projection style), as well as $\mathsf{Id}$-types. However, these are given parametrically in the mode $m$. In the particular case of the $\mathsf{Id}$-eliminator, $\mathsf{J}$, we simply ensure the assumptions $x : (1 \mid A), y : (1 \mid A), p : (1 \mid \mathsf{Id}_A(x,y))$ are all given under the modality 1.

## 1.2 Algebraic Syntax

In order to introduce MTT rigorously we present it as *algebraic syntax* [Car78; Tay99; KKA19]. This is to say that we write out a specification of our type theory in the language of *generalized algebraic theories*. This approach offers a number of technical advantages:

1. We absolve ourselves from having to prove tedious syntactic metatheorems, e.g. admissibility of substitution.

2. We automatically obtain a notion of *model* of our theory, which is given in entirely algebraic terms.

3. In addition to a definition of model, We also automatically obtain a notion of *homomorphism of models*. This might be rather strict and not fit for every purpose, but it does subsume the *semantic interpretation map* (see next points).

4. We automatically obtain an *initial model* for the algebraic theory, which we consider as our main formal object of study.

5. The unique morphism of models from this initial model to any other is the *semantic interpretation map*. We then have no need to explicitly describe these semantic maps and prove that they are well-defined on non-unique derivations, as done in e.g. [Hof97].

Amongst other things, the theorems proven in the aforementioned works imply all of the above points.

While this approach is straightforward and uncluttered, some readers might object to the lack of a more traditional formulation, e.g. a traditional *named syntax* with variables and a substitution operation, like the one we informally presented in Section 1.1. We believe that it is indeed possible to define such a syntax and systematically show how to *elaborate* its terms to the algebraic syntax.

However, such a named syntax would not be suitable for implementation: for that purpose we ought to develop an entirely different *algorithmic syntax*. We believe that such a syntax can be constructed as an extension of existing *bidirectional* presentations of type theory [Coq96; PT00] as has been done for existing modal calculi [GSB19]. Such a bidirectional presentation would be a midpoint between the maximally annotated algebraic syntax we present here and the more typical unannotated named syntax from Section 1.1; it would contain only a select few annotations to ensure the decidability of typechecking while maintaining readability. The development of an algorithmic syntax and the proof of its decidability is a substantial endeavor (requiring a proof of normalization), and is orthogonal to the foundational metatheory of MTT that we are currently developing. We therefore leave its construction for future work. Moreover, we refrain from developing a formal account of a traditional named syntax which would be superseded by such an algorithmic syntax. Instead, we content ourselves with working formally only with the algebraic syntax at present.

The definition of the algebraic syntax begins by defining the different sorts (contexts, types, terms, etc.) that constitute our type theory. In order to support multiple modes, our sorts will be parameterised in modes. Thus, rather than having e.g. a sort of types, we have a sort of types *at mode* $m \in \mathcal{M}$, and likewise for contexts at mode $m$, terms at mode $m$, etc.

Moreover, we take care to index our types by *levels*. The reason for doing so is to introduce a hierarchy of sizes, which we can then use to introduce universes. For simplicity, we stratify our types in two levels, drawn from the set $\mathcal{L} = \{0, 1\}$. There are no technical obstacles on the way to a richer hierarchy, but these two suffice for our purposes: we aim to divide our types into *small types* (i.e. those that can be reified in a universe) and *large types* (which also include the universe itself). This allows a simplified treatment of universes *à la Coquand* [Coq13]. In order to enforce cumulativity we will also include an explicit *coercion* operator, which includes small types into large types.

The levelled approach raises an obvious question: on which level should we admit terms, 0 or 1? We could follow the approach of Sterling [Ste19] in allowing terms at all levels. Unfortunately, this formulation requires the introduction of term-level coercions, which bring with them numerous equations that relate term formers at different levels with the coercions. Thus, for the sake of simplicity we will only allow the formation of terms at large types. Similarly, we will only allow the extension of a context by a large type.

MTT has four sorts, which are introduced by the following rules:

$$\frac{m : \mathcal{M}}{\mathsf{ctx}_m \ \mathsf{sort}} \qquad \frac{\ell : \mathcal{L} \qquad m : \mathcal{M} \qquad \Gamma : \mathsf{ctx}_m}{\mathsf{type}^{\ell}_m(\Gamma)\ \mathsf{sort}} \qquad \frac{m : \mathcal{M} \qquad \Gamma : \mathsf{ctx}_m \qquad A : \mathsf{type}^1_m(\Gamma)}{\mathsf{tm}_m(\Gamma, A)\ \mathsf{sort}}$$

$$\frac{m : \mathcal{M} \qquad \Gamma, \Delta : \mathsf{ctx}_m}{\mathsf{sb}_m(\Gamma, \Delta)\ \mathsf{sort}}$$

In the interest of clarity, we will use the following shorthands to denote elements of these sorts:

$$\begin{aligned}
\Gamma\ \mathsf{ctx} \mathrel{@} m &\triangleq \Gamma : \mathsf{ctx}_m \\
\Gamma \vdash A\ \mathsf{type}_\ell \mathrel{@} m &\triangleq A : \mathsf{type}^\ell_m(\Gamma) \\
\Gamma \vdash M : A \mathrel{@} m &\triangleq M : \mathsf{tm}_m(\Gamma, A) \\
\Gamma \vdash \delta : \Delta \mathrel{@} m &\triangleq \delta : \mathsf{sb}_m(\Gamma, \Delta)
\end{aligned}$$

Even though we will make ample use of this more familiar notation, we will try to adhere to the rigours of algebraic syntax, in particular by carefully avoiding overloading/ambiguity and enforcing presupposition.

The type theory itself is introduced by the following judgments. In the interest of brevity, we elide the following standard rules:

- the congruence rules pushing substitutions inside terms and types;
- the congruence rules pushing explicit lifts inside of type formers;
- the associativity, unit, and weakening laws for the explicit substitutions;
- the $\beta$ laws for $\Pi$, $\Sigma$, $\mathbb{B}$ and $\mathsf{Id}$;
- the $\eta$ laws for $\Pi$ and $\Sigma$;

Note also that given $\Delta \vdash \gamma : \Gamma \mathrel{@} m$ and $\Gamma.🔒_\mu \vdash A\ \mathsf{type}_\ell \mathrel{@} m$ we write

$$\Delta.(\mu \mid A[\gamma.🔒_\mu]) \vdash \gamma^+ \triangleq (\gamma \circ \uparrow).\mathbf{v}_0 : \Gamma.(\mu \mid A) \mathrel{@} m$$

for the ‘weakened’ substitution.

$$\boxed{\Gamma\ \mathsf{ctx} \mathrel{@} m}$$

$$\frac{}{\cdot\ \mathsf{ctx} \mathrel{@} m} \qquad \frac{\Gamma\ \mathsf{ctx} \mathrel{@} m \qquad \mu : \mathrm{Hom}_{\mathcal{M}}(n, m)}{\Gamma.🔒_\mu\ \mathsf{ctx} \mathrel{@} n}$$

$$\frac{\Gamma\ \mathsf{ctx} \mathrel{@} m \qquad \mu : \mathrm{Hom}_{\mathcal{M}}(n, m) \qquad \Gamma.🔒_\mu \vdash A\ \mathsf{type}_1 \mathrel{@} n}{\Gamma.(\mu \mid A)\ \mathsf{ctx} \mathrel{@} m}$$

$$\frac{\Gamma\ \mathsf{ctx} \mathrel{@} m \qquad \nu : \mathrm{Hom}_{\mathcal{M}}(o,n) \qquad \mu : \mathrm{Hom}_{\mathcal{M}}(n,m)}{\Gamma.\lock_{\mu}.\lock_{\nu} = \Gamma.\lock_{\mu\circ\nu}\ \mathsf{ctx} \mathrel{@} o} \qquad \frac{\Gamma\ \mathsf{ctx} \mathrel{@} m}{\Gamma.\lock_{1} = \Gamma\ \mathsf{ctx} \mathrel{@} m}$$

$$\boxed{\Gamma \vdash \delta : \Delta \mathrel{@} m}$$

$$\frac{\Gamma\ \mathsf{ctx} \mathrel{@} m}{\Gamma \vdash \cdot : \cdot \mathrel{@} m} \qquad \frac{\Gamma\ \mathsf{ctx} \mathrel{@} n \qquad \mu : \mathrm{Hom}_{\mathcal{M}}(n,m) \qquad \Gamma.\lock_{\mu} \vdash A\ \mathsf{type}_1 \mathrel{@} n}{\Gamma.(\mu \mid A) \vdash {\uparrow} : \Gamma \mathrel{@} m} \qquad \frac{\Gamma\ \mathsf{ctx} \mathrel{@} m}{\Gamma \vdash \mathsf{id} : \Gamma \mathrel{@} m}$$

$$\frac{\Gamma, \Delta, \Xi\ \mathsf{ctx} \mathrel{@} m \qquad \Gamma \vdash \gamma : \Delta \mathrel{@} m \qquad \Delta \vdash \delta : \Xi \mathrel{@} m}{\Gamma \vdash \delta \circ \gamma : \Xi \mathrel{@} m}$$

$$\frac{\Gamma, \Delta\ \mathsf{ctx} \mathrel{@} m \qquad \mu : \mathrm{Hom}_{\mathcal{M}}(n,m) \qquad \Gamma \vdash \delta : \Delta \mathrel{@} m}{\Gamma.\lock_{\mu} \vdash \delta.\lock_{\mu} : \Delta.\lock_{\mu} \mathrel{@} n}$$

$$\frac{\Gamma\ \mathsf{ctx} \mathrel{@} m \qquad \mu, \nu : \mathrm{Hom}_{\mathcal{M}}(n,m) \qquad \alpha : \nu \Rightarrow \mu}{\Gamma.\lock_{\mu} \vdash \key^{\alpha}_{\Gamma} : \Gamma.\lock_{\nu} \mathrel{@} n}$$

$$\frac{\begin{array}{c}\mu : \mathrm{Hom}_{\mathcal{M}}(n,m) \\ \Gamma, \Delta\ \mathsf{ctx} \mathrel{@} m \qquad \Gamma \vdash \delta : \Delta \mathrel{@} m \qquad \Delta.\lock_{\mu} \vdash A\ \mathsf{type}_1 \mathrel{@} n \qquad \Gamma.\lock_{\mu} \vdash M : A[\delta.\lock_{\mu}] \mathrel{@} n\end{array}}{\Gamma \vdash \delta.M : \Delta.(\mu \mid A) \mathrel{@} m}$$

$$\boxed{\Gamma \vdash \gamma = \delta : \Delta \mathrel{@} m}$$

$$\frac{\begin{array}{c}\Gamma_0, \Gamma_1\ \mathsf{ctx} \mathrel{@} n \qquad \Delta\ \mathsf{ctx} \mathrel{@} n \qquad \Delta.\lock_{\mu} \vdash A\ \mathsf{type}_1 \mathrel{@} m \\ \mu : \mathrm{Hom}_{\mathcal{M}}(m,n) \qquad \Gamma_0 \vdash \gamma : \Gamma_1 \mathrel{@} n \qquad \Gamma_1 \vdash \delta : \Delta \mathrel{@} n \qquad \Gamma_1.\lock_{\mu} \vdash M : A[\delta.\lock_{\mu}] \mathrel{@} m\end{array}}{\Gamma_0 \vdash (\delta.M) \circ \gamma = (\delta \circ \gamma).M[\gamma.\lock_{\mu}] : \Delta.(\mu \mid A) \mathrel{@} n}$$

$$\frac{\Gamma, \Delta\ \mathsf{ctx} \mathrel{@} o \qquad \mu : \mathrm{Hom}_{\mathcal{M}}(m,n) \qquad \nu : \mathrm{Hom}_{\mathcal{M}}(n,o) \qquad \Gamma \vdash \delta : \Delta \mathrel{@} m}{\Gamma.\lock_{\nu\circ\mu} \vdash \delta.\lock_{\nu\circ\mu} = \delta.\lock_{\nu}.\lock_{\mu} : \Delta.\lock_{\nu\circ\mu} \mathrel{@} m}$$

$$\frac{\Gamma, \Delta\ \mathsf{ctx} \mathrel{@} m \qquad \Gamma \vdash \delta : \Delta \mathrel{@} m}{\Gamma \vdash \delta.\lock_{1} = \delta : \Delta \mathrel{@} m} \qquad \frac{\Gamma\ \mathsf{ctx} \mathrel{@} n \qquad \mu : \mathrm{Hom}_{\mathcal{M}}(m,n)}{\Gamma.\lock_{\mu} \vdash \mathsf{id}.\lock_{\mu} = \mathsf{id} : \Gamma.\lock_{\mu} \mathrel{@} m}$$

$$\frac{\Gamma, \Delta, \Xi\ \mathsf{ctx} \mathrel{@} n \qquad \mu : \mathrm{Hom}_{\mathcal{M}}(m,n) \qquad \Gamma \vdash \delta : \Delta \mathrel{@} n \qquad \Delta \vdash \xi : \Xi \mathrel{@} n}{\Gamma.\lock_{\mu} \vdash (\xi \circ \delta).\lock_{\mu} = \xi.\lock_{\mu} \circ \delta.\lock_{\mu} : \Xi.\lock_{\mu} \mathrel{@} m}$$

$$\frac{\Gamma\ \mathsf{ctx} \mathrel{@} n \qquad \mu : \mathrm{Hom}_{\mathcal{M}}(m,n)}{\Gamma.\lock_{\mu} \vdash \mathsf{id} = \key^{1_{\mu}}_{\Gamma} : \Gamma.\lock_{\mu} \mathrel{@} m}$$

$$\frac{\Gamma, \Delta\ \mathsf{ctx} \mathrel{@} n \qquad \mu, \nu : \mathrm{Hom}_{\mathcal{M}}(m,n) \qquad \Gamma \vdash \delta : \Delta \mathrel{@} n \qquad \alpha : \nu \Rightarrow \mu}{\Gamma.\lock_{\mu} \vdash \key^{\alpha}_{\Gamma} \circ (\delta.\lock_{\mu}) = (\delta.\lock_{\nu}) \circ \key^{\alpha}_{\Delta} : \Delta.\lock_{\nu} \mathrel{@} m}$$

$$\frac{\Gamma\ \mathsf{ctx} \mathrel{@} m \qquad \mu_0, \mu_1, \mu_2 : \mathrm{Hom}_{\mathcal{M}}(n,m) \qquad \alpha_0 : \mu_0 \Rightarrow \mu_1 \qquad \alpha_1 : \mu_1 \Rightarrow \mu_2}{\Gamma.\lock_{\mu_2} \vdash \key^{\alpha_1\circ\alpha_0}_{\Gamma} = \key^{\alpha_0}_{\Gamma} \circ \key^{\alpha_1}_{\Gamma} : \Gamma.\lock_{\mu_0} \mathrel{@} n}$$

$$\frac{\Gamma\ \mathsf{ctx} @ m \qquad \nu_0,\nu_1 : \mathrm{Hom}_{\mathcal{M}}(o,n) \qquad \mu_0,\mu_1 : \mathrm{Hom}_{\mathcal{M}}(n,m) \qquad \beta : \nu_0 \Rightarrow \nu_1 \qquad \alpha : \mu_0 \Rightarrow \mu_1}{\Gamma.\text{🔒}_{\mu_0\circ\nu_0} \vdash \text{🔑}^{\alpha\star\beta}_{\Gamma} = \text{🔑}^{\alpha}_{\Gamma}.\text{🔒}_{\nu_1} \circ \text{🔑}^{\beta}_{\Gamma.\text{🔒}_{\mu_0}} : \Gamma.\text{🔒}_{\mu_1\circ\nu_1} @ o}$$

$$\boxed{\Gamma \vdash A\ \mathsf{type}_\ell @ m}$$

$$\frac{\Gamma\ \mathsf{ctx} @ m}{\Gamma \vdash \mathbb{B}\ \mathsf{type}_\ell @ m} \qquad \frac{\Gamma\ \mathsf{ctx} @ m}{\Gamma \vdash \mathsf{U}\ \mathsf{type}_1 @ m} \qquad \frac{\Gamma\ \mathsf{ctx} @ m \qquad \Gamma \vdash M : \mathsf{U} @ m}{\Gamma \vdash \mathsf{El}(M)\ \mathsf{type}_0 @ m}$$

$$\frac{\ell \le \ell' \qquad \Gamma\ \mathsf{ctx} @ m \qquad \Gamma \vdash A\ \mathsf{type}_\ell @ m}{\Gamma \vdash {\Uparrow}A\ \mathsf{type}_{\ell'} @ m}$$

$$\frac{\Gamma\ \mathsf{ctx} @ m \qquad \Gamma \vdash A\ \mathsf{type}_\ell @ m \qquad \Gamma \vdash M, N : {\Uparrow}A @ m}{\Gamma \vdash \mathsf{Id}_A(M,N)\ \mathsf{type}_\ell @ m}$$

$$\frac{\Gamma\ \mathsf{ctx} @ m \qquad \mu : \mathrm{Hom}_{\mathcal{M}}(n,m) \qquad \Gamma.\text{🔒}_{\mu} \vdash A\ \mathsf{type}_\ell @ n}{\Gamma \vdash \langle \mu \mid A \rangle\ \mathsf{type}_\ell @ m}$$

$$\frac{\mu : \mathrm{Hom}_{\mathcal{M}}(n,m) \qquad \Gamma\ \mathsf{ctx} @ m \qquad \Gamma.\text{🔒}_{\mu} \vdash A\ \mathsf{type}_\ell @ n \qquad \Gamma.(\mu \mid {\Uparrow}A) \vdash B\ \mathsf{type}_\ell @ m}{\Gamma \vdash (\mu \mid A) \to B\ \mathsf{type}_\ell @ m}$$

$$\frac{\Gamma\ \mathsf{ctx} @ m \qquad \Gamma \vdash A\ \mathsf{type}_\ell @ m \qquad \Gamma.(1 \mid {\Uparrow}A) \vdash B\ \mathsf{type}_\ell @ m}{\Gamma \vdash \sum(A,B)\ \mathsf{type}_\ell @ m}$$

$$\frac{\Gamma, \Delta\ \mathsf{ctx} @ m \qquad \Delta \vdash A\ \mathsf{type}_\ell @ m \qquad \Gamma \vdash \delta : \Delta @ m}{\Gamma \vdash A[\delta]\ \mathsf{type}_\ell @ m}$$

$$\boxed{\Gamma \vdash A = B\ \mathsf{type}_\ell @ m}$$

$$\frac{\mu : \mathrm{Hom}_{\mathcal{M}}(n,m) \qquad \Gamma, \Delta\ \mathsf{ctx} @ m \qquad \Gamma \vdash \delta : \Delta @ m \qquad \Delta.\text{🔒}_{\mu} \vdash A\ \mathsf{type}_\ell @ n}{\Gamma \vdash \langle \mu \mid A \rangle[\delta] = \langle \mu \mid A[\delta.\text{🔒}_{\mu}] \rangle\ \mathsf{type}_\ell @ m}$$

$$\frac{\begin{array}{c}\mu : \mathrm{Hom}_{\mathcal{M}}(n,m) \\ \Gamma, \Delta\ \mathsf{ctx} @ m \qquad \Gamma \vdash \delta : \Delta @ m \qquad \Gamma.\text{🔒}_{\mu} \vdash A\ \mathsf{type}_\ell @ n \qquad \Gamma.(\mu \mid {\Uparrow}A) \vdash B\ \mathsf{type}_\ell @ m\end{array}}{\Gamma \vdash ((\mu \mid A) \to B)[\delta] = (\mu \mid A[\delta.\text{🔒}_{\mu}]) \to B[\delta^{+}]\ \mathsf{type}_\ell @ m}$$

$$\frac{\Gamma \vdash A\ \mathsf{type}_\ell @ m}{\Gamma \vdash {\Uparrow}A = A\ \mathsf{type}_\ell @ m} \qquad \frac{\ell_0 \le \ell_1 \le \ell_2 \qquad \Gamma \vdash A\ \mathsf{type}_{\ell_0} @ m}{\Gamma \vdash {\Uparrow}{\Uparrow}A = {\Uparrow}A\ \mathsf{type}_{\ell_2} @ m}$$

$$\frac{\begin{array}{c}\ell \le \ell' \\ \mu \in \mathrm{Hom}_{\mathcal{M}}(n,m) \qquad \Gamma\ \mathsf{ctx} @ m \qquad \Gamma.\text{🔒}_{\mu} \vdash A\ \mathsf{type}_\ell @ n \qquad \Gamma.(\mu \mid {\Uparrow}A) \vdash B\ \mathsf{type}_\ell @ m\end{array}}{\Gamma \vdash {\Uparrow}((\mu \mid A) \to B) = (\mu \mid {\Uparrow}A) \to {\Uparrow}B\ \mathsf{type}_{\ell'} @ m}$$

$$\frac{\Gamma\ \mathsf{ctx} @ m \qquad \Gamma \vdash A\ \mathsf{type}_0 @ m}{\Gamma \vdash \mathsf{El}(\mathsf{Code}(A)) = A\ \mathsf{type}_0 @ m}$$

$$\boxed{\Gamma \vdash M : A @ m}$$

$$\frac{\mu : \mathrm{Hom}_{\mathcal{M}}(n,m) \qquad \Gamma\ \mathsf{ctx} @ m \qquad \Gamma.\text{🔒}_\mu \vdash A\ \mathsf{type}_1 @ n}{\Gamma.(\mu \mid A).\text{🔒}_\mu \vdash \mathbf{v}_0 : A[\uparrow.\text{🔒}_\mu] @ n} \qquad \frac{\Gamma\ \mathsf{ctx} @ m}{\Gamma \vdash \mathsf{tt}, \mathsf{ff} : \mathbb{B} @ m}$$

$$\frac{\begin{array}{c}\Gamma\ \mathsf{ctx} @ m \\ \Gamma.(1 \mid \mathbb{B}) \vdash A\ \mathsf{type}_1 @ m \qquad \Gamma \vdash M_t : A[\mathsf{id}.\mathsf{tt}] @ m \qquad \Gamma \vdash M_f : A[\mathsf{id}.\mathsf{ff}] @ m \qquad \Gamma \vdash N : \mathbb{B} @ m\end{array}}{\Gamma \vdash \mathsf{if}(A; M_t; M_f; N) : A[\mathsf{id}.N] @ m}$$

$$\frac{\Gamma\ \mathsf{ctx} @ m \qquad \Gamma \vdash A\ \mathsf{type}_0 @ m}{\Gamma \vdash \mathsf{Code}(A) : \mathsf{U} @ m} \qquad \frac{\Gamma\ \mathsf{ctx} @ m \qquad \Gamma \vdash A\ \mathsf{type}_1 @ m \qquad \Gamma \vdash M : A @ m}{\Gamma \vdash \mathsf{refl}(M) : \mathsf{Id}_A(M, M) @ m}$$

$$\frac{\begin{array}{c}\Gamma\ \mathsf{ctx} @ m \qquad \Gamma \vdash A\ \mathsf{type}_1 @ m \qquad \Gamma.(1 \mid A).(1 \mid A[\uparrow]).(1 \mid \mathsf{Id}_{A[\uparrow^2]}(\mathbf{v}_1, \mathbf{v}_0)) \vdash B\ \mathsf{type}_1 @ m \\ \Gamma.(1 \mid A) \vdash M : B[\uparrow.\mathbf{v}_0.\mathbf{v}_0.\mathsf{refl}(\mathbf{v}_0)] @ m \qquad \Gamma \vdash N_0, N_1 : A @ m \qquad \Gamma \vdash P : \mathsf{Id}_A(N_0, N_1) @ m\end{array}}{\Gamma \vdash \mathsf{J}(B, M, P) : B[\mathsf{id}.N_0.N_1.P] @ m}$$

$$\frac{\Gamma\ \mathsf{ctx} @ m \qquad \mu : \mathrm{Hom}_{\mathcal{M}}(n,m) \qquad \Gamma.\text{🔒}_\mu \vdash A\ \mathsf{type}_1 @ n \qquad \Gamma.\text{🔒}_\mu \vdash M : A @ n}{\Gamma \vdash \mathsf{mod}_\mu(M) : \langle \mu \mid A \rangle @ m}$$

$$\frac{\begin{array}{c}\nu : \mathrm{Hom}_{\mathcal{M}}(o,n) \\ \mu : \mathrm{Hom}_{\mathcal{M}}(n,m) \qquad \Gamma\ \mathsf{ctx} @ m \qquad \Gamma.\text{🔒}_\mu.\text{🔒}_\nu \vdash A\ \mathsf{type}_1 @ o \qquad \Gamma.\text{🔒}_\mu \vdash M_0 : \langle \nu \mid A \rangle @ n \\ \Gamma.(\mu \mid \langle \nu \mid A \rangle) \vdash B\ \mathsf{type}_1 @ m \qquad \Gamma.(\mu \circ \nu \mid A) \vdash M_1 : B[\uparrow.\mathsf{mod}_\nu(\mathbf{v}_0)] @ m\end{array}}{\Gamma \vdash \mathsf{let}_\mu\ \mathsf{mod}_\nu(\_) \leftarrow M_0\ \mathsf{in}\ M_1 : B[\mathsf{id}.M_0] @ m}$$

$$\frac{\begin{array}{c}\mu : \mathrm{Hom}_{\mathcal{M}}(n,m) \\ \Gamma\ \mathsf{ctx} @ m \qquad \Gamma.\text{🔒}_\mu \vdash A\ \mathsf{type}_1 @ n \qquad \Gamma.(\mu \mid A) \vdash B\ \mathsf{type}_1 @ m \qquad \Gamma.(\mu \mid A) \vdash M : B @ m\end{array}}{\Gamma \vdash \lambda(M) : (\mu \mid A) \to B @ m}$$

$$\frac{\begin{array}{c}\mu : \mathrm{Hom}_{\mathcal{M}}(n,m) \qquad \Gamma\ \mathsf{ctx} @ m \qquad \Gamma.\text{🔒}_\mu \vdash A\ \mathsf{type}_1 @ n \\ \Gamma.(\mu \mid A) \vdash B\ \mathsf{type}_1 @ m \qquad \Gamma \vdash M_0 : (\mu \mid A) \to B @ m \qquad \Gamma.\text{🔒}_\mu \vdash M_1 : A @ n\end{array}}{\Gamma \vdash M_0(M_1) : B[\mathsf{id}.M_1] @ m}$$

$$\frac{\begin{array}{c}\Gamma\ \mathsf{ctx} @ m \\ \Gamma \vdash A\ \mathsf{type}_1 @ m \qquad \Gamma.(1 \mid A) \vdash B\ \mathsf{type}_1 @ m \qquad \Gamma \vdash M_0 : A @ m \qquad \Gamma \vdash M_1 : B[\mathsf{id}.M_0] @ m\end{array}}{\Gamma \vdash (M_0, M_1) : \textstyle\sum(A, B) @ m}$$

$$\frac{\Gamma\ \mathsf{ctx} @ m \qquad \Gamma \vdash A\ \mathsf{type}_1 @ m \qquad \Gamma.(1 \mid A) \vdash B\ \mathsf{type}_1 @ m \qquad \Gamma \vdash M : \textstyle\sum(A, B) @ m}{\Gamma \vdash \mathsf{pr}_0(M) : A @ m \qquad \Gamma \vdash \mathsf{pr}_1(M) : B[\mathsf{id}.\mathsf{pr}_0(M)] @ m}$$

$$\frac{\Gamma, \Delta\ \mathsf{ctx} @ m \qquad \Delta \vdash A\ \mathsf{type}_1 @ m \qquad \Gamma \vdash \delta : \Delta @ m \qquad \Delta \vdash M : A @ m}{\Gamma \vdash M[\delta] : A[\delta] @ m}$$

$$\boxed{\Gamma \vdash M = N : A @ m}$$

$$\frac{\mu : \mathrm{Hom}_{\mathcal{M}}(n,m) \qquad \Gamma \vdash \delta : \Delta @ m \qquad \Delta.\text{🔒}_\mu \vdash A\ \mathsf{type}_1 @ n \qquad \Gamma.\text{🔒}_\mu \vdash M : A[\delta.\text{🔒}_\mu] @ n}{\Gamma.\text{🔒}_\mu \vdash \mathbf{v}_0[(\delta.M).\text{🔒}_\mu] = M : A[\delta.\text{🔒}_\mu] @ n}$$

$$\frac{\Gamma\ \mathsf{ctx} @ m \qquad \Gamma \vdash M : \mathsf{U} @ m}{\Gamma \vdash \mathsf{Code}(\mathsf{El}(M)) = M : \mathsf{U} @ m}$$

$$\frac{\begin{array}{c}\nu : \mathrm{Hom}_{\mathcal{M}}(o,n) \\ \mu : \mathrm{Hom}_{\mathcal{M}}(n,m) \qquad \Gamma\ \mathsf{ctx} @ m \qquad \Gamma.\text{🔒}_{\mu}.\text{🔒}_{\nu} \vdash A\ \mathsf{type}_1 @ o \qquad \Gamma.\text{🔒}_{\mu}.\text{🔒}_{\nu} \vdash M_0 : A @ o \\ \Gamma.(\mu \mid \langle \nu \mid A\rangle) \vdash B\ \mathsf{type}_1 @ m \qquad \Gamma.(\mu \circ \nu \mid A) \vdash M_1 : B[\uparrow.\mathsf{mod}_{\mu}(\mathbf{v}_0)] @ m\end{array}}{\Gamma \vdash \mathsf{let}_{\mu}\ \mathsf{mod}_{\nu}(\_) \leftarrow \mathsf{mod}_{\nu}(M_0)\ \mathsf{in}\ M_1 = M_1[\mathsf{id}.M_0] : B[\mathsf{id}.\mathsf{mod}_{\nu}(M_0)] @ m}$$

$$\frac{\begin{array}{c}\mu : \mathrm{Hom}_{\mathcal{M}}(n,m) \\ \Delta, \Gamma\ \mathsf{ctx} @ m \qquad \Gamma \vdash \delta : \Delta @ m \qquad \Delta.\text{🔒}_{\mu} \vdash A\ \mathsf{type}_1 @ n \qquad \Delta.\text{🔒}_{\mu} \vdash M : A @ n\end{array}}{\Gamma \vdash \mathsf{mod}_{\mu}(M)[\delta] = \mathsf{mod}_{\mu}(M[\delta.\text{🔒}_{\mu}]) : \langle \mu \mid A[\delta.\text{🔒}_{\mu}]\rangle @ m}$$

$$\frac{\begin{array}{c}\nu : \mathrm{Hom}_{\mathcal{M}}(o,n) \qquad \mu : \mathrm{Hom}_{\mathcal{M}}(n,m) \\ \Gamma, \Delta\ \mathsf{ctx} @ m \qquad \Gamma \vdash \delta : \Delta @ m \qquad \Delta.\text{🔒}_{\mu}.\text{🔒}_{\nu} \vdash A\ \mathsf{type}_1 @ o \qquad \Delta.\text{🔒}_{\mu} \vdash M_0 : \langle \nu \mid A\rangle @ n \\ \Delta.(\mu \mid \langle \nu \mid A\rangle) \vdash B\ \mathsf{type}_1 @ m \qquad \Delta.(\mu \circ \nu \mid A) \vdash M_1 : B[\uparrow.\mathsf{mod}_{\mu}(\mathbf{v}_0)] @ m\end{array}}{\Gamma \vdash (\mathsf{let}_{\mu}\ \mathsf{mod}_{\nu}(\_) \leftarrow M_0\ \mathsf{in}\ M_1)[\delta] = \mathsf{let}_{\mu}\ \mathsf{mod}_{\nu}(\_) \leftarrow M_0[\delta.\text{🔒}_{\mu}]\ \mathsf{in}\ M_1[\delta^{+}] : B[\delta.M_0[\delta.\text{🔒}_{\mu}]] @ m}$$

**Notation 1.2.1.** For the sake of readability, when opening a modal term under the modality 1 we will suppress the 1 in the $\mathsf{let}_1$ part of the term, and write $\mathsf{let}\ \mathsf{mod}_{\mu}(\_) \leftarrow M\ \mathsf{in}\ N$ instead.

**Notation 1.2.2.** When working informally we will freely use "pattern matching" notation when $\lambda$-abstracting over either dependent pairs or modal types. For example, we will write $\lambda(x_0, x_1).\ x_0$ to mean the term $\lambda u.\ \mathsf{pr}_0(u)$, and $\lambda\langle \mu \mid x\rangle.\ N$ to mean the term $\lambda u.\ \mathsf{let}\ \mathsf{mod}_{\mu}(x) \leftarrow u\ \mathsf{in}\ N$.

**Remark 1.2.3.** Notice that the locks $-.\text{🔒}_{\mu}$ act as functors, and the keys $\text{🔑}^{-}_{\_}$ act as natural transformations. Consequently, our type theory is forced to contain a calculus of (strict) 2-categories, which comes with equations that govern the behaviour of 1-cells and 2-cells. Indeed, the equations for keys given above suffice to derive the two ways of internally stating the *interchange laws*, viz.

$$\frac{\begin{array}{c}\Gamma\ \mathsf{ctx} @ m \qquad \nu_0, \nu_1, \nu_2 : \mathrm{Hom}_{\mathcal{M}}(o,n) \qquad \mu_0, \mu_1, \mu_2 : \mathrm{Hom}_{\mathcal{M}}(n,m) \\ \alpha_0 : \mu_0 \Rightarrow \mu_1 \qquad \alpha_1 : \mu_1 \Rightarrow \mu_2 \qquad \beta_0 : \nu_0 \Rightarrow \nu_1 \qquad \beta_1 : \nu_1 \Rightarrow \nu_2\end{array}}{\Gamma.\text{🔒}_{\mu_2 \circ \nu_2} \vdash \text{🔑}^{\alpha_0 \star \beta_0}_{\Gamma} \circ \text{🔑}^{\alpha_1 \star \beta_1}_{\Gamma} = \text{🔑}^{\alpha_1 \circ \alpha_0}_{\Gamma}.\text{🔒}_{\nu_0} \circ \text{🔑}^{\beta_1 \circ \beta_0}_{\Gamma.\text{🔒}_{\mu_2}} : \Gamma.\text{🔒}_{\mu_0 \circ \nu_0} @ o}$$

$$\frac{\begin{array}{c}\Gamma\ \mathsf{ctx} @ m \qquad \nu_0, \nu_1, \nu_2 : \mathrm{Hom}_{\mathcal{M}}(o,n) \qquad \mu_0, \mu_1, \mu_2 : \mathrm{Hom}_{\mathcal{M}}(n,m) \\ \alpha_0 : \mu_0 \Rightarrow \mu_1 \qquad \alpha_1 : \mu_1 \Rightarrow \mu_2 \qquad \beta_0 : \nu_0 \Rightarrow \nu_1 \qquad \beta_1 : \nu_1 \Rightarrow \nu_2\end{array}}{\Gamma.\text{🔒}_{\mu_2 \circ \nu_2} \vdash \text{🔑}^{\alpha_0 \star \beta_0}_{\Gamma} \circ \text{🔑}^{\alpha_1 \star \beta_1}_{\Gamma} = \text{🔑}^{\beta_1 \circ \beta_0}_{\Gamma.\text{🔒}_{\mu_0}} \circ \text{🔑}^{\alpha_1 \circ \alpha_0}_{\Gamma}.\text{🔒}_{\nu_2} : \Gamma.\text{🔒}_{\mu_0 \circ \nu_0} @ o}$$

In fact, the second version of the interchange law follows from the first one and the equation that expresses the naturality of $\text{🔑}^{-}_{\_}$. Conversely, except the two laws for the identity 2-cell and naturality, the given equations follow from one of the two interchange laws. $\lhd$

**Remark 1.2.4** (Universes à la Coquand)**.** It may come as a surprise that, even though the universe of MTT is 'à la Tarski,' we do not require rules for introducing *codes* in $\mathsf{U}$. For example, one would expect a $\mathsf{U}$-constructor $(\mu \mid A) \,\widehat{\to}\, B : \mathsf{U}$ that would mimic the $\Pi$-formation rule, introduced for example by a rule of the following form:

$$\frac{\Gamma\ \mathsf{ctx} @ m \qquad \mu \in \mathrm{Hom}_{\mathcal{M}}(n,m) \qquad \Gamma.\text{🔒}_{\mu} \vdash A : \mathsf{U} @ n \qquad \Gamma.(\mu \mid A) \vdash B : \mathsf{U} @ m}{\Gamma \vdash (\mu \mid A) \,\widehat{\to}\, B : \mathsf{U} @ m}$$

In fact, these codes are *definable*: it suffices to use the 'inverse' to $\mathsf{El}(-)$:

$$(\mu \mid A) \,\widehat{\to}\, B \triangleq \mathsf{Code}((\mu \mid \mathsf{El}(A)) \to \mathsf{El}(B))$$

This rule automatically satisfies the desired property, i.e. that a $\Pi$-code decodes to the actual $\Pi$-type. We thus avoid the tedious exercise of postulating enough constructors to construct all the desired universe codes. In an informal sense, $\mathsf{Code}(-)$ and $\mathsf{El}(-)$ witness an *isomorphism* between terms of type $U$ and types of size $0$.

In our examples we will often suppress both $\mathsf{El}(-)$ and $\mathsf{Code}(-)$, and in some straightfoward cases we even elide the coercion $\Uparrow -$. This not only makes our terms more perspicuous, but can also be formally justified by an *elaboration procedure* which inserts the missing isomorphisms and coercions when needed. $\lhd$

## 1.3 Basic Examples of MTT

We present a few terms that are definable irrespective of the underlying mode theory. We first write these out in informal syntax, and discuss them. Finally, we collect the proper algebraic terms that we formally consider.

### 1.3.1 Functoriality of the Modal Types

First, we show that the type former $\langle \mu \mid A \rangle$ is functorial in the modality $\mu$ up to equivalence. To begin, the following terms demonstarte that $\langle 1 \mid A \rangle$ is equivalent to $A$.

$$
\begin{aligned}
&\mathbf{triv} &&: (x : A) \to \langle 1 \mid A \rangle \\
&\mathbf{triv} &&\triangleq \lambda x.\ \mathsf{mod}_1(x) \\[4pt]
&\mathbf{triv}^{\text{-}1} &&: (x : \langle 1 \mid A \rangle) \to A \\
&\mathbf{triv}^{\text{-}1} &&\triangleq \lambda x.\ \mathsf{let}\ \mathsf{mod}_1(y) \leftarrow x\ \mathsf{in}\ y \\[4pt]
&\_ &&: (x : A) \to \mathsf{Id}_A(x, \mathbf{triv}^{\text{-}1}(\mathbf{triv}(x))) \\
&\_ &&\triangleq \lambda x.\ \mathsf{refl}(x) \\[4pt]
&\_ &&: (x : \langle 1 \mid A \rangle) \to \mathsf{Id}_{\langle 1 \mid A \rangle}(x, \mathbf{triv}(\mathbf{triv}^{\text{-}1}(x))) \\
&\_ &&\triangleq \lambda x.\ \mathsf{let}\ \mathsf{mod}_1(y) \leftarrow x\ \mathsf{in}\ \mathsf{refl}(\mathsf{mod}_1(y))
\end{aligned}
$$

The unnamed paths above then provide internal equalities between $\mathbf{triv}(\mathbf{triv}^{\text{-}1}(M))$ and $M : \langle \mu \mid A \rangle$ and $\mathbf{triv}^{\text{-}1}(\mathbf{triv}(N))$ and $N : A$.

Next, we can show how to compose modalities: we construct an equivalence $\langle \nu \mid \langle \mu \mid A \rangle \rangle \cong \langle \nu \circ \mu \mid A \rangle$. This equivalence is particularly important, as it enables the interaction between modalities.

$$
\begin{aligned}
&\mathbf{comp}_{\mu,\nu} &&: (x : \langle \nu \mid \langle \mu \mid A \rangle \rangle) \to \langle \nu \circ \mu \mid A \rangle \\
&\mathbf{comp}_{\mu,\nu} &&\triangleq \lambda x.\ \mathsf{let}\ \mathsf{mod}_\nu(y) \leftarrow x\ \mathsf{in}\ (\mathsf{let}_\nu\ \mathsf{mod}_\mu(z) \leftarrow y\ \mathsf{in}\ \mathsf{mod}_{\nu\circ\mu}(z)) \\[4pt]
&\mathbf{comp}^{\text{-}1}_{\mu,\nu} &&: (x : \langle \nu \circ \mu \mid A \rangle) \to \langle \nu \mid \langle \mu \mid A \rangle \rangle \\
&\mathbf{comp}^{\text{-}1}_{\mu,\nu} &&\triangleq \lambda x.\ \mathsf{let}\ \mathsf{mod}_{\nu\circ\mu}(y) \leftarrow x\ \mathsf{in}\ \mathsf{mod}_\nu(\mathsf{mod}_\mu(y)) \\[4pt]
&\_ &&: (x : \langle \nu \mid \langle \mu \mid A \rangle \rangle) \to \mathsf{Id}_{\langle \nu \mid \langle \mu \mid A \rangle \rangle}(x, \mathbf{comp}^{\text{-}1}_{\mu,\nu}(\mathbf{comp}_{\mu,\nu}(x))) \\
&\_ &&\triangleq \lambda x.\ \mathsf{let}\ \mathsf{mod}_\nu(y) \leftarrow x\ \mathsf{in}\ \mathsf{let}_\nu\ \mathsf{mod}_\mu(z) \leftarrow y\ \mathsf{in}\ \mathsf{refl}(\mathsf{mod}_\nu(\mathsf{mod}_\mu(z))) \\[4pt]
&\_ &&: (x : \langle \nu \circ \mu \mid A \rangle) \to \mathsf{Id}_{\langle \nu\circ\mu \mid A \rangle}(x, \mathbf{comp}_{\mu,\nu}(\mathbf{comp}^{\text{-}1}_{\mu,\nu}(x))) \\
&\_ &&\triangleq \lambda x.\ \mathsf{let}\ \mathsf{mod}_{\nu\circ\mu}(y) \leftarrow x\ \mathsf{in}\ \mathsf{refl}(\mathsf{mod}_{\nu\circ\mu}(y))
\end{aligned}
$$

This example crucially depends on the additional $\nu$ annotation in $\mathsf{let}_\nu$. We thus see that the modal elimination rule surreptitiously introduces functoriality with respect to modalities.

### 1.3.2 Modal Types Preserve Dependent Sums

We can also show that modal types preserve dependent sums up to equivalence. The essential content of this theorem derives from the fact that $\langle \mu \mid - \rangle$ behaves in a right-adjoint-like manner.

To begin, we consider the simpler case of products, i.e. the case where $B$ does *not* depend on $A$.

With some sugar for pattern matching on pairs, we construct the terms

$$\begin{aligned}
&p_0 \;:\; (x : \langle \mu \mid \textstyle\sum(A, B)\rangle) \to \textstyle\sum(\langle \mu \mid A\rangle, \langle \mu \mid B\rangle) \\
&p_0 \triangleq \lambda x.\ \mathsf{let}\ \mathsf{mod}_\mu(y) \leftarrow x\ \mathsf{in}\ (\mathsf{mod}_\mu(\mathsf{pr}_0(y)), \mathsf{mod}_\mu(\mathsf{pr}_1(y)))
\end{aligned}$$

$$\begin{aligned}
&p_1 \;:\; (x : \textstyle\sum(\langle \mu \mid A\rangle, \langle \mu \mid B\rangle)) \to \langle \mu \mid \textstyle\sum(A, B)\rangle \\
&p_1 \triangleq \lambda(x_0, x_1).\ \mathsf{let}\ \mathsf{mod}_\mu(y_0) \leftarrow x_0\ \mathsf{in}\ (\mathsf{let}\ \mathsf{mod}_\mu(y_1) \leftarrow x_1\ \mathsf{in}\ \mathsf{mod}_\mu((y_0, y_1)))
\end{aligned}$$

$$\begin{aligned}
&_ \;:\; (x : \langle \mu \mid \textstyle\sum(A, B)\rangle) \to \mathsf{Id}_{\langle \mu \mid \sum(A, B)\rangle}(x, p_1(p_0(x))) \\
&_ \triangleq \lambda x.\ \mathsf{let}\ \mathsf{mod}_\mu(y) \leftarrow x\ \mathsf{in}\ \mathsf{refl}(\mathsf{mod}_\mu(y))
\end{aligned}$$

$$\begin{aligned}
&_ \;:\; (x : \textstyle\sum(\langle \mu \mid A\rangle, \langle \mu \mid B\rangle)) \to \mathsf{Id}_{\sum(\langle \mu \mid A\rangle, \langle \mu \mid B\rangle)}(x, p_0(p_1(x))) \\
&_ \triangleq \lambda(x_0, x_1).\ \mathsf{let}\ \mathsf{mod}_\mu(y_0) \leftarrow x_0\ \mathsf{in}\ (\mathsf{let}\ \mathsf{mod}_\mu(y_1) \leftarrow x_1\ \mathsf{in}\ \mathsf{refl}((\mathsf{mod}_\mu(y_0), \mathsf{mod}_\mu(y_1))))
\end{aligned}$$

That the penultimate term typechecks depends on the $\eta$-rule for $\Sigma$-types, which is part of our system.

Adapting this statement to dependent sums (i.e. the case where $B$ depends on $A$) is not straightforward. In fact, even the theorem itself is hard to state. Writing $B[a]$ for a type with a free variable $a : A$, one end of the equivalence is $\langle \mu \mid (a : A) \times B[a]\rangle)$, but it is not immediately evident what the other one should be: the obvious choice of $\langle \mu \mid A\rangle) \times B[a]$ might not even be well-typed, as $B[a]$ is no longer under the modality $\mu$. We must hence apply a *correction*, and replace $B$ with

$$B'[x] \triangleq \mathsf{let}\ \mathsf{mod}_\mu(y) \leftarrow x\ \mathsf{in}\ \langle \mu \mid B[y]\rangle$$

This type opens the modal variable $x : \langle \mu \mid A\rangle$ and substitutes it in the correct modal context. We can then define

$$\begin{aligned}
&p_0 \;:\; (x : \langle \mu \mid (a : A) \times B[a]\rangle) \to (a : \langle \mu \mid A\rangle) \times B'[a] \\
&p_0 \triangleq \lambda x.\ \mathsf{let}\ \mathsf{mod}_\mu(y) \leftarrow x\ \mathsf{in}\ (\mathsf{mod}_\mu(\mathsf{pr}_0(y)), \mathsf{mod}_\mu(\mathsf{pr}_1(y)))
\end{aligned}$$

$$\begin{aligned}
&p_1 \;:\; (x : (a : \langle \mu \mid A\rangle) \times B'[a]) \to \langle \mu \mid (a : A) \times B[a]\rangle \\
&p_1 \triangleq \lambda(x_0, x_1).\ (\mathsf{let}\ \mathsf{mod}_\mu(y_0) \leftarrow x_0\ \mathsf{in}\ (\lambda x_1'.\ \mathsf{let}\ \mathsf{mod}_\mu(y_1) \leftarrow x_1\ \mathsf{in}\ \mathsf{mod}_\mu((y_0, y_1))))(x_1)
\end{aligned}$$

$$\begin{aligned}
&_ \;:\; (x : \langle \mu \mid (a : A) \times B[a]\rangle) \to \mathsf{Id}_{\langle \mu \mid (a:A) \times B[a]\rangle}(x, p_1(p_0(x))) \\
&_ \triangleq \lambda x.\ \mathsf{let}\ \mathsf{mod}_\mu(y) \leftarrow x\ \mathsf{in}\ \mathsf{refl}(\mathsf{mod}_\mu(y))
\end{aligned}$$

$$\begin{aligned}
&_ \;:\; (x : \langle \mu \mid (a : A) \times B[a]\rangle) \to \mathsf{Id}_{\langle \mu \mid (a:A) \times B[a]\rangle}(x, p_0(p_1(x))) \\
&_ \triangleq \lambda(x_0, x_1).\ (\mathsf{let}\ \mathsf{mod}_\mu(y_0) \leftarrow x_0\ \mathsf{in}\ (\lambda x_1'.\ \mathsf{let}\ \mathsf{mod}_\mu(y_1) \leftarrow x_1\ \mathsf{in}\ \mathsf{refl}((\mathsf{mod}_\mu(y_0), \mathsf{mod}_\mu(y_1)))))(x_1)
\end{aligned}$$

### 1.3.3 Dependent K

Another interesting and useful term that we obtain is a version of *dependent K*, which is a dependent version of the K (for Kripke) axiom of modal logic [Bir+20]. The dependent K axiom states that modal types weakly distribute over dependent products. As before, a slight contortion of the codomain $B[x]$ to

$$B'[x] \triangleq \mathsf{let}\ \mathsf{mod}_\mu(y) \leftarrow x\ \mathsf{in}\ \langle \mu \mid B(y)\rangle$$

is necessary. We can then define the term

$$\begin{aligned}
&k \;:\; (x : \langle \mu \mid (a : A) \to B[a]\rangle) \to (a : \langle \mu \mid A\rangle) \to B'[a] \\
&k \triangleq \lambda x_0.\ \lambda x_1.\ \mathsf{let}\ \mathsf{mod}_\mu(y_0) \leftarrow x_0\ \mathsf{in}\ \mathsf{let}\ \mathsf{mod}_\mu(y_1) \leftarrow x_1\ \mathsf{in}\ \mathsf{mod}_\mu(y_0(y_1))
\end{aligned}$$

This term will be convenient for some of our central examples. Consequently, in keeping with the literature we will use the shorthand

$$M \circledast_\mu N \triangleq k(M, N)$$

We will also occasionally suppress the subscript when it can be reasonably inferred from context.

### 1.3.4 2-Cells Between Modalities Induce Natural Transformations

Thus far our tautologies have only dealt with reflecting equalities in the mode theory theory into equivalences in the type theory. We can also reflect the enrichment, 2-cells, into the type theory. These give rise not to equivalences, but functions (natural transformations).

For instance, let us suppose that we have $\alpha : \mu \Rightarrow \nu$.

$$\begin{aligned} t \ &: (x : \langle \mu \mid A \rangle) \to \langle \nu \mid A^{\alpha} \rangle \\ t \ &\triangleq \lambda x.\ \mathsf{let}\ \mathsf{mod}_{\mu}(y) \leftarrow x\ \mathsf{in}\ \mathsf{mod}_{\nu}(y^{\alpha}) \end{aligned}$$

$$t_{\mathrm{e}} \triangleq \lambda(\mathsf{let}\ \mathsf{mod}_{\mu}(\_\_) \leftarrow \mathbf{v}_0\ \mathsf{in}\ \mathsf{mod}_{\nu}(\mathbf{v}_0[\text{🔑}^{\alpha}_{\cdot.(1\mid\langle\mu\mid A\rangle).(\mu\mid A)}]))$$

In this case, because the sugared term hides some interesting aspects of the proof, we have also included the explicit term for reference. This term appears frequently in examples, and so we have again fixed some dedicated notation. We will write $\mathbf{coe}[\alpha : \mu \Rightarrow \nu](M)$ for $t(M)$, and $\mathbf{coe}[\mu \leq \nu](M)$ for the special case that $\mathcal{M}$ is only a poset-enriched category.

### 1.3.5 The Explicit Terms

$$\begin{aligned} i_0 \ &: \langle 1 \mid A \rangle \to A[\uparrow] \\ i_0 \ &\triangleq \lambda(\mathsf{let}\ \mathsf{mod}_1(\_\_) \leftarrow \mathbf{v}_0\ \mathsf{in}\ \mathbf{v}_0) \end{aligned}$$

$$\begin{aligned} i_1 \ &: A \to \langle 1 \mid A[\uparrow] \rangle \\ i_1 \ &\triangleq \lambda(\mathsf{mod}_1(\mathbf{v}_0)) \end{aligned}$$

$$\begin{aligned} c_0 \ &: \langle \mu \circ \nu \mid A \rangle \to \langle \mu \mid \langle \nu \mid A[\uparrow] \rangle \rangle \\ c_0 \ &\triangleq \lambda(\mathsf{let}\ \mathsf{mod}_{\mu\circ\nu}(\_\_) \leftarrow \mathbf{v}_0\ \mathsf{in}\ \mathsf{mod}_{\mu}(\mathsf{mod}_{\nu}(\mathbf{v}_1))) \end{aligned}$$

$$\begin{aligned} c_1 \ &: \langle \mu \mid \langle \nu \mid A \rangle \rangle \to \langle \mu \circ \nu \mid A[\uparrow] \rangle \\ c_1 \ &\triangleq \lambda(\mathsf{let}\ \mathsf{mod}_{\mu}(\_\_) \leftarrow \mathbf{v}_0\ \mathsf{in}\ \mathsf{let}_{\mu}\ \mathsf{mod}_{\nu}(\_\_) \leftarrow \mathbf{v}_0\ \mathsf{in}\ \mathsf{mod}_{\mu\circ\nu}(\mathbf{v}_0)) \end{aligned}$$

$$B' \triangleq \mathsf{let}\ \mathsf{mod}_{\mu}(\_\_) \leftarrow \mathbf{v}_0\ \mathsf{in}\ \langle \mu \mid B[\uparrow^2.\text{🔒}_{\mu}.\mathbf{v}_0] \rangle$$

$$\begin{aligned} p_0 \ &: \langle \mu \mid \textstyle\sum(A, B) \rangle \to \textstyle\sum(\langle \mu \mid A \rangle, B') \\ p_0 \ &\triangleq \lambda(\mathsf{let}\ \mathsf{mod}_{\mu}(\_\_) \leftarrow \mathbf{v}_0\ \mathsf{in}\ (\mathsf{mod}_{\mu}(\mathsf{pr}_0(\mathbf{v}_0)), \mathsf{mod}_{\mu}(\mathsf{pr}_1(\mathbf{v}_0)))]) \end{aligned}$$

$$\begin{aligned} p_1 \ &: \textstyle\sum(\langle \mu \mid A \rangle, B') \to \langle \mu \mid \textstyle\sum(A, B) \rangle \\ p_1 \ &\triangleq \lambda((\mathsf{let}\ \mathsf{mod}_{\mu}(\_\_) \leftarrow \mathsf{pr}_0(\mathbf{v}_1)\ \mathsf{in}\ \lambda(\mathsf{let}\ \mathsf{mod}_{\mu}(\_\_) \leftarrow \mathbf{v}_0\ \mathsf{in}\ \mathsf{mod}_{\mu}((\mathbf{v}_2, \mathbf{v}_0))))(\mathsf{pr}_1(\mathbf{v}_0))) \end{aligned}$$

$$\begin{aligned} k \ &: \langle \mu \mid A \to B \rangle \to \langle \mu \mid A \rangle \to B' \\ k \ &\triangleq \lambda(\lambda(\mathsf{let}\ \mathsf{mod}_{\mu}(\_\_) \leftarrow \mathbf{v}_1\ \mathsf{in}\ \mathsf{let}\ \mathsf{mod}_{\mu}(\_\_) \leftarrow \mathbf{v}_1\ \mathsf{in}\ \mathsf{mod}_{\mu}(\mathbf{v}_1(\mathbf{v}_0)))) \end{aligned}$$

## 1.4 Models of MTT

In Section 1.2 we introduced an algebraic syntax for MTT, and stated that this is our main formal object of study. One advantage of this first-order algebraic definition is that it enables us to use a powerful kit of technology [Car78; Tay99; KKA19] which—amongst other things—automatically guarantees that (a) there exists a *category of models*, and (b) the syntax constitutes an initial object for this category.

However, the models that inhabit this category are exactly algebras for this generalized algebraic theory, which are intractable to directly construct. In this section, we undertake the task of decomposing this algebraic notion of model into smaller, more practicable parcels. These will be given in the style of *natural models* [Awo18], which are a category-theoretic reformulation of *categories with families (CwFs)* [Dyb96].

We find this relatively recent technology helpful, as it concisely encodes the many naturality conditions normally required of a CwF. Moreover, natural models also aid with uncovering the implicit universal properties of type-theoretic connectives, which are not evident in a CwF formulation.

In Section 1.4.1 we will deconstruct and analyze the standard notion of model given by the generalized algebraic theory of MTT in terms of natural models. Following that, in Section 1.4.2 we will show that the stronger notion of *dependent right adjoint* can be used to define such a model. Finally, in Section 1.4.3 we will discuss the morphisms between standard models of MTT.

### 1.4.1 Models of the GAT

#### *Context Structure*

First, we observe that a model of our type theory must contain a set of contexts at each mode. Along with the substitutions found at each mode $m \in \mathcal{M}$—which can be composed associatively and come with a unit—these sets are readily seen to form a category, for which we write $\mathcal{C}[m]$.

Furthermore, the functions that interpret the locks on contexts must be functors: the rules for equality of contexts clearly ask that $-.\text{🔒}_\mu$ distributes over composition of substitutions, and preserves the identity substitution. Thus, for each modality $\mu : \mathrm{Hom}_{\mathcal{M}}(n, m)$ we obtain a functor

$$[\![\text{🔒}_\mu]\!] : \mathcal{C}[m] \to \mathcal{C}[n]$$

Notice that this is contravariant in the modality, as it is *the action of locks on contexts.* Similarly, the equations for each 2-cell $\alpha : \nu \Rightarrow \mu$ in $\mathcal{M}$ induce a natural transformation

$$[\![\text{🔑}^\alpha]\!] : [\![\text{🔒}_\mu]\!] \Rightarrow [\![\text{🔒}_\nu]\!]$$

This is also contravariant in the 2-cell $\alpha$, as is the action of *keys* on locks.

We can package these aspects of our model in the following definition.

**Definition 1.4.1.** A *context structure* for a mode theory $\mathcal{M}$ is a (strict) 2-functor

$$[\![-]\!] : \mathcal{M}^{\mathsf{coop}} \to \mathbf{Cat}_1$$

where $\mathcal{M}^{\mathsf{coop}}$ is the 2-category $\mathcal{M}$ with the direction of *both* 1-cells and 2-cells reversed, and $\mathbf{Cat}_1$ is the full subcategory of (large) categories with a terminal object.

This double contravariance may seem peculiar at first sight. However, recall that the 2-category $\mathcal{M}$ specifies the behaviour of the modal types $\langle \mu \mid - \rangle$, which are supposed to have a right-adjoint-like behaviour, with the corresponding left-adjoint-like operators being the lock functors $-.\text{🔒}_\mu$. Being left-adjoint-like, the interpretation $[\![\text{🔒}_-]\!]$ of each modality will behave with variance opposite to the specification of $\mathcal{M}$. Of course, this is merely an analogy, as these constructors are not truly adjoints but merely present similar behaviour.

*Types and Context Extension*

We now study the structure necessary to encode types, terms and context extension. We temporarily ignore the universe, the details of which we will discuss in more depth in Section 1.4.1.

We begin with the definition of a representable natural transformation:

**Definition 1.4.2** (Representable natural transformation)**.** Let $\mathbb{C}$ be a small category, and let $P, Q : \mathbf{PSh}(\mathbb{C})$ be presheaves on $\mathbb{C}$. A natural transformation $\alpha : P \Rightarrow Q$ is *representable* just if for every $\Gamma : \mathbb{C}$ and $x : \mathbf{y}(\Gamma) \Rightarrow P$ (equivalently $x \in P(\Gamma)$) there exists a $y : \mathbf{y}(\Delta) \Rightarrow Q$ (equivalently $y \in Q(\Gamma)$) and a morphism $\gamma : \Delta \to \Gamma$ in $\mathbb{C}$ such that there is a pullback square

$$\begin{tikzcd} \mathbf{y}(\Delta) \arrow[r, "y"] \arrow[d, "\mathbf{y}(\gamma)"'] \arrow[dr, phantom, very near start, "\lrcorner"] & Q \arrow[d, "\alpha"] \\ \mathbf{y}(\Gamma) \arrow[r, "x"'] & P \end{tikzcd}$$

The following notion of model of type theory is used by Awodey [Awo18].

**Definition 1.4.3** (Natural model)**.** Let $\mathbb{C}$ be a small category with a terminal object $\mathbf{1}$, and let and $\widetilde{\mathcal{T}}, \mathcal{T} : \mathbf{PSh}(\mathbb{C})$. A *natural model of type theory* is a representable natural transformation $\tau : \widetilde{\mathcal{T}} \Rightarrow \mathcal{T}$.

It is shown in *op. cit.* that this corresponds to the usual notion of CwF, and that one can use it this formulation to write down very concise definitions of the gadgets necessary to interpret various type formers, and in particular intensional identity types. We note that the representability of the natural transformation $\tau : \widetilde{\mathcal{T}} \Rightarrow \mathcal{T}$ is a clever way to encode context extension and comprehension in a manner that automatically ensures naturality with respect to substitution; see [Fio12; Awo18] for more details. Our objective here is to adapt this formulation to the multi-mode setting.

To begin, for each mode $m \in \mathcal{M}$ we define two presheaves $\mathcal{T}_m : \mathbf{PSh}(\mathcal{C}[m])$ and $\widetilde{\mathcal{T}}_m : \mathbf{PSh}(\mathcal{C}[m])$ on the context category $\mathcal{C}[m]$. The first one maps every $\Gamma : \mathcal{C}[m]$ to the set of types $\mathsf{type}^1_m(\Gamma)$ over it, and the second one maps $\Gamma$ to the pairs $(A \in \mathsf{type}^1_m(\Gamma), M \in \mathsf{tm}_m(\Gamma, A))$, i.e. the set of pointed types. We thus obtain a natural transformation $\tau_m : \widetilde{\mathcal{T}}_m \Rightarrow \mathcal{T}_m$, which maps each term-pair $(A, M)$ to the type $A$ at each context. It follows that the fibres of $\tau_m$ are the terms of a given type.

Our context extension rule postulates that for any object $\Gamma : \mathcal{C}[m]$, modality $\mu \in \mathrm{Hom}(n, m)$ and $A \in \mathsf{type}^1_n([\![\text{🔒}_\mu]\!]\Gamma)$ there exists an object $\Gamma.(\mu \mid A) : \mathcal{C}[m]$. This construction comes with a morphism $\mathbf{p} : \mathrm{Hom}_{\mathcal{C}[m]}(\Gamma.(\mu \mid A), \Gamma)$, and a term

$$\mathbf{q} \in \mathsf{tm}_n([\![\text{🔒}_\mu]\!](\Gamma.(\mu \mid A)), A[[\![\text{🔒}_\mu]\!](\mathbf{p})])$$

The object $\Gamma.(\mu \mid A)$ is universal with respect to $\mathbf{p}$ and $\mathbf{q}$, in the sense that for any object $\Delta : \mathcal{C}[m]$, morphism $\gamma \in \mathrm{Hom}_{\mathcal{C}[m]}(\Delta, \Gamma)$, and term $M \in \mathsf{tm}_n([\![\text{🔒}_\mu]\!]\Delta, A[\gamma.\text{🔒}_\mu])$ there is a *unique* substitution $\gamma.M : \Delta \to \Gamma.(\mu \mid A)$ such that

$$\mathbf{p} \circ (\gamma.M) = \gamma : \Delta \to \Gamma \tag{1.4}$$
$$\mathbf{q}[(\gamma.M).\text{🔒}_\mu] = M : \mathsf{tm}_n([\![\text{🔒}_\mu]\!](\Gamma), A[\gamma.\text{🔒}_\mu]) \tag{1.5}$$

As usual, Eq. (1.5) is only well-typed because of Eq. (1.4). This definition is very close to the ordinary presentation of context extension in CwFs; the main difference is that we must account for the fact that the type by which we extend is found in a different mode than the context that is being extended.

With this in hand, we can encode modal context extension as follows. Writing $\lfloor - \rfloor$ for the Yoneda isomorphism, we require that for each $\mu : \mathrm{Hom}_{\mathcal{M}}(n, m)$, context $\Gamma : \mathcal{C}[m]$, and $A : \mathsf{type}^1_n([\![\text{🔒}_\mu]\!](\Gamma))$, there is a chosen context $\Gamma'$ (cf. $\Gamma.(\mu \mid A)$), a chosen morphism $\mathbf{p} : \Gamma' \to \Gamma$, and a chosen morphism $\lfloor \mathbf{q} \rfloor : \mathbf{y}([\![\text{🔒}_\mu]\!]\Gamma') \to \widetilde{\mathcal{T}_n}$ that makes the following square commute:

$$\begin{tikzcd}
\mathbf{y}([\![\text{🔒}_\mu]\!]\Gamma') \arrow[r, "\lfloor \mathbf{q} \rfloor"] \arrow[d, "\mathbf{y}([\![\text{🔒}_\mu]\!]\mathbf{p})"'] & \widetilde{\mathcal{T}_n} \arrow[d, "\tau_n"] \\
\mathbf{y}([\![\text{🔒}_\mu]\!]\Gamma) \arrow[r, "\lfloor A \rfloor"'] & \mathcal{T}_n
\end{tikzcd}$$

We have surreptitiously 'decoded' the top arrow into a term $\mathbf{q} \in \mathsf{tm}_n([\![\text{🔒}_\mu]\!](\Gamma'), A[[\![\text{🔒}_\mu]\!]\mathbf{p}])$ by using the Yoneda isomorphism and the fact the square commutes. This is notational convention we will silently use without comment when applicable.

We also require that $\Gamma'$, $\mathbf{p}$, and $\mathbf{q}$ are universal for this diagram. That is, given $\Delta : \mathcal{C}[m]$, $\gamma : \mathrm{Hom}_{\mathcal{C}[m]}(\Delta, \Gamma)$, and $\lfloor M \rfloor : \mathbf{y}([\![\text{🔒}_\mu]\!]\Delta) \Rightarrow \widetilde{\mathcal{T}_n}$, there must be a unique morphism $\gamma' : \Delta \to \Gamma'$ (which stands for $\gamma.M$) such that the following square commutes:

$$\begin{tikzcd}
\mathbf{y}([\![\text{🔒}_\mu]\!](\Delta)) \arrow[dr, dashed, "\mathbf{y}([\![\text{🔒}_\mu]\!]\gamma')"] \arrow[drr, bend left, "\lfloor M \rfloor"] \arrow[ddr, bend right, "\mathbf{y}([\![\text{🔒}_\mu]\!]\gamma)"'] & & \\
& \mathbf{y}([\![\text{🔒}_\mu]\!]\Gamma') \arrow[r, "\lfloor \mathbf{q} \rfloor"] \arrow[d, "\mathbf{y}([\![\text{🔒}_\mu]\!]\mathbf{p})"] & \widetilde{\mathcal{T}_n} \arrow[d, "\tau_n"] \\
& \mathbf{y}([\![\text{🔒}_\mu]\!]\Gamma) \arrow[r, "\lfloor A \rfloor"'] & \mathcal{T}_n
\end{tikzcd}$$

This diagram is not a pullback, but we can use the Yoneda lemma to make it into one. Recall that for any functor $f : \mathcal{C} \to \mathcal{D}$ we can define the precomposition functor $f^* : \mathbf{PSh}(\mathcal{D}) \to \mathbf{PSh}(\mathcal{C})$, which on objects is

$$f^*(P) \triangleq \mathcal{C} \xrightarrow{f} \mathcal{D} \xrightarrow{P} \mathbf{Set}$$

Then, for any $c : \mathcal{C}$ and $Q : \mathbf{PSh}(\mathcal{D})$ we can use the Yoneda lemma to establish a series of natural isomorphisms

$$\mathrm{Hom}_{\mathbf{PSh}(\mathcal{D})}(\mathbf{y}(f(c)), Q) \cong Q(f(c)) = f^*Q(c) \cong \mathrm{Hom}_{\mathbf{PSh}(\mathcal{C})}(\mathbf{y}(c), f^*Q)$$

We can then *transpose* the diagram in order to obtain

$$\begin{tikzcd}
\mathbf{y}(\Delta) \arrow[drr, bend left, "\lfloor M \rfloor"] \arrow[dr, dashed, "\gamma'"] \arrow[ddr, bend right, "\gamma"'] & & \\
& \mathbf{y}(\Gamma') \arrow[r, "\lfloor \mathbf{q} \rfloor"] \arrow[d, "\mathbf{y}(\mathbf{p})"'] & [\![\text{🔒}_\mu]\!]^* \widetilde{\mathcal{T}}_n \arrow[d, "{[\![\text{🔒}_\mu]\!]^* \tau_n}"] \\
& \mathbf{y}(\Gamma) \arrow[r, "\lfloor A \rfloor"'] & [\![\text{🔒}_\mu]\!]^* \mathcal{T}_n
\end{tikzcd} \tag{1.6}$$

where $\gamma' : \Delta \to \Gamma'$ is the unique arrow that makes the diagram commute.

Observe now that we are able to carry out this step whenever the right hand morphism of the transposed diagram is a natural model. We are thus led to the following definition.

**Definition 1.4.4.** A *modal natural model* on a context structure $[\![-]\!] : \mathcal{M}^{\mathsf{coop}} \to \mathbf{Cat}_1$ consists of a family of natural transformations of presheaves

$$\left(\tau_m : \widetilde{\mathcal{T}}_m \Rightarrow \mathcal{T}_m\right)_{m \in \mathcal{M}}$$

where $\widetilde{\mathcal{T}}_m, \mathcal{T}_m : \mathbf{PSh}(\mathcal{C}[m])$ such that for every $\mu : \mathrm{Hom}_{\mathcal{M}}(m, n)$ the natural transformation

$$[\![\text{🔒}_\mu]\!]^* \tau_n : [\![\text{🔒}_\mu]\!]^* \widetilde{\mathcal{T}}_n \Rightarrow [\![\text{🔒}_\mu]\!]^* \mathcal{T}_n$$

is a natural model.

We will write $\Gamma.(\mu \mid A)$ for the object $\Gamma'$ that makes (1.6) a pullback, as we do in the type theory.

**Remark 1.4.5.** Observe that $[\![\text{🔒}_\mu]\!] : \mathcal{C}[m] \to \mathcal{C}[n]$ and $[\![\text{🔒}_\mu]\!]^* : \mathbf{PSh}(\mathcal{C}[n]) \to \mathbf{PSh}(\mathcal{C}[m])$ are very different functors. The former, which is given as part of the definition of a model, is a functor between categories of contexts, and does not need to satisfy any particular properties. The latter, which is canonically defined once we specify $[\![\text{🔒}_\mu]\!]$, acts on *presheaves* on those categories of contexts, and it is well-known that it comes with both a left and a right adjoint, $[\![\text{🔒}_\mu]\!]_!$ and $[\![\text{🔒}_\mu]\!]_*$, given by left and right Kan extension respectively. This functor is used for technical purposes in the model, does not appear in the type theory, and neither it nor its adjoints need descend from presheaf categories to the (sub)categories of contexts. $\lhd$

*An Intermezzo: Higher-Order Abstract Syntax*

In order to show how to model the various type formers in the style of natural models we need a mechanism for representing binding structure in $\mathbf{PSh}(\mathcal{C}[m])$. We mostly recapitulate material found in [Awo18], which we then adapt to modal types.

The main device used for encoding binding structure is that of *polynomial endofunctors*. Given a 'display map' $\ell : E \to B$, we may use the internal language of the presheaf topos to define the corresponding polynomial functor $\mathbf{P}_{\ell : E \to B} : \mathbf{PSh}(\mathcal{C}[m]) \to \mathbf{PSh}(\mathcal{C}[m])$ by

$$\mathbf{P}_{\ell : E \to B}(A) \triangleq \sum_{b:B} A^{\ell^{-1}(b)}$$

When specialised to the 'modalised' natural model $\ell \triangleq [\![\text{🔒}_\mu]\!]^*(\tau_n) : [\![\text{🔒}_\mu]\!]^*\widetilde{\mathcal{T}_n} \Rightarrow [\![\text{🔒}_\mu]\!]^*\mathcal{T}_n$, this functor has a very useful property: morphisms $\mathbf{y}(\Gamma) \Rightarrow \mathbf{P}_{[\![\text{🔒}_\mu]\!]^*\tau_n}(\mathcal{T}_m)$ are in bijection with tuples

$$(A \in \mathcal{T}_n([\![\text{🔒}_\mu]\!](\Gamma)), B \in \mathcal{T}_m(\Gamma.(\mu \mid A)))$$

This enables the representation of a type $\Gamma.\text{🔒}_\mu \vdash A\ \mathsf{type}_1 \mathrel{@} n$ and a type $\Gamma.(\mu \mid A) \vdash B\ \mathsf{type}_1 \mathrel{@} m$ that modally depends on it as a single morphism $\mathbf{y}(\Gamma) \Rightarrow \mathbf{P}_{[\![\text{🔒}_\mu]\!]^*\tau_n}(\mathcal{T}_n)$. To prove this, we may show a more general

**Lemma 1.4.6.** *Morphisms $Y \xrightarrow{g} \mathbf{P}_{\ell:E\to B}(X)$ are in bijection with diagrams*

$$\begin{tikzcd}
X & Y \times_B E \arrow[l, "g_2"'] \arrow[r] \arrow[d] \arrow[dr, phantom, "\lrcorner", very near start] & E \arrow[d, "\ell"] \\
 & Y \arrow[r, "g_1"'] & B
\end{tikzcd}$$

The proof may be found in the paper by Awodey [Awo18, Lemma 5]. The above bijection is then demonstrated by taking $\ell$ to be the modalised natural model, $X \triangleq \mathcal{T}_n$, and $Y \triangleq \mathbf{y}(\Gamma)$.

### $\prod$ *Structure*

A model is equipped with a $\prod$-structure if for $\mu : \mathrm{Hom}_{\mathcal{M}}(n, m)$ we have a pullback square

$$\begin{tikzcd}
\mathbf{P}_{[\![\text{🔒}_\mu]\!]^*\tau_n}(\widetilde{\mathcal{T}_m}) \arrow[r, "\mathbf{lam}"] \arrow[d] \arrow[dr, phantom, "\lrcorner", very near start] & \widetilde{\mathcal{T}_m} \arrow[d, "\tau_m"] \\
\mathbf{P}_{[\![\text{🔒}_\mu]\!]^*\tau_n}(\mathcal{T}_m) \arrow[r, "\prod"'] & \mathcal{T}_m
\end{tikzcd}$$

Using the insight provided by Lemma 1.4.6, we see that the morphism $\prod$ models the formation rule, while **lam** models the introduction rule. The $\beta$-law for $\prod$-types is equivalent to the existence of a mediating morphism given by the pullback, and the $\eta$-law follows from its uniqueness. A detailed discussion of these points may be found in Awodey [Awo18].

### $\Sigma$ *Structure*

A model is equipped with a $\sum$-structure if for each $m : \mathcal{M}$ we have a pullback square

$$\begin{tikzcd}
\sum_{A:\mathcal{T}_m} \sum_{B:\mathcal{T}_m^{\tau_m^{-1}(A)}} \sum_{M:\tau_m^{-1}(A)} \tau_m^{-1}(B(M)) \arrow[r, "\mathbf{pair}"] \arrow[d] \arrow[dr, phantom, "\lrcorner", very near start] & \widetilde{\mathcal{T}_m} \arrow[d, "\tau_m"] \\
\mathbf{P}_{\tau_m}(\mathcal{T}_m) \arrow[r, "\sum"'] & \mathcal{T}_m
\end{tikzcd}$$

As with $\prod$-types, this precisely corresponds to the usual CwF formulation of $\sum$-types [Dyb96; Hof97]. Again, a more detailed discussion may be found in [Awo18], and so we omit the details.

*Modal Structure*

Interpreting the modal types $\langle \mu \mid A \rangle$ in the natural models style is a little more complicated. The reason is that $\langle \mu \mid A \rangle$ behaves very much like a *positive type former*, which comes with a 'let-style' pattern-matching eliminator, and no $\eta$-rule. These features render its behaviour closer to that of intensional identity types, and unlike the simpler pullback squares required of $\prod$ and $\sum$. We will describe this construction in two steps.

First, for each $\mu : \mathrm{Hom}_{\mathcal{M}}(n, m)$ the formation and introduction rules for $\langle \mu \mid - \rangle$ are given by a commuting square

$$\begin{tikzcd}[\![\text{🔒}_\mu]\!]^*\widetilde{\mathcal{T}}_n \arrow[r, "\mathbf{mod}_\mu"] \arrow[d, "[\![\text{🔒}_\mu]\!]^*\tau_n"'] & \widetilde{\mathcal{T}}_m \arrow[d, "\tau_m"] \\ [\![\text{🔒}_\mu]\!]^*\mathcal{T}_n \arrow[r, "\mathbf{Mod}_\mu"'] & \mathcal{T}_m \end{tikzcd} . \tag{1.7}$$

It is easy to see that, by Yoneda, $\mathbf{Mod}_\mu$ may be used to map every type $\Gamma.\text{🔒}_\mu \vdash A$ $\mathsf{type}_1$ $@\, n$ to a type $\Gamma \vdash \langle \mu \mid A \rangle$ $\mathsf{type}_1$ $@\, m$, and similarly the arrow $\mathbf{mod}_\mu$ can be used to model the introduction rule. Unfortunately, asking that this square be a pullback is too strong a requirement with respect to the elimination rule. In fact, we will see in Section 1.4.2 that it corresponds precisely to $\mathbf{Mod}_\mu$ being a *dependent right adjoint* [Bir+20]. We may instead phrase the elimination rule in terms of the existence of a *lifting structure* for the above diagram. We define these in the internal language (i.e. extensional type theory) of the presheaf topos $\mathbf{PSh}(\mathcal{C}[m])$, which forces them to be *natural*.[3]

**Definition 1.4.7** (Left lifting structure)**.** Given presheaves $\vdash A, I, B$ type, a family $b : B \vdash E[b]$ type and a section $a : A \vdash i[a] : I$, we define the type $\vdash i[-] \pitchfork E[-]$ type of *left lifting structures* for $i$ with respect to $E$ to be

$$i[-] \pitchfork E[-] \triangleq \prod_{C : I \to B} \prod_{c : \prod_{a:A} E[C(i[a])]} \left\{ j : \prod_{p:I} E[C(p)] \mid \forall a : A.\, j(i[a]) = c(a) \right\}$$

Informally, left lifting structures provide *diagonal fillers* $j$ for the diagram

$$\begin{tikzcd} A \arrow[r, "\langle C(i[-]), c \rangle"] \arrow[d, "i[-]"'] & \sum_{b:B} E[b] \arrow[d, "\pi_1"] \\ I \arrow[r, "C"'] \arrow[ur, dotted, "j"] & B \end{tikzcd}$$

Intuitively, $C : I \to B$ is the *motive* of an elimination: we would like to prove $E[C(p)]$ for all $p : I$. At the same time, $c : \prod_{a:A} E[C(i[a])]$ is a given section that specifies the desired computational behaviour of this elimination at the 'special case' $A$. The left lifting structure then provides a section $j$ of $E[-]$ defined on all of $I$. This section is above $C$, and extends $c$. Note that these fillers are not necessarily unique. Moreover, they are automatically *natural*: as all the types involved in this definition are closed, we are at liberty to weaken the context.

[3] This formulation has its origins in unpublished work by Jonathan Sterling, Daniel Gratzer, Carlo Angiuli, and Lars Birkedal.

This style of lifting structure is an essential ingredient in recent work on models of intensional identity types. First, they play an important rôle in natural models: Awodey [Awo18, Lemma 19] shows that they precisely correspond to enriched left lifting properties in the sense of categorical homotopy theory [Rie14, §13]. In fact, the above definition given above is a word-for-word restatement in the internal language. Second, such lifting structures are also central devices in internal presentations of models of cubical type theory, in particular the recent work of Orton and Pitts [OP18].

We can now approach this in a manner similar to that used for intensional identity types in *op. cit.* Recall that the elimination rule for $\langle \nu \mid A \rangle$ is

$$\frac{\begin{array}{c}\nu : \mathrm{Hom}_{\mathcal{M}}(o,n) \\ \mu : \mathrm{Hom}_{\mathcal{M}}(n,m) \qquad \Gamma\ \mathsf{ctx} \mathbin{@} m \qquad \Gamma.\text{\faLock}_{\mu}.\text{\faLock}_{\nu} \vdash A\ \mathsf{type}_1 \mathbin{@} o \qquad \Gamma.\text{\faLock}_{\mu} \vdash M_0 : \langle \nu \mid A \rangle \mathbin{@} n \\ \Gamma.(\mu \mid \langle \nu \mid A \rangle) \vdash B\ \mathsf{type}_1 \mathbin{@} m \qquad \Gamma.(\mu \circ \nu \mid A) \vdash M_1 : B[\uparrow.\mathsf{mod}_{\nu}(\mathbf{v}_0)] \mathbin{@} m\end{array}}{\Gamma \vdash \mathsf{let}_{\mu}\ \mathsf{mod}_{\nu}(\_) \leftarrow M_0\ \mathsf{in}\ M_1 : B[\mathsf{id}.M_0] \mathbin{@} m}$$

The first thing we ought to do is remove the 'implicit cut' with $M_0$. We construct the substitution

$$\Gamma.(\mu \mid \langle \nu \mid A \rangle).(\mu \circ \nu \mid A[\uparrow.\text{\faLock}_{\mu \circ \nu}]) \vdash \sigma \triangleq \uparrow^2.\mathbf{v}_0 : \Gamma.(\mu \circ \nu \mid A) \mathbin{@} m$$

It then suffices to construct the elimination rule

$$\frac{\begin{array}{c}\nu : \mathrm{Hom}_{\mathcal{M}}(o,n) \qquad \mu : \mathrm{Hom}_{\mathcal{M}}(n,m) \qquad \Gamma\ \mathsf{ctx} \mathbin{@} m \qquad \Gamma.\text{\faLock}_{\mu}.\text{\faLock}_{\nu} \vdash A\ \mathsf{type}_1 \mathbin{@} o \\ \Gamma.(\mu \mid \langle \nu \mid A \rangle) \vdash B\ \mathsf{type}_1 \mathbin{@} m \qquad \Gamma.(\mu \circ \nu \mid A) \vdash M_1 : B[\uparrow.\mathsf{mod}_{\nu}(\mathbf{v}_0)] \mathbin{@} m\end{array}}{\Gamma.(\mu \mid \langle \nu \mid A \rangle) \vdash \mathsf{let}_{\mu}\ \mathsf{mod}_{\nu}(\_) \leftarrow \mathbf{v}_0\ \mathsf{in}\ M_1[\sigma] : B \mathbin{@} m}$$

because we can calculate that

$$\Gamma \vdash (\mathsf{let}_{\mu}\ \mathsf{mod}_{\nu}(\_) \leftarrow \mathbf{v}_0\ \mathsf{in}\ M_1[\sigma])[\mathsf{id}.M_0] = \mathsf{let}_{\mu}\ \mathsf{mod}_{\nu}(\_) \leftarrow M_0\ \mathsf{in}\ M_1 : B[\mathsf{id}.M_0] \mathbin{@} m$$

We can rephrase this as the existence of a diagonal filler in the diagram

$$\begin{tikzcd}
\mathbf{y}(\Gamma.(\mu \circ \nu \mid A)) \arrow[r, "\lfloor M_1[\sigma] \rfloor"] \arrow[d, "\mathbf{y}(\uparrow.\mathsf{mod}_{\nu}(\mathbf{v}_0))"'] & \widetilde{\mathcal{T}}_m \arrow[d, "\tau_m"] \\
\mathbf{y}(\Gamma.(\mu \mid \langle \nu \mid A \rangle)) \arrow[r, "\lfloor B \rfloor"'] \arrow[ur, "\lfloor \mathsf{let}_{\mu}\ \mathsf{mod}_{\nu}(_) \leftarrow \mathbf{v}_0\ \mathsf{in}\ M_1[\sigma] \rfloor" description] & \mathcal{T}_m
\end{tikzcd}$$

We can use a left lifting structure on a carefully chosen slice category to obtain such diagonal fillers. The internal language approach still applies, because of the well-known lemma stating that the slice of a presheaf topos is also a presheaf topos, but over the corresponding category of elements instead.. In symbols, the lemma states that for any $P : \mathbf{PSh}(\mathcal{C})$ we have an equivalence $\mathbf{PSh}(\mathcal{C})/P \simeq \mathbf{PSh}(\int_{\mathcal{C}} P)$.

First, given $\nu : \mathrm{Hom}_{\mathcal{M}}(o, n)$ we construct the following pullback:

$$\begin{tikzcd}
{[\![\text{🔒}_\nu]\!]^*\widetilde{\mathcal{T}}_o} \arrow[dr, dashed, "m"] \arrow[drr, bend left, "\mathbf{mod}_\nu"] \arrow[ddr, bend right, "{[\![\text{🔒}_\nu]\!]^*\tau_o}"'] & & \\
& M \arrow[r] \arrow[d, "h"'] \arrow[dr, phantom, very near start, "\lrcorner"] & \widetilde{\mathcal{T}}_n \arrow[d, "\tau_n"] \\
& {[\![\text{🔒}_\nu]\!]^*\mathcal{T}_o} \arrow[r, "\mathbf{Mod}_\nu"'] & \mathcal{T}_n
\end{tikzcd}$$

The outer commuting square is that given by the formation and introduction for $\langle \nu \mid - \rangle$, as in (1.7). Intuitively, $M$ is a 'generic $\nu$-modal terms object' that consists of terms $\Gamma \vdash M : \langle \nu \mid A \rangle \mathbin{@} n$, where $\Gamma.\text{🔒}_\nu \vdash A\ \mathsf{type}_1 \mathbin{@} o$. We know that $[\![\text{🔒}_\mu]\!]^*$ has a left adjoint, so it preserves pullbacks. Applying it to this diagram yields

$$\begin{tikzcd}
{[\![\text{🔒}_{\mu\circ\nu}]\!]^*\widetilde{\mathcal{T}}_o} \arrow[dr, dashed, "{[\![\text{🔒}_\mu]\!]^*m}"] \arrow[drr, bend left, "{[\![\text{🔒}_\mu]\!]^*\mathbf{mod}_\nu}"] \arrow[ddr, bend right, "{[\![\text{🔒}_{\mu\circ\nu}]\!]^*\tau_o}"'] & & \\
& {[\![\text{🔒}_\mu]\!]^*M} \arrow[r] \arrow[d, "{[\![\text{🔒}_\mu]\!]^*h}"'] \arrow[dr, phantom, very near start, "\lrcorner"] & {[\![\text{🔒}_\mu]\!]^*\widetilde{\mathcal{T}}_n} \arrow[d, "{[\![\text{🔒}_\mu]\!]^*\tau_n}"] \\
& {[\![\text{🔒}_{\mu\circ\nu}]\!]^*\mathcal{T}_o} \arrow[r, "{[\![\text{🔒}_\mu]\!]^*\mathbf{Mod}_\nu}"'] & {[\![\text{🔒}_\mu]\!]^*\mathcal{T}_n}
\end{tikzcd} \tag{1.8}$$

We have also used the fact that $(-)^*$ is functorial to contract the two locks into one. Moreover, we get that the unique mediating morphism is indeed $[\![\text{🔒}_\mu]\!]^*m$.

From this point onwards we will also work in the slice $\mathbf{PSh}(\mathcal{C}[m])/Z$, where $Z \triangleq [\![\text{🔒}_{\mu\circ\nu}]\!]^*\mathcal{T}_o$. In order to model the elimination rule we will ask for a left lifting structure *in the slice category*, of type

$$\vdash \mathbf{open}^\mu_\nu : [\![\text{🔒}_\mu]\!]^*m \pitchfork Z^*(\tau_m) \tag{1.9}$$

where both of these are considered as morphisms in the slice $\mathbf{PSh}(\mathcal{C}[m])/Z$, respectively of type

$$[\![\text{🔒}_\mu]\!]^*m : [\![\text{🔒}_{\mu\circ\nu}]\!]^*\tau_o \to [\![\text{🔒}_\mu]\!]^*h$$
$$Z^*(\tau_m) : Z^*(\widetilde{\mathcal{T}}_m) \to Z^*(\mathcal{T}_m)$$

Following Awodey [Awo18] we may calculate that this models the rule. We suppose its premises, and construct the diagram of Fig. 1.1. The right (both top and bottom) part of the diagram is just (1.8). The bottom composite is easily seen to correspond to the application of the introduction rule of $\langle \nu \mid - \rangle$ to the type $\Gamma.\text{🔒}_\mu.\text{🔒}_\nu \vdash A\ \mathsf{type}_1 \mathbin{@} o$, and hence to the type $\Gamma.\text{🔒}_\mu \vdash \langle \nu \mid A \rangle\ \mathsf{type}_1 \mathbin{@} n$. The outer bottom square is the natural model pullback square that defines the object $\Gamma.(\mu \mid \langle \nu \mid A \rangle)$, and we thus get a mediating morphism to $[\![\text{🔒}_\mu]\!]^*M$, and that the bottom-left square is also a pullback. The left (both

$$
\begin{tikzcd}
\mathbf{y}(\Gamma.(\mu \circ \nu \mid A)) \arrow[r, "\lfloor \mathbf{q} \rfloor"] \arrow[d, dashed, "\mathbf{y}(\mathbf{p}.\mathbf{mod}_\nu(\mathbf{q}))"] \arrow[dd, bend right, "\mathbf{y}(\mathbf{p})"'] \arrow[drr, bend left, "\lfloor \mathsf{mod}_\nu(\mathbf{q}) \rfloor"] & \llbracket \text{🔒}_{\mu\circ\nu} \rrbracket^* \widetilde{\mathcal{T}_o} \arrow[d, "\llbracket \text{🔒}_{\mu} \rrbracket^* m"] \arrow[dr, "\llbracket \text{🔒}_{\mu} \rrbracket^* \mathbf{mod}_\nu"] \arrow[dd, bend right, "\llbracket \text{🔒}_{\mu\circ\nu} \rrbracket^* \tau_o"'] & \\
\mathbf{y}(\Gamma.(\mu \mid \langle \nu \mid A \rangle)) \arrow[r, dashed] \arrow[d, "\mathbf{y}(\mathbf{p})"] \arrow[dr, phantom, "\lrcorner", very near start] \arrow[rr, bend left, "\lfloor \mathbf{q} \rfloor"] & \llbracket \text{🔒}_{\mu} \rrbracket^* M \arrow[r] \arrow[d, "\llbracket \text{🔒}_{\mu} \rrbracket^* h"] \arrow[dr, phantom, "\lrcorner", very near start] & \llbracket \text{🔒}_{\mu} \rrbracket^* \widetilde{\mathcal{T}_n} \arrow[d, "\llbracket \text{🔒}_{\mu} \rrbracket^* \tau_n"] \\
\mathbf{y}(\Gamma) \arrow[r, "\lfloor A \rfloor"] \arrow[rr, bend right, "\lfloor \langle \nu \mid A \rangle \rfloor"'] & \llbracket \text{🔒}_{\mu\circ\nu} \rrbracket^* \mathcal{T}_o \arrow[r, "\llbracket \text{🔒}_{\mu} \rrbracket^* \mathbf{Mod}_\nu"'] & \llbracket \text{🔒}_{\mu} \rrbracket^* \mathcal{T}_n
\end{tikzcd}
$$

Figure 1.1: Modelling the elimination rule

top and bottom) part of the diagram is the natural model pullback square that defines the object $\Gamma.(\mu \mid \langle \mu \circ \nu \mid A \rangle)$. We hence get a mediating morphism $\mathbf{p}.\mathbf{mod}_\nu(\mathbf{q}) : \Gamma.(\mu \mid \langle \mu \circ \nu \mid A \rangle) \to \Gamma.(\mu \mid \langle \nu \mid A \rangle)$. Finally, for the same reasons as the bottom composite, the top composite is easily seen to correspond to the term $\mathsf{mod}_\nu(\mathbf{q})$.

We write $\sum_Z : \mathbf{PSh}(\mathcal{C}[m])/Z \to \mathbf{PSh}(\mathcal{C}[m])$ for the usual domain projection functor, so that $\sum_Z \dashv Z^*$. Now, using the usual approach to slice categories—where the cartesian product $\times_Z$ is the pullback—we see from the diagram that

$$
\begin{aligned}
\sum_Z (\lfloor A \rfloor \times_Z \llbracket \text{🔒}_{\mu\circ\nu} \rrbracket^* \widetilde{\mathcal{T}_o}) &\cong \mathbf{y}(\Gamma.(\mu \circ \nu \mid A)) \\
\sum_Z (\lfloor A \rfloor \times_Z \llbracket \text{🔒}_{\mu} \rrbracket^* h) &\cong \mathbf{y}(\Gamma.(\mu \mid \langle \nu \mid A \rangle)) \\
\sum_Z (\mathsf{id}_{\lfloor A \rfloor} \times_Z \llbracket \text{🔒}_{\mu} \rrbracket^* m) &\cong \mathbf{y}(\mathbf{p}.\mathbf{mod}_\nu(\mathbf{q}))
\end{aligned}
\tag{1.10}
$$

Recall that we are trying to find a diagonal filler to the diagram

$$\boxed{\mathbf{PSh}(\mathcal{C}[m])}$$

$$\begin{tikzcd}
\mathbf{y}(\Gamma.(\mu \circ \nu \mid A)) \arrow[r, "\lfloor M_1 \rfloor"] \arrow[d, "\mathbf{y}(\mathbf{p}.\mathbf{mod}_\nu(\mathbf{q}))"'] & \widetilde{\mathcal{T}}_m \arrow[d, "\tau_m"] \\
\mathbf{y}(\Gamma.(\mu \mid \mathbf{Mod}_\nu(A))) \arrow[r, "\lfloor B \rfloor"'] \arrow[ur, dashed] & \mathcal{T}_m
\end{tikzcd} \tag{1.11}$$

We use the adjunction $\sum_Z \dashv Z^*$ to transpose this diagram, and we compose with the isomorphisms (1.10) to obtain the following diagram in $\mathbf{PSh}(\mathcal{C}[m])/Z$:

$$\begin{tikzcd}
\lfloor A \rfloor \times_Z [\![\text{🔒}_{\mu\circ\nu}]\!]^* \widetilde{\mathcal{T}}_o \arrow[r, "\widehat{\lfloor M_1 \rfloor}"] \arrow[d, "\mathsf{id} \times_Z [\![\text{🔒}_\mu]\!]^* m"'] & Z^*(\widetilde{\mathcal{T}}_m) \arrow[d, "Z^*(\tau_m)"] \\
\lfloor A \rfloor \times_Z [\![\text{🔒}_\mu]\!]^* h \arrow[r, "\widehat{\lfloor B \rfloor}"'] \arrow[ur, dashed, "\mathbf{open}^\mu_\nu" description] & Z^*(\mathcal{T}_m)
\end{tikzcd}$$

We can then use the lifting structure to prove a diagonal filler. Transposing this diagram back along the adjunction provides a filler for (1.11). The naturality of all these steps (composing isomorphisms, transposition, and lifting structure) ensure that the provided filler is natural.

*Boolean Structure*

A boolean structure is defined similarly to the structure for modal types. First, we require two operations:

$$\begin{tikzcd}
1 \arrow[r, bend left, "\mathbf{tt}"] \arrow[r, bend right, "\mathbf{ff}"'] \arrow[d, equal] & \widetilde{\mathcal{T}}_m \arrow[d, "\tau_m"] \\
1 \arrow[r, "\mathbf{Bool}"'] & \mathcal{T}_m
\end{tikzcd}$$

Modelling the elimination rule requires naturally given diagonal fillers for all squares

$$\begin{tikzcd}
1+1 \arrow[r] \arrow[d, "{[\mathbf{tt}, \mathbf{ff}]}"'] & \widetilde{\mathcal{T}}_m \arrow[d, "\tau_m"] \\
\tau_m^{-1}(\mathbf{Bool}) \arrow[r] \arrow[ur, dotted] & \mathcal{T}_m
\end{tikzcd}$$

where $\tau_m^{-1}(\mathbf{Bool})$ is the fibre of $\tau_m$ over $\mathbf{Bool}$, and the map $[\mathbf{tt}, \mathbf{ff}]$ is obtained as the cotuple of the maps obtained by factoring $\mathbf{tt}$ and $\mathbf{ff}$ through the fibre. Requiring a left lifting structure

$$\mathbf{if} : [\mathbf{tt}, \mathbf{ff}] \pitchfork \tau_m[-]$$

in the internal language then provides enough naturality to yield diagonal fillers for all squares

$$\begin{tikzcd}
\mathbf{y}(\Gamma) + \mathbf{y}(\Gamma) \arrow[r] \arrow[d, "{[\mathsf{id}.\mathbf{tt}, \mathsf{id}.\mathbf{ff}]}"'] & \widetilde{\mathcal{T}}_m \arrow[d, "\tau_m"] \\
\mathbf{y}(\Gamma.\mathbf{Bool}) \arrow[r] \arrow[ur, dotted] & \mathcal{T}_m
\end{tikzcd}$$

*Intensional Identity Structure*

We model the intensional identity type in exactly the same way as Awodey [Awo18]. First, we ask for a commuting square

$$\begin{tikzcd}
\sum_{A:\mathcal{T}_m} \tau_m^{-1}(A) \arrow[r, "\mathbf{refl}"] \arrow[d] & \widetilde{\mathcal{T}}_m \arrow[d, "\tau_m"] \\
\sum_{A:\mathcal{T}_m} \tau_m^{-1}(A) \times \tau_m^{-1}(A) \arrow[r, "\mathbf{Id}"'] & \mathcal{T}_m
\end{tikzcd}$$

which models the formation and introduction rules for $\mathbf{Id}$. Observe that

$$\sum_{A:\mathcal{T}_m} \tau_m^{-1}(A) \times \tau_m^{-1}(A) \cong \widetilde{\mathcal{T}}_m \times_{\mathcal{T}_m} \widetilde{\mathcal{T}}_m$$

For the path induction principle, we construct the pullback square

$$\begin{tikzcd}
\sum_{A:\mathcal{T}_m} \tau_m^{-1}(A) \arrow[dr, dashed, "i"] \arrow[drr, bend left, "\mathbf{refl}"] \arrow[ddr, bend right, "\sum_{A:\mathcal{T}_m} \delta"'] & & \\
& I \arrow[r] \arrow[d] \arrow[dr, phantom, "\lrcorner", very near start] & \widetilde{\mathcal{T}}_m \arrow[d, "\tau_m"] \\
& \sum_{A:\mathcal{T}_m} \tau_m^{-1}(A) \times \tau_m^{-1}(A) \arrow[r, "\mathbf{Id}"'] & \mathcal{T}_m
\end{tikzcd}$$

and require a left lifting structure

$$A : \mathcal{T}_m \vdash \mathbf{J} : i(A, -) \pitchfork \tau_m[-]$$

A detailed proof that this is sound for intensional identity types may be found in [Awo18, §2.4].

*The Universe of Small Types*

Until this point we have conveniently avoided discussing the difference between small and large types, i.e. whether our types are in $\mathsf{type}^0_m$ or $\mathsf{type}^1_m$. We were able to do that precisely because *terms can only inhabit large types*, so most of our judgments only mention large types. Of course, this is not the case when it comes to judgments pertaining to the universe.

First, our model comes equipped with a set of small types, viz. a presheaf $\mathcal{S}_m$ for each mode $m$ along with a natural transformation $\mathbf{lift} : \mathcal{S}_m \Rightarrow \mathcal{T}_m$. Moreover, we require that the formation rules for each type factor through $\mathcal{S}_m$. That is, we require a mediating morphism in each of the following diagrams.

*Pi*

$$\begin{tikzcd}
\sum_{A : \llbracket 🔒_\mu \rrbracket^* \mathcal{S}_n} \mathcal{S}_m^{\llbracket 🔒_\mu \rrbracket^* \tau_n^{-1}(\mathbf{lift}(A))} \arrow[r, dotted] \arrow[d] & \mathcal{S}_m \arrow[d, "\mathbf{lift}"] \\
\sum_{A : \llbracket 🔒_\mu \rrbracket^* \mathcal{T}_n} \mathcal{T}_m^{\llbracket 🔒_\mu \rrbracket^* \tau_n^{-1}(A)} \arrow[r, "\prod"'] & \mathcal{T}_m
\end{tikzcd}$$

*Sigma*

$$\begin{tikzcd}
\sum_{A : \mathcal{S}_m} \mathcal{S}_m^{\tau_m^{-1}(\mathbf{lift}(A))} \arrow[r, dotted] \arrow[d] & \mathcal{S}_m \arrow[d, "\mathbf{lift}"] \\
\sum_{A : \mathcal{T}_m} \mathcal{T}_m^{\tau_m^{-1}(A)} \arrow[r, "\sum"'] & \mathcal{T}_m
\end{tikzcd}$$

$\langle \mu \mid - \rangle$

$$\begin{tikzcd}
\llbracket 🔒_\mu \rrbracket^* \mathcal{S}_n \arrow[r, dotted] \arrow[d] & \mathcal{S}_m \arrow[d, "\mathbf{lift}"] \\
\llbracket 🔒_\mu \rrbracket^* \mathcal{T}_n \arrow[r, "\mathbf{Mod}_\mu"'] & \mathcal{T}_m
\end{tikzcd}$$

*Intensional Identity*

$$\begin{tikzcd}
\sum_{A : \mathcal{S}_m} \tau_m^{-1}(\mathbf{lift}(A)) \times \tau_m^{-1}(\mathbf{lift}(A)) \arrow[r, dotted] \arrow[d] & \mathcal{S}_m \arrow[d, "\mathbf{lift}"] \\
\sum_{A : \mathcal{T}_m} \tau_m^{-1}(A) \times \tau_m^{-1}(A) \arrow[r, "\mathbf{Id}"'] & \mathcal{T}_m
\end{tikzcd}$$

*Bool*

$$\begin{tikzcd} 1 \arrow[r, dotted] \arrow[d] & \mathcal{S}_m \arrow[d, "\mathbf{lift}"] \\ 1 \arrow[r, "\mathbf{Bool}"'] & \mathcal{T}_m \end{tikzcd}$$

The existence of these factorisations implies that type formation is closed under the set of small types. Finally, the universe of small types is interpreted by distinguished morphism $1 \xrightarrow{\mathbf{Uni}} \mathcal{T}_m$ for each $m \in \mathcal{M}$, which is such that

$$\tau_m^{-1}(\mathbf{Uni}) \cong \mathcal{S}_m$$

*The Full Definition*

Collecting our work, we have that

**Definition 1.4.8.** A model of MTT over $\mathcal{M}$ consists of

- a context structure for $\mathcal{M}$ (Definition 1.4.1), and a
- a modal natural model on that context structure (Definition 1.4.4)

such that the modal natural model supports

- dependent product types
- dependent sum types (at each mode)
- intensional identity types (at each mode)
- modal types
- a boolean type (at each mode), and
- a universe of small types

### 1.4.2 *Models from Dependent Right Adjoints*

In Section 1.4.1 we showed how to decompose the algebraic notion of model of MTT into a more modular and attractive presentation using the language of Awodey's [Awo18] natural models. However, there is already a general notion of dependent modality in modal type theory, namely that of a *dependent right adjoint (DRA)* [Bir+20]. In this section we use the language of natural models to generalise the definition of DRA to a multimode setting. Furthermore, we construct a model from multimode DRAs between models of type theory that support a similar array of types ($\sum$, $\prod$, $\mathsf{Id}$, $\mathbb{B}$). This construction demonstrates that DRAs constitute an even stronger notion of modality.

*Dependent Right Adjoints in Natural Models*

A dependent right adjoint is an adaption of the notion of adjunction to the dependent setting, as it has an action on both terms and types. In particular, given a pair of natural models $(\mathcal{C}, \tau_{\mathcal{C}})$ and $(\mathcal{D}, \tau_{\mathcal{D}})$, the data of a DRA comprises a functor $L : \mathcal{D} \to \mathcal{C}$ between the underlying context categories, as well as a pullback diagram of the following shape in $\mathbf{PSh}(\mathcal{D})$:

$$\begin{tikzcd}
L^*\widetilde{\mathcal{T}}_{\mathcal{C}} \arrow[r, "r"] \arrow[d, "L^*\tau_{\mathcal{C}}"'] \arrow[dr, phantom, "\lrcorner", very near start] & \widetilde{\mathcal{T}}_{\mathcal{D}} \arrow[d, "\tau_{\mathcal{D}}"] \\
L^*\mathcal{T}_{\mathcal{C}} \arrow[r, "R"'] & \mathcal{T}_{\mathcal{D}}
\end{tikzcd}$$

Here $R$ is the action on types, and $r$ is its action on terms. It is evident that the pullback square forces $(R, r)$ to behave as a DRA in the sense of Birkedal et al. [Bir+20], as it intuitively defines for each term $\Gamma \vdash M : R(A)$ a unique term $L(\Gamma) \vdash N : A$ such that $\Gamma \vdash M = r(N) : R(A)$. We must also ask that the DRA preserve size if we wish for a small modal type; in that case, we also require a $R'$ such that

$$\begin{tikzcd}
L^*\mathcal{S}_{\mathcal{C}} \arrow[r, dotted, "R'"] \arrow[d] & \mathcal{S}_{\mathcal{D}} \arrow[d, "\mathbf{lift}"] \\
L^*\mathcal{T}_{\mathcal{C}} \arrow[r, "R"'] & \mathcal{T}_{\mathcal{D}}
\end{tikzcd}$$

**Remark 1.4.9.** This definition is slightly more general than the one presented in Birkedal et al. [Bir+20], which forced the DRA to be an *endoadjunction* (that is, $L$ was an endofunctor and there was only one natural model). There are no technical obstacles to generalizing any of the results in the paper to this setting, though the syntax must be made more general (permitting two modes, instead of just one). $\lhd$

**Remark 1.4.10.** With a proper definition of a *morphism* of models, it can be shown that an adjunction between categories of contexts gives rise to a dependent right adjoint when the right adjoint is part of a morphism of models [Nuy18a]. The converse is not in general true: a dependent right adjoint need not have any action on the full category of contexts. If, however, the category of contexts is entirely generated by $\cdot$ and $\Gamma.A$ (if the model is *democratic*), the converse is true [Bir+20]. $\lhd$

*When $\mathbf{Mod}_\mu$ is a DRA*

**Theorem 1.4.11.** *Suppose that we have an collection $(\mathcal{C}[m], \widetilde{\mathcal{T}}_m \xrightarrow{\tau_m} \mathcal{T}_m)_{m \in \mathcal{M}}$ of models of MLTT, indexed by modes $m \in \mathcal{M}$. Suppose further that we have functorial choice of size-preserving dependent right adjoints, $([\![\text{🔒}_\mu]\!], \mathbf{Mod}_\mu, \mathbf{mod}_\mu)$, from $(\mathcal{C}[m], \widetilde{\mathcal{T}}_m \xrightarrow{\tau_m} \mathcal{T}_m)$ to $(\mathcal{C}[n], \widetilde{\mathcal{T}}_n \xrightarrow{\tau_n} \mathcal{T}_n)$ for each modality $\mu \in \mathrm{Hom}_{\mathcal{M}}(m, n)$. Finally, assume that there exists a functorial choice of natural transformations $[\![\text{🔑}^\alpha]\!] : [\![\text{🔒}_\nu]\!] \Rightarrow [\![\text{🔒}_\mu]\!]$ for each $\alpha : \mu \Rightarrow \nu$. Then we can assemble these data into a model of MTT where the type theory at each mode $m$ is interpreted by $(\mathcal{C}[m], \widetilde{\mathcal{T}}_m \xrightarrow{\tau_m} \mathcal{T}_m)$.*

*Proof.* First, define the 2-functor $\mathcal{M}^{\mathsf{coop}} \to \mathbf{Cat}$ by $m \mapsto \mathcal{C}[m]$, $\mu \mapsto [\![\text{🔒}_\mu]\!]$, and $\alpha \mapsto [\![\text{🔑}^\alpha]\!]$. We need to show how to define the structure necessary for interpreting context extension and type formers.

*Modal Context Extension*

Suppose we have an arrow $\mathbf{y}(\Gamma) \xrightarrow{\lfloor A \rfloor} [\![\text{🔒}_\mu]\!]^*\mathcal{T}_n$. We would like for each $n \in \mathcal{M}$ a pullback square

$$\begin{tikzcd}
\mathbf{y}(\Gamma') \arrow[r, "\lfloor q' \rfloor"] \arrow[d, "\mathbf{y}(p')"'] \arrow[dr, phantom, "\lrcorner", very near start] & {[\![\text{🔒}_\mu]\!]^*\widetilde{\mathcal{T}}_n} \arrow[d, "{[\![\text{🔒}_\mu]\!]^*\tau_n}"] \\
\mathbf{y}(\Gamma) \arrow[r, "\lfloor A \rfloor"'] & {[\![\text{🔒}_\mu]\!]^*\mathcal{T}_n}
\end{tikzcd}$$

Write $\lfloor \mathbf{Mod}_\mu(A) \rfloor \triangleq \mathbf{Mod}_\mu \circ \lfloor A \rfloor$, and form the natural model pullback square for $\Gamma' \triangleq \Gamma.\mathbf{Mod}_\mu(A)$. Pasting this with the DRA pullback square for $\mathbf{Mod}_\mu$ forms the following diagram:

$$\begin{tikzcd}
\mathbf{y}(\Gamma.\mathbf{Mod}_\mu(A)) \arrow[r, dotted, "\widehat{\lfloor \mathbf{q} \rfloor}"] \arrow[d, "\mathbf{y}(\mathbf{p})"'] \arrow[dr, phantom, "\lrcorner", very near start] \arrow[rr, bend left] & {[\![\text{🔒}_\mu]\!]^*\widetilde{\mathcal{T}}_n} \arrow[r, "\mathbf{mod}_\mu"] \arrow[d, "{[\![\text{🔒}_\mu]\!]^*\tau_n}"] \arrow[dr, phantom, "\lrcorner", very near start] & \widetilde{\mathcal{T}}_m \arrow[d, "\tau_m"] \\
\mathbf{y}(\Gamma) \arrow[r, "\lfloor A \rfloor"'] & {[\![\text{🔒}_\mu]\!]^*\mathcal{T}_n} \arrow[r, "\mathbf{Mod}_\mu"'] & \mathcal{T}_m
\end{tikzcd} \tag{1.12}$$

As the outer square commutes, we can fill in the dotted arrow. By the pullback lemma, because both the outer square and the rightmost square are pullbacks, so is the leftmost. Therefore, letting $\Gamma.(\mu \mid A) \triangleq \Gamma.\mathbf{Mod}_\mu(A)$ completes the proof that $(\tau_m)_{m\in\mathcal{M}}$ is a modal natural model.

*$\sum$, Boolean, Intensional Identity, and Small Type and Universe Structures*

The structures for $\sum$, $\mathbf{Bool}$, $\mathbf{Id}$, and $\mathbf{Uni}$ are *mode-local*, so we may simply reuse the equivalent data given for each natural model $\widetilde{\mathcal{T}}_m \xrightarrow{\tau_m} \mathcal{T}_m$.

*Modal Types*

This is the heart of the proof. First, we need a commuting square

$$\begin{tikzcd}
{[\![\text{🔒}_\mu]\!]^*\widetilde{\mathcal{T}}_n} \arrow[r, "\mathbf{mod}_\mu"] \arrow[d] \arrow[dr, phantom, "\lrcorner", very near start] & \widetilde{\mathcal{T}}_m \arrow[d, "\tau_m"] \\
{[\![\text{🔒}_\mu]\!]^*\mathcal{T}_n} \arrow[r, "\mathbf{Mod}_\mu"'] & \mathcal{T}_m
\end{tikzcd} \tag{1.13}$$

Such a square is given as part of a DRA by definition, and is in fact a pullback!

To model the elimination rule, recall that definition of the object $M$ used in the lifting condition:

$$\begin{tikzcd}
\llbracket \text{🔒}_{\mu\circ\nu} \rrbracket^* \widetilde{\mathcal{T}}_o \arrow[dr, dashed, "\llbracket \text{🔒}_\mu \rrbracket^* m"] \arrow[drr, bend left, "\llbracket \text{🔒}_\mu \rrbracket^* \mathbf{mod}_\nu"] \arrow[ddr, bend right, "\llbracket \text{🔒}_{\mu\circ\nu} \rrbracket^* \tau_o"'] & & \\
& \llbracket \text{🔒}_\mu \rrbracket^* M \arrow[r] \arrow[d] \arrow[dr, phantom, "\lrcorner", very near start] & \llbracket \text{🔒}_\mu \rrbracket^* \widetilde{\mathcal{T}}_n \arrow[d, "\llbracket \text{🔒}_\mu \rrbracket^* \tau_n"] \\
& \llbracket \text{🔒}_{\mu\circ\nu} \rrbracket^* \mathcal{T}_o \arrow[r, "\llbracket \text{🔒}_\mu \rrbracket^* \mathbf{Mod}_\nu"'] & \llbracket \text{🔒}_\mu \rrbracket^* \mathcal{T}_n
\end{tikzcd}$$

As $\llbracket \text{🔒}_\mu \rrbracket^*$ preserves pullbacks the outer square is a pullback too, and so $\llbracket \text{🔒}_\mu \rrbracket^* m$ must be an isomorphism. The elimination rule for $\mathbf{Mod}_\mu(A)$ requires us to construct a left-lifting structure:

$$\vdash \mathbf{open}^\mu_\nu : (\llbracket \text{🔒}_\mu \rrbracket^* m) \pitchfork (\llbracket \text{🔒}_{\mu\circ\nu} \rrbracket^* \mathcal{T}_o)^* (\tau_m[-])$$

Using the inverse of $\llbracket \text{🔒}_\mu \rrbracket^* m$ we can construct this by

$$\mathbf{open}^\mu_\nu \triangleq \lambda C.\ \lambda c.\ c \circ \llbracket \text{🔒}_\mu \rrbracket^* (m^{-1})$$

### $\prod$ *Structure*

Equipping each $\widetilde{\mathcal{T}}_m \xrightarrow{\tau_m} \mathcal{T}_m$ with a modal $\prod$ structure is relatively straightforward to do in the internal language; intuitively, the reason is the isomorphism

$$(\llbracket \text{🔒}_\mu \rrbracket^* \tau_n)^{-1}(A) \cong \tau_m^{-1}(\mathbf{Mod}_\mu A)$$

which is derived from the fact $\Gamma.(\mu \mid A) \triangleq \Gamma.\mathbf{Mod}_\mu A$ (where the first dot is the defined context extension, and the second dot is given by the natural model). However, we can also prove it in a more abstract way: we paste together the two pullback squares

$$\begin{tikzcd}
\mathbf{P}_{\llbracket \text{🔒}_\mu \rrbracket^* \tau_n}(\widetilde{\mathcal{T}}_m) \arrow[r, "\phi_{\widetilde{\mathcal{T}}_m}"] \arrow[d, "\mathbf{P}_{\llbracket \text{🔒}_\mu \rrbracket^* \tau_n}(\tau_m)"'] \arrow[dr, phantom, "\lrcorner", very near start] & \mathbf{P}_{\tau_m}(\widetilde{\mathcal{T}}_m) \arrow[r, "\mathbf{lam}"] \arrow[d, "\mathbf{P}_{\tau_m}(\tau_m)"'] \arrow[dr, phantom, "\lrcorner", very near start] & \widetilde{\mathcal{T}}_m \arrow[d, "\tau_m"] \\
\mathbf{P}_{\llbracket \text{🔒}_\mu \rrbracket^* \tau_n}(\mathcal{T}_m) \arrow[r, "\phi_{\mathcal{T}_m}"'] & \mathbf{P}_{\tau_m}(\mathcal{T}_m) \arrow[r, "\prod"'] & \mathcal{T}_m
\end{tikzcd}$$

The square on the right is the pullback that interprets $\prod$ in the natural model $\tau_m$. The square on the left is a naturality square of the natural transformation

$$\phi : \mathbf{P}_{\llbracket \text{🔒}_\mu \rrbracket^* \tau_n}(-) \Rightarrow \mathbf{P}_{\tau_m}(-)$$

which exists because the pullback square (1.13) defines a morphism of polynomials. Moreover, the naturality squares of $\phi$ are cartesian: see the thesis of Newstead [New18, §§1.2.16–1.2.18].

$\square$

**Corollary 1.4.12.** *MTT is consistent (there is no term* $\cdot \vdash M : \mathsf{Id}_{\mathbb{B}}(\mathsf{tt}, \mathsf{ff}) \mathbin{@} m$*) for any mode theory* $\mathcal{M}$*.*

*Proof.* Suppose that we have a model of MLTT with one universe in some category $\mathcal{C}$. We can construct a functor $\mathcal{M}^{\mathsf{coop}} \to \mathbf{Cat}$ by sending every mode to $\mathcal{C}$, and every modality and 2-cell to the identity functor and natural transformation respectively. It is clear that this is stricty 2-functorial, and that each identity functor is a DRA. Hence, Theorem 1.4.11 tells us that there is a model of MTT with each mode being interpreted by $\mathcal{C}$. Therefore, if a term $M : \mathsf{Id}_{\mathbb{B}}(\mathsf{tt}, \mathsf{ff})$ was definable in MTT, it would also be possible to construct in every model of MLTT, and so would be definable in MLTT itself. But it is well-known that MLTT is consistent: see Coquand [Coq18] for a particularly short proof. □

### 1.4.3 Morphisms of Models

Returning to the models induced by the generalized algebraic syntax, we observe that not only does a GAT induce a collection of models, it also determines a notion of morphism between models which forms a category. Though traditionally neglected, homomorphisms of models are of fundamental importance for metatheoretic proofs, and will be used for our proof of canonicity.

Recent notions of CwF morphisms [CD14; Bir+20] are relatively weak, in that they preserve the CwF structure up to isomorphism. In contrast, our metatheoretic proofs require that we revert to a purely algebraic notion of morphism that preserves all structure on-the-nose, as originally introduced by Dybjer [Dyb96], and recently adapted to natural models by Newstead [New18, §2.3]. While we are ready to believe that one may construct a biequivalence or biadjunction relating these to more semantically natural morphisms [Uem19], we do not pursue this matter further.

**Definition 1.4.13.** A morphism $(\mathcal{C}, \widetilde{\mathcal{T}}_c \xrightarrow{\tau_c} \mathcal{T}_c) \to (\mathcal{D}, \widetilde{\mathcal{T}}_d \xrightarrow{\tau_d} \mathcal{T}_d)$ of natural models comprises a functor $F : \mathcal{C} \to \mathcal{D}$ as well as a commuting square

$$\begin{tikzcd} \widetilde{\mathcal{T}}_c \arrow[r, "\widetilde{\varphi}"] \arrow[d, "\tau_c"'] & F^*\widetilde{\mathcal{T}}_d \arrow[d, "F^*\tau_d"] \\ \mathcal{T}_c \arrow[r, "\varphi"'] & F^*\mathcal{T}_d \end{tikzcd} \tag{1.14}$$

such that $F(\mathbf{1}) = \mathbf{1}$ and the canonical morphism $F(\Gamma.A) \to F\Gamma.\varphi(A)$ is an identity.

The type $\varphi(A)$ in the last line is defined as follows. Given $\lfloor A \rfloor : \mathbf{y}(\Gamma) \Rightarrow \mathcal{T}_c$ we let

$$k \triangleq \mathbf{y}(\Gamma) \xrightarrow{\lfloor A \rfloor} \mathcal{T}_c \xrightarrow{\varphi} F^*\widetilde{\mathcal{T}}_d$$

We then have by Yoneda a natural isomorphism

$$\mathrm{Hom}_{\mathbf{PSh}(\mathcal{C})}(\mathbf{y}(\Gamma), F^*\mathcal{T}_d) \cong F^*\mathcal{T}_d(\Gamma) = \mathcal{T}_d(F\Gamma) \cong \mathrm{Hom}_{\mathbf{PSh}(\mathcal{D})}(\mathbf{y}(F\Gamma), \mathcal{T}_d) \tag{1.15}$$

We define $\lfloor \phi(A) \rfloor : \mathbf{y}(F\Gamma) \Rightarrow \mathcal{T}_d$ to be $k$ transported under this natural isomorphism. We also define

$$\lfloor M \rfloor : \mathbf{y}(\Gamma) \Rightarrow \mathcal{T}_c \quad \longmapsto \quad \lfloor \widetilde{\varphi}(M) \rfloor : \mathbf{y}(F\Gamma) \Rightarrow \mathcal{T}_d$$

which maps a term $\Gamma \vdash M : A$ to a term $F\Gamma \vdash \widetilde{\varphi}(M) : \varphi(A)$ in a similar manner.

Returning to the last condition in the definition, we may now form the diagram

$$\begin{tikzcd}
\mathbf{y}(F(\Gamma.A)) \arrow[dr, dashed] \arrow[drr, bend left, "\lfloor\widetilde{\varphi}(\mathbf{q})\rfloor"] \arrow[ddr, bend right, "\mathbf{y}(F\mathbf{p})"'] & & \\
& \mathbf{y}(F\Gamma.\varphi(A)) \arrow[r, "\lfloor\mathbf{q}\rfloor"] \arrow[d, "\mathbf{y}(\mathbf{p})"'] \arrow[dr, phantom, "\lrcorner", very near start] & \widetilde{\mathcal{T}}_d \arrow[d, "\tau_d"] \\
& \mathbf{y}(F\Gamma) \arrow[r, "\lfloor\varphi(A)\rfloor"'] & \mathcal{T}_d
\end{tikzcd}$$

where the outer square is the diagram composed by pasting together the context extension diagram for $\Gamma.A$ and (1.14), followed by transposing along the natural isomorphism (1.15). We then ask that the unique induced arrow be the identity.

We can lift these natural transformations to the formation data of the connectives (making special use of the final equality for the polynomial functors). For instance, we can define a morphism

$$\mathbf{P}_{\tau_c}(\mathcal{T}_c) \xrightarrow{(\varphi,\widetilde{\varphi})} \mathbf{P}_{F^*\tau_d}(F^*\mathcal{T}_d) \triangleq \mathbf{P}_{\tau_c}(\mathcal{T}_c) \to \mathbf{P}_{F^*\tau_d}(\mathcal{T}_c) \xrightarrow{\mathbf{P}_{F^*\tau_d}(\varphi)} \mathbf{P}_{F^*\tau_d}(F^*\mathcal{T}_d)$$

The first component comes from a natural transformation $\mathbf{P}_{\tau_c}(-) \Rightarrow \mathbf{P}_{F^*\tau_d}(-)$, which exists because (1.14) not only commutes, but is a pullback square. That is a nontrivial fact proven laboriously by Newstead [New18, §§2.3.14]. A more conceptual proof is given by Uemura [Uem19, p. 3.14] in the language of discrete fibrations.

We then require that all our connectives, $\prod$, $\sum$, **refl**, strictly commute with these morphisms. Finally, we can extend this to a model of MTT by requiring not just a functor, but a natural transformation from $\mathcal{C} \to \mathcal{D}$, where $\mathcal{C}, \mathcal{D} : \mathcal{M}^{\mathsf{coop}} \to \mathbf{Cat}$ satisfying the obvious generalizations of the conditions written above. Specifying this formally:

**Definition 1.4.14.** A morphism between two models of MTT, $\mathcal{C}, \mathcal{D}$, is given by a 2-natural transformation: $F : \mathcal{C} \to \mathcal{D}$. Moreover, we require a choice of commuting squares:

$$\begin{tikzcd}
\widetilde{\mathcal{U}}_{\mathcal{C}[m]} \arrow[r, "\widetilde{\varphi}_m"] \arrow[d, "\tau_{\mathcal{C}[m]}"'] & F^*_m\widetilde{\mathcal{U}}_{\mathcal{D}[m]} \arrow[d, "F^*\tau_{\mathcal{D}[m]}"] \\
\mathcal{U}_{\mathcal{C}[m]} \arrow[r, "\varphi_m"'] & F^*\mathcal{U}_{\mathcal{D}[m]}
\end{tikzcd}$$

Moreover, we require that $(\varphi, \widetilde{\varphi})$ strictly commutes with all operations.

$$
\begin{aligned}
F_m(\Gamma.(\mu \mid A)) &= F_m(\Gamma).(\mu \mid \varphi(A)) & & \\
\textstyle\prod \circ (\varphi, \varphi) &= \varphi \circ \textstyle\prod & \mathbf{lam} \circ (\varphi, \widetilde{\varphi}) &= \widetilde{\varphi} \circ \mathbf{lam} \\
\textstyle\sum \circ (\varphi, \varphi) &= \varphi \circ \textstyle\sum & \mathbf{pair} \circ (\varphi, \widetilde{\varphi}) &= \widetilde{\varphi} \circ \mathbf{pair} \\
\mathbf{Mod}_\mu \circ [\![\text{🔒}_\mu]\!]^* \varphi &= \varphi \circ \mathbf{Mod}_\mu & \mathbf{mod}_\mu \circ [\![\text{🔒}_\mu]\!]^* \widetilde{\varphi} &= \widetilde{\varphi} \circ \mathbf{mod}_\mu \\
& & \mathbf{open}^\nu_\mu \circ (\widetilde{\varphi}, [\![\text{🔒}_\mu]\!]^* \widetilde{\varphi}) &= \widetilde{\varphi} \circ \mathbf{open}^\nu_\mu \\
\mathbf{Bool} &= \varphi \circ \mathbf{Bool} & \mathbf{tt} = \widetilde{\varphi} \circ \mathbf{tt} \quad \mathbf{ff} &= \widetilde{\varphi} \circ \mathbf{ff} \\
& & \mathbf{if} \circ (\widetilde{\varphi}, \widetilde{\varphi}, \widetilde{\varphi}) &= \widetilde{\varphi} \circ \mathbf{if} \\
\mathbf{Id} \circ (\varphi, \widetilde{\varphi}, \widetilde{\varphi}) &= \varphi \circ \mathbf{Id} & \mathbf{refl} \circ \widetilde{\varphi} &= \widetilde{\varphi} \circ \mathbf{refl} \\
& & \mathbf{J} \circ (\widetilde{\varphi}, \widetilde{\varphi}) &= \widetilde{\varphi} \circ \mathbf{J}
\end{aligned}
$$

**Remark 1.4.15** (The Initiality of Syntax)**.** Under this definition of homomorphism, we immediately have an initial model [Car78; KKA19]. We will *define* this model to be our syntax and designate it $(\mathbb{S}[m])_{m \in \mathcal{M}}$. ◁

## 1.5 Canonicity

At this point we have developed a rich theory of the syntax and semantics of MTT. We can instantiate the syntax with different mode theories in order to obtain various modal calculi. However, we would like to show that—irrespective of the mode theory of choice—the resulting syntax is well-behaved. In this section we establish one of the basic properties which measures this, i.e. canonicity.

**Proposition 1.5.1** (Canonicity). *If* $\vdash M : \mathbb{B} @ m$*, then either* $\vdash M = \mathsf{tt} : \mathbb{B} @ m$ *or* $\vdash M = \mathsf{ff} : \mathbb{B} @ m$*.*

Traditionally this kind of result would be established by producing a rewriting system along with a lengthy PER model construction. Instead, we will opt for a proof given by constructing a *glued* model of MTT [KHS19]. Our gluing proof proceeds by defining a model of MTT in which contexts and types are pairs of a syntactic context, and a proof-relevant predicate on its elements.

In order to simplify the proof, we will assume that locks preserve the empty context. This amounts to the equation

$$\cdot.\text{🔒}_{\mu} = \cdot \ \mathsf{ctx} @ m$$

for each $\mu : \mathrm{Hom}_{\mathcal{M}}(m, n)$. Together with the universal property of the terminal context, this implies

$$\cdot.\text{🔒}_{\mu} \vdash \text{🔑}^{\alpha}_{\cdot} = \mathsf{id} = \cdot : \cdot.\text{🔒}_{\nu} @ m$$

Requiring this equation unfortunately limits our class of models to those where the left adjoint strictly preserves the terminal product.

**Remark 1.5.2.** In what follows we will assume the existence of two Grothendieck universes $\mathcal{V}' \subset \mathcal{V} :$ **Set**. We could make do with just one, but this would introduce some contortions. However, that is both unnecessary and uninteresting. We will assume that the sets of contexts, substitutions, types, and terms from the syntactic model are $\mathcal{V}'$-small. $\lhd$

### 1.5.1 Defining the Glued Model

*The Glued Context Categories*

**Definition 1.5.3** (Glued Contexts). A glued context $\Gamma$ at mode $m$ consists of a context $\Gamma^{\lhd} \in \mathsf{ctx}_m$, a predicate $\Gamma^{\blacktriangleright} \in \mathcal{V}$, and a function

$$\phi_{\Gamma} : \Gamma^{\blacktriangleright} \to \mathsf{sb}_m(\cdot, \Gamma^{\lhd})$$

**Remark 1.5.4.** We will henceforth use the metavariable $\Gamma$ to range over *glued contexts*, and explicitly specify when we intend it to range over syntactic contexts. $\lhd$

**Definition 1.5.5** (Glued Substitutions). A glued substitution from $\Delta$ to $\Gamma$ at mode $m$ is a pair of a substitution $\gamma^{\lhd} \in \mathsf{sb}_m(\Delta^{\lhd}, \Gamma^{\lhd})$ and a function $\gamma^{\blacktriangleright} : \Delta^{\blacktriangleright} \to \Gamma^{\blacktriangleright}$ such that

$$\forall x \in \Delta^{\blacktriangleright}.\ \phi_{\Gamma}(\gamma^{\blacktriangleright}(x)) = \gamma^{\lhd} \circ \phi_{\Delta}(x)$$

Glued contexts and glued substitutions form a category, viz. the comma category

$$\mathcal{C}[m] \triangleq (1_{\mathcal{V}} \downarrow \mathsf{sb}_m(\cdot, -))$$

which we take as the category of contexts at mode $m$. Next, we define a 2-functor from $\mathcal{M}$ sending each $m$ to $\mathcal{C}[m]$. We define for each $\mu : \mathrm{Hom}_{\mathcal{M}}(m, n)$ a functor $[\![\text{🔒}_{\mu}]\!] : \mathcal{C}[n] \to \mathcal{C}[m]$ as follows. Suppose we are given the following arrow in $\mathcal{C}[n]$:

$$\begin{tikzcd}
\Delta^{\blacktriangleright} \arrow[rr, "\gamma^{\blacktriangleright}"] \arrow[d, "\phi_{\Delta}"] & & \Gamma^{\blacktriangleright} \arrow[d, "\phi_{\Gamma}"] \\
\mathsf{sb}_n(\cdot, \Delta^{\lhd}) \arrow[rr, "\gamma^{\lhd} \circ -"'] & & \mathsf{sb}_n(\cdot, \Gamma^{\lhd})
\end{tikzcd}$$

We will send it to the following arrow in $\mathcal{C}[m]$:

$$\begin{tikzcd}
\Delta^{\blacktriangleright} \arrow[rr, "\gamma^{\blacktriangleright}"] \arrow[d, "\phi_{\Delta.\text{🔒}_{\mu}}"] & & \Gamma^{\blacktriangleright} \arrow[d, "\phi_{\Gamma.\text{🔒}_{\mu}}"] \\
\mathsf{sb}_m(\cdot, \Delta^{\lhd}.\text{🔒}_{\mu}) \arrow[rr, "\gamma^{\lhd}.\text{🔒}_{\mu} \circ -"'] & & \mathsf{sb}_m(\cdot, \Gamma^{\lhd}.\text{🔒}_{\mu})
\end{tikzcd}$$

where the function $\phi_{\Delta.\text{🔒}_{\mu}}$ is defined by

$$\phi_{\Delta.\text{🔒}_{\mu}}(x) \triangleq \phi_{\Delta}(x).\text{🔒}_{\mu} \;:\; \cdot \to \Delta^{\lhd}.\text{🔒}_{\mu}$$

Notice that the equation $\cdot.\text{🔒}_{\mu} = \cdot$ is necessary to ensure that this definition is well-typed. The diagram commutes, as locks act functorially on substitutions, and this assignment is functorial for the same reason. It is also functorial in $\mu$, because $\Gamma.\text{🔒}_{\mu}.\text{🔒}_{\nu} = \Gamma.\text{🔒}_{\mu\circ\nu}$, and $\Gamma.\text{🔒}_{1} = \Gamma$.

Next, we can define a 2-cell $[\![\text{🔒}_{\mu}]\!] \Rightarrow [\![\text{🔒}_{\nu}]\!]$ whenever $\alpha : \nu \Rightarrow \mu$. The component at $(\Gamma^{\blacktriangleright}, \Gamma^{\lhd}, \phi_{\Gamma})$ is

$$\begin{tikzcd}
\Gamma^{\blacktriangleright} \arrow[rr, equal] \arrow[d, "\phi_{\Gamma.\text{🔒}_{\mu}}"] & & \Gamma^{\blacktriangleright} \arrow[d, "\phi_{\Gamma.\text{🔒}_{\mu}} < \nu >"] \\
\mathsf{sb}_m(\cdot, \Gamma^{\lhd}.\text{🔒}_{\mu}) \arrow[rr, "\text{🔑}^{\alpha}_{\Gamma^{\lhd}} \circ -"'] & & \mathsf{sb}_m(\cdot, \Gamma^{\lhd}.\text{🔒}_{\nu})
\end{tikzcd}$$

This diagram commutes because of the equation $\text{🔑}^{\alpha}_{\cdot} = \mathsf{id}$, so it defines a morphism in the comma category. Naturality of both this component and of the assignment of this cell to $\alpha : \nu \Rightarrow \mu$ follows from the numerous naturality equations pertaining to keys and their composition.

This completes the definition of a strict 2-functor $\mathcal{M}^{\mathsf{coop}} \to \mathbf{Cat}_1$, as discussed in Section 1.4.1.

### *The Glued Natural Model Structure*

Next we must define the modal natural model structure for each category of contexts.

**Remark 1.5.6.** For the rest of this section we will freely use type-theoretic notation, viewing a proof-relevant predicate $\Gamma^{\blacktriangleright} \to \mathsf{sb}_m(\cdot, \Gamma^{\lhd})$ as a family fibred over $\mathsf{sb}_m(\cdot, \Gamma^{\lhd})$, i.e. a map $\mathsf{sb}_m(\cdot, \Gamma^{\lhd}) \to \mathcal{V}$.

As in the definition of the gluing category, we will follow the convention that symbols annotated with $(-)^{\blacktriangleright}$ correspond to proof-relevant constructions—i.e. members of the predicate, or maps between predicates—whereas symbols annotated with $(-)^{\lhd}$ correspond to syntactic constructions, e.g. terms, contexts, substitutions. In particular, $\gamma^{\blacktriangleright}$ will no longer refer to a fibred map between proof-relevant predicates, as it meant in the preceding discussion. We will also often use 'generalized element' notation: lower-case Greek letters will correspond to generalized elements of families denoted by upper-case Greek letters.

In other words, when $\gamma^{\blacktriangleright} \in \Gamma^{\blacktriangleright}$ and $\phi_\Gamma(\gamma^{\blacktriangleright}) = \gamma^{\lhd} : \cdot \to \Gamma^{\lhd}$, we will abusively write $\gamma^{\blacktriangleright} : \Gamma^{\blacktriangleright}(\gamma^{\lhd})$. That is, we will view $\gamma^{\blacktriangleright}$ as living in the fibre of $\phi_\Gamma$ over $\gamma^{\lhd}$. This amounts to considering $\gamma^{\blacktriangleright}$ as a *proof* that the predicate $\Gamma^{\blacktriangleright}$ holds at the substitution $\gamma^{\lhd}$. Observe that if $\gamma^{\blacktriangleright} : \Gamma.\text{🔒}_\mu^{\blacktriangleright}(\gamma^{\lhd})$ then $\gamma^{\lhd}$ must be of the form $\theta^{\lhd}.\text{🔒}_\mu$ for some $\theta^{\lhd} : \cdot \to \Gamma^{\lhd}$ with $\gamma^{\blacktriangleright} : \Gamma^{\blacktriangleright}(\theta^{\lhd})$. $\lhd$

We begin by defining the following presheaves over $\mathcal{C}[m]$:

$$
\begin{aligned}
&\mathcal{T}_m(\Gamma) \triangleq \{\\
&\quad A^{\lhd} \in \mathsf{type}^1_m(\Gamma^{\lhd});\\
&\quad A^{\blacktriangleright} : (\gamma^{\lhd} : \mathsf{sb}_m(\cdot, \Gamma^{\lhd})) \to (\gamma^{\blacktriangleright} : \Gamma^{\blacktriangleright}(\gamma^{\lhd})) \to \mathsf{tm}_m(\cdot, A^{\lhd}[\gamma^{\lhd}]) \to \mathcal{V}\\
&\}\\
&\widetilde{\mathcal{T}}_m(\Gamma) \triangleq \{\\
&\quad A^{\lhd} \in \mathsf{type}^1_m(\Gamma^{\lhd});\\
&\quad A^{\blacktriangleright} : (\gamma^{\lhd} : \mathsf{sb}_m(\cdot, \Gamma^{\lhd})) \to (\gamma^{\blacktriangleright} : \Gamma^{\blacktriangleright}(\gamma^{\lhd})) \to \mathsf{tm}_m(\cdot, A^{\lhd}[\gamma^{\lhd}]) \to \mathcal{V}\\
&\quad M^{\lhd} \in \mathsf{tm}_m(\Gamma^{\lhd}, A^{\lhd});\\
&\quad M^{\blacktriangleright} : (\gamma^{\lhd} : \mathsf{sb}_m(\cdot, \Gamma^{\lhd})) \to (\gamma^{\blacktriangleright} : \Gamma^{\blacktriangleright}(\gamma^{\lhd})) \to A^{\blacktriangleright}(\gamma^{\lhd}, \gamma^{\blacktriangleright}, M^{\lhd}[\gamma^{\lhd}])\\
&\}\\
&\tau_m(\Gamma) \triangleq (A^{\lhd}, A^{\blacktriangleright}, M^{\lhd}, M^{\blacktriangleright}) \mapsto (A^{\lhd}, A^{\blacktriangleright})
\end{aligned}
$$

Thus a type over a context $\Gamma = (\Gamma^{\lhd}, \varphi_\Gamma)$ in the glued model consists of a type $\Gamma^{\lhd} \vdash A^{\lhd}\ \mathsf{type}_1 \mathbin{@} m$ of the system, along with a predicate—a family of $\mathcal{V}$-small sets indexed over both closing substitutions $\gamma^{\lhd}$ that satisfy the predicate $\Gamma^{\blacktriangleright}$ and terms of type $A^{\lhd}$ closed under that substitution.

In addition to these, a term over $\Gamma$ in the glued model also comes with a term $\Gamma^{\lhd} \vdash M^{\lhd} : A^{\lhd} \mathbin{@} m$ of that type, along with a section $M^{\blacktriangleright}$ of the aforementioned family. This section produces a proof that the predicate holds at that term after we close it by applying any appropriate substitution $\gamma^{\lhd}$ of which the predicate $\Gamma^{\blacktriangleright}$ holds. The reindexing action of these presheaves is defined by composition of substitutions, as well as the action of substitution on types and terms.

We must show that this defines a representable natural transformation in the sense of Section 1.4.1. Suppose that we have a morphism $\lfloor A \rfloor : \mathrm{Hom}(\mathbf{y}(\Gamma), [\![\text{🔒}_\mu]\!]^*\mathcal{T}_n)$ with $\mu : \mathrm{Hom}_{\mathcal{M}}(n, m)$. By Yoneda, this is exactly a type $A^{\lhd} \in \mathsf{type}^1_n(\Gamma^{\lhd}.\text{🔒}_\mu)$ along with a family

$$A^{\blacktriangleright} : (\gamma^{\lhd} : \mathsf{sb}_m(\cdot, \Gamma^{\lhd}.\text{🔒}_\mu)) \to \Gamma.\text{🔒}_\mu^{\blacktriangleright}(\gamma^{\lhd}) \to \mathsf{tm}_m(\cdot, A^{\lhd}[\gamma^{\lhd}]) \to \mathcal{V}$$

We will show that the following object in $\mathcal{C}[n]$ satisfies the required universal property:

$$
\begin{aligned}
\Gamma.(\mu \mid A)^{\lhd} &= \Gamma^{\lhd}.(\mu \mid A^{\lhd})\\
\Gamma.(\mu \mid A)^{\blacktriangleright} &= \lambda(\gamma^{\lhd}.M^{\lhd}).\ (\gamma^{\blacktriangleright} : \Gamma^{\blacktriangleright}(\gamma^{\lhd})) \times A^{\blacktriangleright}(\gamma^{\lhd}.\text{🔒}_\mu, \gamma^{\blacktriangleright}, M^{\lhd})
\end{aligned}
$$

where we have surreptitiously used the universal property of context extension to 'pattern match' on a closing substitution for $\Gamma^{\lhd}.(\mu \mid A^{\lhd})$ and decompose it to a substitution $\gamma^{\lhd} : \cdot \to \Gamma^{\lhd}$, and a term $\cdot.\text{🔒}_\mu \vdash M^{\lhd} : A^{\lhd}[\gamma^{\lhd}.\text{🔒}_\mu] \mathbin{@} n$ whose context simplifies to the empty context by our assumption. We also know that $\gamma^{\blacktriangleright}$ is in $\Gamma.\text{🔒}_\mu^{\blacktriangleright}(\gamma^{\lhd}.\text{🔒}_\mu)$, because $\varphi_{\Gamma.\text{🔒}_\mu}(\gamma^{\blacktriangleright}) \triangleq \varphi_\Gamma(\gamma^{\blacktriangleright}).\text{🔒}_\mu = \gamma^{\lhd}.\text{🔒}_\mu$.

It remains to show that this fits into a pullback square. We define $\mathbf{p} : \Gamma.(\mu \mid A) \to \Gamma$ as follows: on the syntactic level it simply postcomposition $\uparrow \circ - : \mathsf{sb}_m(\cdot, \Gamma^{\lhd}.(\mu \mid A^{\lhd})) \to \mathsf{sb}_m(\cdot, \Gamma^{\lhd})$. This is 'tracked' on the level of proof predicates by fibrewise projecting the first component of $(\gamma^{\blacktriangleright} : \Gamma^{\blacktriangleright}(\gamma^{\triangleleft})) \times A^{\blacktriangleright}(\gamma^{\triangleleft}.\text{🔒}_{\mu}, \gamma^{\blacktriangleright}, M^{\triangleleft})$. We may define $\mathbf{q} : \widetilde{\mathcal{T}}_n(\llbracket \text{🔒}_{\mu} \rrbracket(\Gamma.(\mu \mid A)))$ by letting $M^{\triangleleft} \triangleq \mathbf{v}_0$, and defining the section $M^{\blacktriangleright}$ to essentially project the second component of the same dependent product.

We must then show that the resulting square

$$\begin{tikzcd}
\mathbf{y}(\Delta) \arrow[drr, bend left, "\lfloor M \rfloor"] \arrow[dr, dashed, "\mathbf{y}(\gamma.M)"] \arrow[ddr, bend right, "\mathbf{y}(\gamma)"'] & & \\
& \mathbf{y}(\Gamma.(\mu \mid A)) \arrow[r, "\lfloor \mathbf{q} \rfloor"] \arrow[d, "\mathbf{y}(\mathbf{p})"'] & \llbracket \text{🔒}_{\mu} \rrbracket^{*}\widetilde{\mathcal{T}}_n \arrow[d, "\llbracket \text{🔒}_{\mu} \rrbracket^{*}\tau_n"] \\
& \mathbf{y}(\Gamma) \arrow[r, "\lfloor A \rfloor"'] & \llbracket \text{🔒}_{\mu} \rrbracket^{*}\mathcal{T}_n
\end{tikzcd}$$

is a pullback by proving it for representables. The syntactic part of $\gamma.M$ is forced to be $\gamma^{\triangleleft}.M^{\triangleleft}$ by the universal property of context extension. This is then tracked on the proof predicate level by the map whose fibre $\Delta^{\blacktriangleright}(\delta^{\triangleleft}) \to \Gamma.(\mu \mid A)^{\blacktriangleright}(\gamma^{\triangleleft} \circ \delta^{\triangleleft}, M^{\triangleleft}[\delta^{\triangleleft}.\text{🔒}_{\mu}])$ is defined by sending $x \in \Delta^{\blacktriangleright}(\delta^{\triangleleft})$ to $(\gamma^{\blacktriangleright}(x), M^{\blacktriangleright}(\delta^{\triangleleft}.\text{🔒}_{\mu}, x))$. In fact, this definition is forced if we want the diagram to commute, and consequently the square is a pullback.

### *The Glued Modal Structure*

We will now show that the glued model supports modal types. This is the main new feature of our type theory, and correspondingly this is the most novel part of this proof. To begin with, we must define a pair of maps making the following diagram commute, where $\mu : \mathrm{Hom}(n, m)$:

$$\begin{tikzcd}
\llbracket \text{🔒}_{\mu} \rrbracket^{*}\widetilde{\mathcal{T}}_n \arrow[r, "\mathbf{mod}_{\mu}"] \arrow[d] & \widetilde{\mathcal{T}}_m \arrow[d, "\tau_m"] \\
\llbracket \text{🔒}_{\mu} \rrbracket^{*}\mathcal{T}_n \arrow[r, "\mathbf{Mod}_{\mu}"'] & \mathcal{T}_m
\end{tikzcd}$$

We define these maps (in slightly informal notation) by mapping a type $(\Gamma^{\triangleleft}.\text{🔒}_{\mu} \vdash A^{\triangleleft}\ \mathsf{type}_1 \mathbin{@} n, A^{\blacktriangleright})$ and a term $(\Gamma^{\triangleleft}.\text{🔒}_{\mu} \vdash M^{\triangleleft} : A^{\triangleleft} \mathbin{@} n, M^{\blacktriangleright})$ to a type and term over $\Gamma^{\triangleleft}$ respectively:

$$\begin{aligned}
\mathbf{Mod}_{\mu}(A)^{\triangleleft} &= \langle \mu \mid A^{\triangleleft} \rangle \\
\mathbf{Mod}_{\mu}(A)^{\blacktriangleright} &= \lambda \gamma^{\triangleleft}, \gamma^{\blacktriangleright}, M^{\triangleleft}. \\
&\qquad (N^{\triangleleft} \in \mathsf{tm}_n(\cdot, A^{\triangleleft}[\gamma^{\triangleleft}.\text{🔒}_{\mu}])) \times (\mathsf{mod}_{\mu}(N^{\triangleleft}) = M^{\triangleleft}) \times A^{\blacktriangleright}(\gamma^{\triangleleft}.\text{🔒}_{\mu}, \gamma^{\blacktriangleright}, N^{\triangleleft}) \\
\mathbf{mod}_{\mu}(M)^{\triangleleft} &= \mathsf{mod}_{\mu}(M^{\triangleleft}) \\
\mathbf{mod}_{\mu}(M)^{\blacktriangleright} &= \lambda \gamma^{\triangleleft}, \gamma^{\blacktriangleright}.\ (M^{\triangleleft}[\gamma^{\triangleleft}.\text{🔒}_{\mu}], \star, M^{\blacktriangleright}(\gamma^{\triangleleft}.\text{🔒}_{\mu}, \gamma^{\blacktriangleright}))
\end{aligned}$$

Thus, the predicate at the modal type holds only of those closed terms that are of the form $\mathsf{mod}_{\mu}(N^{\triangleleft})$ for some appropriately typed term $N^{\triangleleft}$, for which furthermore the predicate $A^{\blacktriangleright}$ holds. This is achieved by existentially quantifying over the latter set, and then using the (extensional) identity type.

To complete the interpretation we must show how to interpret the modal elimination rule in the glued model. Unlike the reductionist, categorical approach of Section 1.4.1, we will not use left lifting structures, as the resulting types become intractable. We will instead directly construct the appropriate term in the model in the language of generalized algebraic theories.

Assume that we have $\nu : \mathrm{Hom}_{\mathcal{M}}(o, n)$, $\mu : \mathrm{Hom}_{\mathcal{M}}(n, m)$, a motive $\Gamma.(\mu \mid \langle \nu \mid A \rangle) \vdash B\ \mathsf{type}_1 \mathrel{@} m$, a term $\Gamma.\text{🔒}_\mu \vdash M_0 : \langle \nu \mid A \rangle \mathrel{@} n$, and a term $\Gamma.(\mu \circ \nu \mid A) \vdash M_1 : B[\uparrow.\mathsf{mod}_\nu(\mathbf{v}_0)] \mathrel{@} m$. We aim to construct a term

$$\Gamma \vdash \mathsf{let}_\mu\ \mathsf{mod}_\nu(\_) \leftarrow M_0\ \mathsf{in}\ M_1 : B[\mathsf{id}.M_0] \mathrel{@} m$$

Each one of those terms in the glued model comes with an associated term in the type theory which we denote by superscripting with $(-)^{\lhd}$. We use these to construct the associated term

$$\Gamma^{\lhd} \vdash (\mathsf{let}_\mu\ \mathsf{mod}_\nu(\_) \leftarrow M_0\ \mathsf{in}\ M_1)^{\lhd} \triangleq \mathsf{let}_\mu\ \mathsf{mod}_\nu(\_) \leftarrow M_0^{\lhd}\ \mathsf{in}\ M_1^{\lhd} : B^{\lhd}[\mathsf{id}.M_0^{\lhd}] \mathrel{@} m \tag{1.16}$$

It remains to construct the relevant section

$$S^{\blacktriangleright} : (\gamma^{\lhd} : \mathsf{sb}_m(\cdot, \Gamma^{\lhd})) \to (\gamma^{\blacktriangleright} : \Gamma^{\blacktriangleright}(\gamma^{\lhd})) \to B[\mathsf{id}.M_0]^{\blacktriangleright}(\gamma^{\lhd}, \gamma^{\blacktriangleright}, (\mathsf{let}_\mu\ \mathsf{mod}_\nu(\_) \leftarrow M_0^{\lhd}\ \mathsf{in}\ M_1^{\lhd})[\gamma^{\lhd}])$$

Expanding the definition of $B[\mathsf{id}.M_0]$ and simplifying makes the codomain equal to

$$B^{\blacktriangleright}(\gamma^{\lhd}.M_0^{\lhd}[\gamma^{\lhd}.\text{🔒}_\mu], (\gamma^{\blacktriangleright}, M_0^{\blacktriangleright}(\gamma^{\lhd}.\text{🔒}_\mu, \gamma^{\blacktriangleright})), \mathsf{let}_\mu\ \mathsf{mod}_\nu(\_) \leftarrow M_0^{\lhd}[\gamma^{\lhd}.\text{🔒}_\mu]\ \mathsf{in}\ M_1^{\lhd}[(\gamma^{\lhd} \circ \uparrow).\mathbf{v}_0])$$

This is expected: the substitution in the first argument is the result of postcomposing $\mathsf{id}.M_0^{\lhd}$ to $\gamma^{\lhd}$. For the second argument we notice that $B$ is in context $\Gamma.(\mu \mid \langle \nu \mid A \rangle)$, and the predicate of a context extension consists of a proof $\gamma^{\blacktriangleright}$ for $\Gamma$ and a proof for the type by which we have extended, for which we promptly substitute whatever the predicate for $M_0$ yields. Finally, after applying the substitution $\gamma^{\lhd}$ the term Eq. (1.16) expands to the third argument above.

At this point, the strategy for finding the canonical form of $\mathsf{let}_\mu\ \mathsf{mod}_\nu(\_) \leftarrow M_0^{\lhd}\ \mathsf{in}\ M_1^{\lhd}$ becomes evident: ignoring closing substitutions for a moment, we must intuitively first find the canonical form $\mathsf{mod}_\nu(N^{\lhd})$ of $M_0^{\lhd}$, substitute $N^{\lhd}$ into $M_1^{\lhd}$, and then find the canonical form of that. The construction of the section $S^{\blacktriangleright}$ will closely follow this strategy.

Thus, given $\gamma^{\lhd}$ and $\gamma^{\blacktriangleright}$ as in the premises of $S^{\blacktriangleright}$, we define $(N^{\lhd}, \star, x^{\blacktriangleright}) \triangleq M_0^{\blacktriangleright}(\gamma^{\lhd}.\text{🔒}_\mu, \gamma^{\blacktriangleright})$, where

$$\begin{aligned} N^{\lhd} &: \mathsf{tm}_o(\cdot, A^{\lhd}[\gamma^{\lhd}.\text{🔒}_{\mu\circ\nu}]) \\ \star &: \mathsf{mod}_\nu(N^{\lhd}) = M_0^{\lhd}[\gamma^{\lhd}.\text{🔒}_\mu] \\ x^{\blacktriangleright} &: A^{\blacktriangleright}(\gamma^{\lhd}.\text{🔒}_{\mu\circ\nu}, \gamma^{\blacktriangleright}, N^{\lhd}) \end{aligned}$$

The predicate for $M_0$ has thus given us a canonical form $\mathsf{mod}_\nu(N^{\lhd})$ for $M_0^{\lhd}[\gamma^{\lhd}.\text{🔒}_\mu]$. This can be combined with $\gamma^{\lhd}$ into a substitution $\gamma^{\lhd}.N^{\lhd} : \mathsf{sb}_m(\cdot, \Gamma^{\lhd}.(\mu \circ \nu \mid A^{\lhd}))$ which is closing for $M_1$. Moreover, we have that

$$(\gamma^{\blacktriangleright}, x^{\blacktriangleright}) : \Gamma.(\mu \circ \nu \mid A)^{\blacktriangleright}(\gamma^{\lhd}.N^{\lhd})$$

Hence, we can now apply the section for $M_1$ to get

$$M_1^{\blacktriangleright}(\gamma^{\lhd}.N^{\lhd}, (\gamma^{\blacktriangleright}, x^{\blacktriangleright})) : B[\uparrow.\mathsf{mod}_\nu(\mathbf{v}_0)]^{\blacktriangleright}(\gamma^{\lhd}.N^{\lhd}, (\gamma^{\blacktriangleright}, x^{\blacktriangleright}), M_1^{\lhd}[\gamma^{\lhd}.N^{\lhd}])$$

Expanding the action of the substitution on $B$ shows that this type is equal to

$$B^{\blacktriangleright}(\gamma^{\lhd}.\mathsf{mod}_\nu(N^{\lhd}), (\gamma^{\blacktriangleright}, (N^{\lhd}, \star, x^{\blacktriangleright})), M_1^{\lhd}[\gamma^{\lhd}.N^{\lhd}])$$

Finally, using the equation $\mathsf{mod}_\nu(N^{\lhd}) = M_0^{\lhd}[\gamma^{\lhd}.\text{🔒}_\mu]$, folding the definition of $M_0^{\blacktriangleright}$, and applying the $\beta$ rule for modal types to $\mathsf{mod}_\nu(N^{\lhd}) = M_0^{\lhd}[\gamma^{\lhd}.\text{🔒}_\mu]$ shows that this is equal to the codomain of $S^{\blacktriangleright}$.

*The Glued* $\prod$ *Structure*

The case of dependent products is fortunate: while not entirely mode-local, the construction essentially follows that of the standard glued model for Martin-Löf type theory. By Lemma 1.4.6, elements of $\mathbf{P}_{[\![\text{\faLock}_\mu]\!]^*\tau_n}(\mathcal{T}_m)(\Gamma)$ bijectively correspond to pairs of types $(A^{\lhd} \in \mathcal{T}_n(\Gamma^{\lhd}.\text{\faLock}_\mu), B^{\lhd} \in \mathcal{T}_m(\Gamma^{\lhd}.(\mu \mid A^{\lhd})))$ along with associated predicates $A^{\blacktriangleright}$ and $B^{\blacktriangleright}$, and similarly for $\mathbf{P}_{[\![\text{\faLock}_\mu]\!]^*\tau_n}(\widetilde{\mathcal{T}_m})(\Gamma)$. We silently transport under these to define the following type over $\Gamma^{\lhd}$:

$$
\begin{aligned}
\textstyle\prod(A,B)^{\lhd} &= (\mu \mid A^{\lhd}) \to B^{\lhd} \\
\textstyle\prod(A,B)^{\blacktriangleright} &= \lambda\gamma^{\lhd}.\ \lambda\gamma^{\blacktriangleright}.\ \lambda M^{\lhd}. \\
&\qquad (N^{\lhd} : \mathsf{tm}_n(\cdot, A^{\lhd}[\gamma^{\lhd}.\text{\faLock}_\mu])) \to \\
&\qquad (N^{\blacktriangleright} : A^{\blacktriangleright}(\gamma^{\lhd}.\text{\faLock}_\mu, \gamma^{\blacktriangleright}, N^{\lhd})) \to \\
&\qquad B^{\blacktriangleright}(\gamma^{\lhd}.N^{\lhd}, (\gamma^{\blacktriangleright}, N^{\blacktriangleright}), M^{\lhd}(N^{\lhd})) \\
\mathbf{lam}(M)^{\lhd} &= \lambda(M^{\lhd}) \\
\mathbf{lam}(M)^{\blacktriangleright} &= \lambda\gamma^{\lhd}, \gamma^{\blacktriangleright}, N^{\lhd}, N^{\blacktriangleright}.\ M^{\blacktriangleright}(\gamma^{\lhd}.N^{\lhd}, (\gamma^{\blacktriangleright}, N^{\blacktriangleright}))
\end{aligned}
$$

We wish to show that these arrows form a pullback square. To this end, suppose that we have some context $\Delta$ as well as arrows $\lfloor M \rfloor : \mathbf{y}(\Delta) \to \widetilde{\mathcal{T}_m}$ and $(\lfloor A \rfloor, \lfloor B \rfloor) : \mathbf{y}(\Delta) \to \mathbf{P}_{[\![\text{\faLock}_\mu]\!]^*\tau_n}(\mathcal{T}_m)$. We want to construct a unique mediating arrow in the diagram

$$
\begin{tikzcd}
\mathbf{y}(\Delta) \arrow[dr, dashed] \arrow[drr, bend left, "\lfloor M \rfloor"] \arrow[ddr, bend right, "(\lfloor A \rfloor, \lfloor B \rfloor)"'] & & \\
& \mathbf{P}_{[\![\text{\faLock}_\mu]\!]^*\tau_n}(\widetilde{\mathcal{T}_m}) \arrow[r, "\mathbf{lam}"] \arrow[d] & \widetilde{\mathcal{T}_m} \arrow[d, "\tau_m"] \\
& \mathbf{P}_{[\![\text{\faLock}_\mu]\!]^*\tau_n}(\mathcal{T}_m) \arrow[r, "\prod"'] & \mathcal{T}_m
\end{tikzcd}
$$

We use the aforementioned bijection to define this arrow as the one corresponding to a pair $(\lfloor A \rfloor, \lfloor M_0 \rfloor)$ where $M_0$ is a term in $\widetilde{\mathcal{T}_m}(\Delta.(\mu \mid A))$ over $B$. We know that $M$ lies over $\prod(A, B)$, so we define

$$
\begin{aligned}
M_0^{\lhd} &= M^{\lhd}[\uparrow](\mathbf{v}_0) \\
M_0^{\blacktriangleright} &= \lambda(\gamma^{\lhd}.N^{\lhd}), (\gamma^{\blacktriangleright}, N^{\blacktriangleright}).\ M^{\blacktriangleright}(\gamma^{\lhd}, \gamma^{\blacktriangleright}, N^{\lhd}, N^{\blacktriangleright})
\end{aligned}
$$

In order to show that this commutes, observe that $\mathbf{lam}(M_0)^{\lhd} = \lambda(M^{\lhd}[\uparrow](\mathbf{v}_0))$, which is equal to $M^{\lhd}$ by the $\eta$ rule. This argument also enforces the uniqueness of the choice of $M_0^{\lhd}$: given some $M_1^{\lhd}$ such that $\lambda(M^{\lhd}[\uparrow](\mathbf{v}_0)) = \lambda(M_1^{\lhd})$, we would have that $M_1^{\lhd} = M^{\lhd}[\uparrow](\mathbf{v}_0)$ by congruence and $\beta$. By applying essentially the same argument in the metatheory we also obtain uniqueness for the semantic section $M_0^{\blacktriangleright}$ associated with $M_0$.

*The Glued Sigma Structure*

The glued sigma structure is identical to the one found in, for instance, Kaposi, Huber, and Sattler [KHS19]. We wish to construct the following pullback

$$\begin{tikzcd}
\sum_{A:\mathcal{T}_m}\sum_{B:\mathcal{T}_m^{\tau_m^{-1}(A)}}\sum_{M:\tau_m^{-1}(A)}\tau_m^{-1}(B(M)) \arrow[r, "\mathbf{pair}"] \arrow[d] \arrow[dr, phantom, "\lrcorner", very near start] & \widetilde{\mathcal{T}}_m \arrow[d, "\tau_m"] \\
\mathbf{P}_{\tau_m}(\mathcal{T}_m) \arrow[r, "\sum"'] & \mathcal{T}_m
\end{tikzcd}$$

We will define $\sum$ and **pair** over it as follows:

$$\begin{aligned}
\textstyle\sum(A,B)^{\triangleleft} &= \textstyle\sum(A^{\triangleleft},B^{\triangleleft}) \\
\textstyle\sum(A,B)^{\blacktriangleright} &= \lambda\gamma^{\triangleleft},\gamma^{\blacktriangleright},(M^{\triangleleft},N^{\triangleleft}).\ (M^{\blacktriangleright} : A^{\blacktriangleright}(\gamma^{\triangleleft},\gamma^{\blacktriangleright})) \times B^{\blacktriangleright}(\nu,\gamma^{\triangleleft}.M^{\triangleleft},(\gamma^{\blacktriangleright},M^{\blacktriangleright})) \\
\mathbf{pair}(M,N)^{\triangleleft} &= (M^{\triangleleft},N^{\triangleleft}) \\
\mathbf{pair}(M,N)^{\blacktriangleright} &= \lambda\gamma^{\triangleleft},\gamma^{\blacktriangleright}.\ (M^{\blacktriangleright}(\gamma^{\triangleleft},\gamma^{\blacktriangleright}),N^{\blacktriangleright}(\gamma^{\triangleleft},\gamma^{\blacktriangleright}))
\end{aligned}$$

The verification that this forms a pullback and commutes is entirely routine and thus elided.

*The Glued Intensional Identity Structure*

The intensional identity structure is a standard construction. It is still complex, however, because lifting structures are troublesome to write down in the standard setting as well as ours. First, however, we define the formation rules as follows:

$$\begin{aligned}
\mathbf{Id}(A,M_0,M_1)^{\triangleleft} &= \mathsf{Id}_{A^{\triangleleft}}(M_0^{\triangleleft},M_1^{\triangleleft}) \\
\mathbf{Id}(A,M_0,M_1)^{\blacktriangleright} &= \lambda\gamma^{\triangleleft},\gamma^{\blacktriangleright},N^{\triangleleft}. \\
&\qquad (M_0^{\triangleleft}[\gamma^{\triangleleft}] = M_1^{\triangleleft}[\gamma^{\triangleleft}]) \times \\
&\qquad (M_0^{\blacktriangleright}(\gamma^{\triangleleft},\gamma^{\blacktriangleright}) = M_1^{\blacktriangleright}(\gamma^{\triangleleft},\gamma^{\blacktriangleright})) \times (N^{\triangleleft} = \mathsf{refl}(M_0^{\triangleleft})) \\
\mathbf{refl}(M)^{\triangleleft} &= \mathsf{refl}(M^{\triangleleft}) \\
\mathbf{refl}(M)^{\blacktriangleright} &= \lambda\gamma^{\triangleleft},\gamma^{\blacktriangleright}.\ (\star,\star,\star)
\end{aligned}$$

For the left lifting structure, bearing in mind the observations made in the construction of $\mathbf{open}_{\mu}$ we write the following:

$$\begin{aligned}
&\mathbf{J}(C,c,N)^{\triangleleft} = \mathsf{J}(C^{\triangleleft},c^{\triangleleft},M^{\triangleleft}) \\
&\mathbf{J}(C,c,N)^{\blacktriangleright} = \lambda\gamma^{\triangleleft},\gamma^{\blacktriangleright}.\ c^{\blacktriangleright}(\gamma^{\triangleleft}.M_0^{\triangleleft}[\gamma^{\triangleleft}],(\gamma^{\blacktriangleright},M_0^{\blacktriangleright}(\gamma^{\triangleleft},\gamma^{\blacktriangleright}))) \\
&\qquad \text{where } N^{\blacktriangleright}(\gamma^{\triangleleft},\gamma^{\blacktriangleright}) : (M_0^{\triangleleft}[\gamma^{\triangleleft}] = M_1^{\triangleleft}[\gamma^{\triangleleft}]) \times (M_0^{\blacktriangleright}(\gamma^{\triangleleft},\gamma^{\blacktriangleright}) = M_1^{\blacktriangleright}(\gamma^{\triangleleft},\gamma^{\blacktriangleright})) \times (N^{\triangleleft} = \mathsf{refl}(M_0^{\triangleleft}))
\end{aligned}$$

*The Glued Boolean Structure*

For the boolean case, we once again must contend with a lifting structure but a far simpler one because $\mathbb{B}$ is a closed type. For the formation and introduction rules, we have the following definitions:

$$\begin{aligned}
\mathbf{Bool}^{\triangleleft} &= \mathbb{B} & \mathbf{Bool}^{\blacktriangleright} &= \lambda\gamma^{\triangleleft},\gamma^{\blacktriangleright},M^{\triangleleft}.\ (M^{\triangleleft}[\gamma^{\triangleleft}] = \mathsf{tt}) + (M^{\triangleleft}[\gamma^{\triangleleft}] = \mathsf{ff}) \\
\mathbf{tt}^{\triangleleft} &= \mathsf{tt} & \mathbf{tt}^{\blacktriangleright} &= \lambda_.\ \iota_0(\star) \\
\mathbf{ff}^{\triangleleft} &= \mathsf{ff} & \mathbf{ff}^{\blacktriangleright} &= \lambda_.\ \iota_1(\star)
\end{aligned}$$

We must now define the left lifting structure we called $\mathbf{if} : [\mathbf{tt}, \mathbf{ff}] \pitchfork \tau_m$. We can just write out the term without further remark in this case:

$$\mathbf{if}(C, [c_0, c_1], M)^{\triangleleft} = \mathsf{if}(C^{\triangleleft}; c_0^{\triangleleft}; c_1^{\triangleleft}; M^{\triangleleft})$$
$$\mathbf{if}(C, [c_0, c_1], M)^{\blacktriangleright} = \lambda \gamma^{\triangleleft}, \gamma^{\blacktriangleright}. \begin{cases} c_0^{\blacktriangleright}(\gamma^{\triangleleft}, \gamma^{\blacktriangleright}) & M^{\blacktriangleright}(\gamma^{\triangleleft}, \gamma^{\blacktriangleright}) = \iota_0(\star) \\ c_1^{\blacktriangleright}(\gamma^{\triangleleft}, \gamma^{\blacktriangleright}) & M^{\blacktriangleright}(\gamma^{\triangleleft}, \gamma^{\blacktriangleright}) = \iota_1(\star) \end{cases}$$

*The Glued Universe Structure*

The construction of the glued structure on universes as been a source of consternation in many previous gluing proofs. We can, however, adapt the methodology of Coquand [Coq18] in order to make this more or less immediate. First, let us recall that we have a subuniverse $\mathcal{V}' \subset \mathcal{V}$. Next, let us define the presheaf of small types as follows:

$$\begin{aligned}
&\mathcal{S}_m(\Gamma) \triangleq \{ \\
&\quad A^{\triangleleft} \in \mathsf{type}^0_m(\Gamma^{\triangleleft}); \\
&\quad A^{\blacktriangleright} : (\mu : \mathrm{Hom}(m, -)) \to (\gamma^{\triangleleft} : \mathsf{sb}_m(\cdot.\text{🔒}_{\mu}, \Gamma^{\triangleleft})) \to (\gamma^{\blacktriangleright} : \Gamma^{\blacktriangleright}(\mu, \gamma)) \to \mathsf{tm}_m(\cdot.\text{🔒}_{\mu}, \Uparrow A^{\triangleleft}[\gamma^{\triangleleft}]) \to \mathcal{V}' \\
&\}
\end{aligned}$$

We can then define an inclusion of $\mathcal{S}_m$ into $\mathcal{T}_m$: $\mathbf{lift}(A^{\triangleleft}, A^{\blacktriangleright}) = (\Uparrow A^{\triangleleft}, A^{\blacktriangleright})$. Notice that we have made use of the fact that $\mathcal{V}' \subset \mathcal{V}$ to make the coercion on $A^{\blacktriangleright}$ entirely silent. Next, we observe that each of the semantic predicates on pi, sigma, $\langle \mu \mid - \rangle$, the identity types, and booleans all preserve $\mathcal{V}'$ smallness. If the inputted semantic predicates are $\mathcal{V}'$ small, then so are the outputted predicates. Therefore, each of these restricts to a small type in $\mathcal{S}_m$, using the lifting operators in the syntax. What remains is to construct an element of $\mathcal{T}_m$ which pulls-back to something isomorphic to $\mathcal{S}_m$. For this, we pick the following:

$$\mathbf{Uni}(*) = (\mathsf{U}, \lambda \mu, \gamma^{\triangleleft}, \gamma^{\blacktriangleright}, M^{\triangleleft}.\ \mathsf{tm}_m(\cdot.\text{🔒}_{\mu}, \Uparrow \mathsf{El}(M^{\triangleleft})) \to \mathcal{V}')$$

Crucially here, $\mathcal{V}'$ is small enough to fit in a proof-relevant predicate valued in $\mathcal{V}$. The pullback of $\tau_m$ along $\mathbf{Uni}$ can be calculated to be isomorphic to the following:

$$\begin{aligned}
&\tau_m^{-1}(\mathbf{Uni})(\Gamma) \cong \{ \\
&\quad M^{\triangleleft} \in \mathsf{tm}_m(\Gamma^{\triangleleft}, \mathsf{U}); \\
&\quad M^{\blacktriangleright} : (\mu : \mathrm{Hom}(m, -)) \to (\gamma^{\triangleleft} : \mathsf{sb}_m(\cdot.\text{🔒}_{\mu}, \Gamma^{\triangleleft})) \to (\gamma^{\blacktriangleright} : \Gamma^{\blacktriangleright}(\gamma)) \to \mathsf{tm}_m(\cdot.\text{🔒}_{\mu}, \Uparrow \mathsf{El}(M^{\triangleleft}[\gamma^{\triangleleft}])) \to \mathcal{V}' \\
&\}
\end{aligned}$$

However, we may now use that $\mathsf{Code}(-)$ and $\mathsf{El}(-)$ provide isomorphisms natural in $\Gamma^{\triangleleft}$ between $\mathsf{type}^0_m(\Gamma^{\triangleleft})$ and $\mathsf{tm}_m(\Gamma^{\triangleleft}, \mathsf{U})$ to conclude that $\mathbf{Uni}$ is the desired glued universe.

### 1.5.2 Deriving Canonicity

With the gluing model constructed, the rest of the proof is surprisingly easy and boils down to one fact:

**Theorem 1.5.7.** *The natural transformation $\pi : \mathcal{C}[m] \mapsto \mathbb{S}[m]$ from the glued model to the syntactic model is a morphism of models.*

*Proof.* This is immediate by inspection of the constructions in the previous section: each construction uses the corresponding syntactic operation and so projecting this out constitutes a morphism of models. $\square$

**Corollary 1.5.8.** *For any closed term $\vdash M : A \mathbin{@} m$, there is a witness for $[\![A]\!]^{\blacktriangleright}(M)$.*

*Proof.* Immediate by initiality and Theorem 1.5.7, we must have $\pi(\llbracket M \rrbracket) = M$, and so $\llbracket M \rrbracket^{\blacktriangleright}$ is the desired witness. $\square$

**Theorem 1.5.9** (Closed Term Canonicity)**.** *If* $\cdot \vdash M : A \mathrel{@} m$ *is a closed term, then the following conditions hold:*

- *If* $A = \mathbb{B}$ *then* $\cdot \vdash M = \mathsf{tt} : \mathbb{B} \mathrel{@} m$ *or* $\cdot \vdash M = \mathsf{ff} : \mathbb{B} \mathrel{@} m$.
- *If* $A = \mathsf{Id}_{A_0}(N_0, N_1)$ *then* $\cdot \vdash N_0 = N_1 : A_0 \mathrel{@} m$ *and* $\cdot \vdash M = \mathsf{refl}(N_0) : \mathsf{Id}_{A_0}(N_0, N_1) \mathrel{@} m$.
- *If* $A = \langle \nu \mid A_0 \rangle$ *then there is a term* $\cdot \vdash N : A_0 \mathrel{@} n$ *such that* $\cdot \vdash M = \mathsf{mod}_{\nu}(N) : \langle \nu \mid A_0 \rangle \mathrel{@} m$.

*Proof.* Immediate by Corollary 1.5.8 and the definition of the semantic predicates at $\mathbb{B}$, $\mathsf{Id}_{A_0}(N_0, N_1)$, and $\langle \mu \mid A_0 \rangle$ respectively. $\square$

## 1.6 *Strictifying a Weaker Lock Structure*

When constructing models of MTT, it will often prove more convenient to use Theorem 1.4.11, but this theorem still requires *strict* models of type theory, and a *strict* 2-functor choosing categories of contexts and dependent right adjoints between them. It has long been understood that many semantic situations are too weak, and require strictification in order to be assembled into a valid model of type theory.

Just as some models require strictification for the interpretation of substitution, some natural applications of MTT will require strictification of the interpretation of locks. Recall from Section 1.4.1 that we require a 2-functor interpreting the modes as categories, the locks as functors, and the 2-cells as natural transformations. In many cases, we will not obtain the equation $[\![\text{🔒}_{\mu\circ\nu}]\!] = [\![\text{🔒}_{\nu}]\!] \circ [\![\text{🔒}_{\mu}]\!]$, instead we will only get a canonical natural isomorphism between the two. Just as with substitution, one can attempt to find some special property of $[\![\text{🔒}_{\mu}]\!]$ or $[\![\text{🔒}_{\nu}]\!]$ which admits a suitably strict interpretation, but it will prove more fruitful to instead construct a general procedure for replacing a pseudo-functorial interpretation with a strictly functorial choice. The question of strictification for pseudo-functors has been well-studied in category theory [Pow89; Lac02]. In particular, for any pseudo-functorial map $F : \mathcal{M}^{\mathsf{coop}} \to \mathbf{Cat}$, we can construct a 2-functor $F' : \mathcal{M}^{\mathsf{coop}} \to \mathbf{Cat}$ together with a pseudonatural equivalence $E : F' \simeq F$.[4]

We will show that we may transfer a "model" of MTT along this pseudonatural transformation and thereby obtain a strict model of MTT.

### 1.6.1 *Preliminary Definitions*

Traditionally, the definition of pseudo-functors is given as maps between bicategories. For our purposes, we are only concerned with strict 2-categories like $\mathcal{M}$ and $\mathbf{Cat}$. Accordingly, we specialize the definition slightly:

**Definition 1.6.1.** A psuedofunctor between 2-categories $F : \mathcal{C} \to \mathcal{D}$ is a map of objects $F_0$ paired with a functor for each pair of objects in $\mathcal{C}$: $F_1^{A,B} : \mathrm{Hom}(A, B) \to \mathrm{Hom}(F_0(A), F_0(B))$. We will suppress the subscripts going forward, however, because they can be inferred from context.

Moreover, we require that for all $A, B, C : \mathcal{C}$ an *associator isomorphism*, an invertible natural 2-cell $\alpha_{A,B,C} : F^{B,C} \circ F^{A,B}(-) \cong F^{A,C}(- \circ -)$ as well as a unitor $\lambda_A : F^{A,A}(1_A) \cong 1_{F(A)}$.

These 2-cells are required to satisfy certain coherence conditions:

1. If $f : A \to B$, $1_f \circ \lambda_A = \alpha_{A,A,B} : F(f) \circ F(1_A) \cong F(f)$.
2. If $f : A \to B$, $\lambda_B \circ 1_f = \alpha_{A,B,B} : F(1_B) \circ F(f) \cong F(f)$.
3. If $f : A \to B$, $g : B \to C$, and $h : C \to D$, then

$$\alpha_{A,C,D} \circ \alpha_{A,B,C} = \alpha_{A,B,D} \circ \alpha_{B,C,D} : F(h) \circ F(g) \circ F(f) \cong F(h \circ g \circ f)$$

**Definition 1.6.2** (Pseudo-natural transformation)**.** A pseudo-natural transformation $\chi$ between two pseudo-functors $F, G : \mathcal{C} \to \mathcal{D}$ is a collection of 1-cells in $\mathcal{D}$, $\chi_C : F(C) \to G(D)$. Moreover, for each morphism $f : C_0 \to C_1$, we require an invertible 2-cell $\chi_f : \chi_{C_1} \circ F(f) \cong G(f) \circ \chi_{C_0}$. This are subject to the expected functoriality rules: $\chi_{1_A} = 1$ and $\chi_g \star \chi_f = \chi_{g \circ f}$.

Finally, we require that $\chi$ interacts correctly with the associator and unitor of $F$ and $G$. See Johnson and Yau [JY20] for details.

A textbook account of these definitions was recently given in Johnson and Yau [JY20].

We will use the following theorem as the heart of our strictification construction. It states that a pseudofunctor in $\mathbf{Cat}$ can be replaced with a strict 2-functor up to a pseudonatural equivalence.

[4] This procedure is akin to the well-known result that any cloven fibration can be replaced with a *split* cloven fibration [Str18].

**Theorem 1.6.3.** *Given a strict 2-category $\mathcal{C}$ and a pseudofunctor $F : \mathcal{C} \to \mathbf{Cat}$, there exists a strict 2-functor $F' : \mathcal{C} \to \mathbf{Cat}$ equipped with a pseudonatural equivalence $F \simeq F'$.*

*Proof.* This is a general corollary of the theorem stating that a pseudo-algebra for a 2-monad can be replaced with a strict algebra [Pow89]. A short description of the specialized result is given on the nLab [nLa15]. It can also be proven using the equivalence between 2-functors/pseudofunctors and split fibrations/fibrations (proven in Johnson and Yau [JY20]) along with Benabou's strictification theorem for Grothendieck fibrations [Str18]. □

### 1.6.2 Precomposition with an Equivalence

Most of the calculations in this construction revolve around precomposition with an equivalence of categories. Accordingly, it is useful to record a few elementary facts about such functors prior to delving into the MTT-specific details.

**Theorem 1.6.4.** *Given a pair of equivalent small categories $f : \mathcal{C}_0 \simeq \mathcal{C}_1$, the precomposition functor $f^*$ is an equivalence which preserves and reflects all limits and colimits, and is moreover a logical morphism $\mathbf{PSh}(\mathcal{C}_1) \to \mathbf{PSh}(\mathcal{C}_0)$.*[5]

*Proof.* It is well-known that $f^*$ is (co)continuous for any functor $f$, as limits and colimits are formed pointwise. Additionally, $f^*$ is an equivalence: supposing that $f$ has a pseudo-inverse $g$, then $g^*$ is pseudo-inverse to $f^*$. This pseudo-inverse also implies that $f^*$ reflects all (co)limits: if $f^*(X)$ is the limit of some diagram $f^* \circ D$, then $(f \circ g)^*(X)$ is the limit of $(f \circ g)^* \circ D$, which in turn implies that $X$ is the limit of $D$ as required.

It remains to show that $f^*$ preserves exponentials and the subobject classifier. To begin with, we will show that $f^*(X^Y) \cong f^*(X)^{f^*(Y)}$. Applying the Yoneda lemma, we fix $Z : \mathbf{PSh}(\mathcal{C}_0)$:

$$
\begin{aligned}
\mathrm{Hom}(Z, f^*(X^Y)) &\cong \mathrm{Hom}(f^*(g^*Z), f^*(X^Y)) \\
&\cong \mathrm{Hom}(g^*Z, X^Y) && f^* \text{ is an equivalence, and therefore full and faithful} \\
&\cong \mathrm{Hom}(g^*Z \times Y, X) \\
&\cong \mathrm{Hom}(f^*(g^*Z \times Y), f^*(X)) && f^* \text{ is full and faithful.} \\
&\cong \mathrm{Hom}(f^*(g^*Z) \times f^*(Y), f^*(X)) && f^* \text{ is continuous.} \\
&\cong \mathrm{Hom}(f^*(g^*Z), f^*(X)^{f^*(Y)}) \\
&\cong \mathrm{Hom}(Z, f^*(X)^{f^*(Y)})
\end{aligned}
$$

We now turn to showing that $f^*$ preserves the subobject classifier. First, we claim that $m : \mathrm{Hom}(X, Y)$ is a monomorphism in $\mathbf{PSh}(\mathcal{C}_0)$ if and only if $f^*(m)$ is a monomorphism. This follows from the fact that $f^*$ preserves and reflects limits, and in particular pullbacks. Moreover, because $f^*$ is essentially surjective, this implies that $\mathbf{Sub}(X) \cong \mathbf{Sub}(f^*(X))$ are naturally isomorphic as posets.

In order to complete the proof that $f^*$ preserves subobject classifiers. Recall that the subobject classifier $\Omega$ is characterized by a natural isomorphism $\mathsf{sb}_X \cong \mathrm{Hom}(X, \Omega)$, where $\mathsf{sb}_X$ is the poset of subobjects of $X$ (quotiented up to isomorphism). We apply the Yoneda lemma again, and fix $Z$ :

[5] We intend to apply this theorem in a setting where $\mathcal{C}_0$ and $\mathcal{C}_1$ are CwFs; then the presheaf categories are where $\mathcal{T}$ and $\widetilde{\mathcal{T}}$ live.

$\mathbf{PSh}(\mathcal{C}_0)$:

$$\begin{aligned}\mathrm{Hom}(Z, f^*\Omega_{\mathcal{C}_1}) &\cong \mathrm{Hom}(f^*(g^*Z'), f^*\Omega_{\mathcal{C}_1}) \\ &\cong \mathrm{Hom}(g^*Z', \Omega_{\mathcal{C}_1}) \\ &\cong \mathbf{Sub}(g^*Z') \\ &\cong \mathbf{Sub}(f^*(g^*Z')) \\ &\cong \mathrm{Hom}(f^*g^*Z', \Omega_{\mathcal{C}_0}) \\ &\cong \mathrm{Hom}(Z, \Omega_{\mathcal{C}_0})\end{aligned}$$ □

**Lemma 1.6.5.** *Fixing an equivalence between small categories $f : \mathcal{C}_0 \simeq \mathcal{C}_1$, for each $C : \mathcal{C}_0$, there exists $D : \mathcal{C}_1$ such that $f^*(\mathbf{y}(D)) \cong \mathbf{y}(C)$. Moreover, if $D : \mathcal{C}_1$, there exists $C : \mathcal{C}_0$ such that $f^*(\mathbf{y}(D)) \cong \mathbf{y}(C)$.*

*Proof.* For the first claim, let us choose $D = f(C)$. It is sufficient to show that $\mathrm{Hom}(C', C) \cong \mathrm{Hom}(f(C'), f(C))$, but this is immediate as $f$ is full and faithful.

For the second claim, we fix any adjoint pseudo-inverse to $f$: $f \dashv g$. We then select $C = g(D)$. We must show $\mathrm{Hom}(C', g(D)) \cong \mathrm{Hom}(f(C'), D)$, but this is immediate because we have assumed $f \dashv g$. □

**Lemma 1.6.6.** *Again fixing an equivalence between small categories $f : \mathcal{C}_0 \simeq \mathcal{C}_1$, if $\tau : \widetilde{\mathcal{T}} \to \mathcal{T}$ is a representable natural transformation in $\mathbf{PSh}(\mathcal{C}_1)$, then $f^*(\tau)$ is a representable natural transformation in $\mathbf{PSh}(\mathcal{C}_0)$.*

*Proof.* From Lemma 1.6.5, it is sufficient to show that a representable pullback exists for each morphism $f^*(\mathbf{y}(D)) \to f^*(\mathcal{T})$. However, because $f^*$ is full, faithful and preserves pullbacks, and $\tau$ is a representable natural transformation, we conclude that there is a pullback of the shape $f^*(\mathbf{y}(D))$, for some $D$. Again, by Lemma 1.6.5, we obtain the desired representable pullback. □

**Lemma 1.6.7.** *Precomposition by an equivalence preserves polynomial functors. More explicitly, if $f : \mathcal{C}_0 \simeq \mathcal{C}_1$ and $\tau : A \to B$, then $f^*(\mathbf{P}_\tau(X)) \cong \mathbf{P}_{f^*\tau}(f^*X) \in \mathrm{Hom}(\mathbf{PSh}(\mathcal{C}_0), \mathbf{PSh}(\mathcal{C}_0))$.*

*Proof.* Fixing $\tau : A \to B$, $\mathbf{P}_\tau = \sum_B \circ \prod_\tau \circ A^*$. We therefore wish to show the following:

$$f^* \circ \sum_B \circ \prod_\tau \circ A^* \cong \sum_{f^*B} \circ \prod_{f^*\tau} \circ (f^*A)^*$$

First, observe that for any functor which preserves terminal objects $F$, $F \circ \sum_X = \sum_{F(X)} \circ F$. Furthermore, Theorem 1.6.4 ensures that $f^*$ is a logical functor, which therefore must preserve $\prod$. Finally, Theorem 1.6.4 ensure that $f^*$ is left exact. We may then calculate:

$$\begin{aligned}f^* \circ \sum_B \circ \prod_\tau \circ A^* &\cong \sum_{f^*B} \circ f^* \circ \prod_\tau \circ A^* \\ &\cong \sum_{f^*B} \circ \prod_{f^*\tau} \circ f^* \circ A^* \\ &\cong \sum_{f^*B} \circ \prod_{f^*\tau} \circ (f^*A)^*\end{aligned}$$ □

**Lemma 1.6.8.** *Given an equivalence $e : \mathcal{C}_0 \simeq \mathcal{C}_1$, $f : \mathrm{Hom}_{\mathbf{PSh}(\mathcal{C}_1)/Z}(E, B)$, $i : \mathrm{Hom}_{\mathbf{PSh}(\mathcal{C}_1)/Z}(A, I)$ and an internal lifting structure $s : i \pitchfork f$ (Definition 1.4.7) in $\mathbf{PSh}(\mathcal{C}_1)/Z$, $e^*s$ induces a lifting structure for $e^*i$ against $e^*f$ in $\mathbf{PSh}(\mathcal{C}_0)/e^*Z$.*

*Proof.* First, we observe that $e^*$ defines an equivalence $e^*/Z$ between $\mathbf{PSh}(\mathcal{C}_1)/Z$ and $\mathbf{PSh}(\mathcal{C}_0)/e^*Z$ by sending $X \to Z$ to $e^*X \to e^*Z$. Moreover, this equivalence is logical because the subobject classifier, exponentials, limits and colimits of $\mathbf{PSh}(\mathcal{C}_{0,1})/Z$ are determined by the subobject classifier, exponentials, limits and colimits of $\mathbf{PSh}(\mathcal{C}_{0,1})$, all of which are preserved by $e^*$ by Theorem 1.6.4.[6]

Next, observe that $(e^*/Z)(s) : (e^*/Z)(i \pitchfork f)$. We will now unfold $(e^*/Z)(i \pitchfork f)$, using the fact that $e^*/Z$ is logical, and therefore preserve dependent products, equality (an equalizer), and comprehension (a pullback). For readability, let us set $\mathfrak{e} = e^*/Z$ in the following:

$$\begin{aligned}
\mathfrak{e}(i \pitchfork f) &\cong \mathfrak{e}(\textstyle\prod_{C:I\to B}\prod_{c:\prod_{a:A}E[C(i(a))]}\left\{j : \prod_{p:I}E[C(p)] \mid \forall a : A.\ j(i(a)) = c(a)\right\}) \\
&\cong \textstyle\prod_{C:\mathfrak{e}(I)\to\mathfrak{e}(B)}\prod_{c:\prod_{a:\mathfrak{e}(A)}\mathfrak{e}(E)[C(\mathfrak{e}(i)(a))]}\left\{j : \prod_{p:\mathfrak{e}(I)}\mathfrak{e}(E)[C(p)] \mid \forall a : \mathfrak{e}(A).\ j(\mathfrak{e}(i)(a)) = c(a)\right\} \\
&\cong \mathfrak{e}(i) \pitchfork \mathfrak{e}(f)
\end{aligned}$$

Accordingly, we may transport $\mathfrak{e}(s) : \mathfrak{e}(i \pitchfork f)$ to an element of the desired $\mathfrak{e}(i) \pitchfork \mathfrak{e}(f)$. □

**Lemma 1.6.9.** *If $f \cong f'$ in the arrow category of $\mathcal{C}$, then $\mathbf{P}_f \cong \mathbf{P}_{f'}$ as functors $\mathcal{C} \to \mathcal{C}$.*

*Proof.* Newstead [New18] shows that $\mathbf{P}_{-}$ is functorial for Cartesian squares. Because all isomorphisms in the arrow category are necessarily Cartesian, this functorial action induces the required isomorphism. □

**Lemma 1.6.10.** *If $f \cong f'$ and $g \cong g'$ in the arrow category of $\mathcal{C}$, then $f \pitchfork g \cong f' \pitchfork g'$ in $\mathcal{C}$.*

### 1.6.3 Weak Models of MTT

Prior to the strictification construction, we must define a *weak* model of MTT.

Traditionally, weak models of type theory have been concerned with weaker notions of *substitution*. In our case, we are predominately concerned with weaker lock structure, but retain a strict notion of substitution:

**Definition 1.6.11** (Weak context categories)**.** A weak choice of context categories is a pseudo-functor $\mathcal{C}[-] : \mathcal{M}^{\mathsf{coop}} \to \mathbf{Cat}_1$.

We may leave most of the structure from Section 1.4 untouched, as this structure only references one modality at a time. Accordingly, rather than duplicating the entire definition, when the connective in the weaker model is identical to the strict version, we will simply reference the stricter definitions.

**Definition 1.6.12** (Weak modal natural model)**.** Identical to Definition 1.4.4.

**Definition 1.6.13** (Weak pi structure)**.** Identical to Section 1.4.1.

**Definition 1.6.14** (Weak sigma structure)**.** Identical to Section 1.4.1.

**Definition 1.6.15** (Weak boolean structure)**.** Identical to Section 1.4.1.

**Definition 1.6.16** (Weak identity structure)**.** Identical to Section 1.4.1.

**Definition 1.6.17** (Weak universe structure)**.** Identical to Section 1.4.1.

[6]The exponentials of a slice category are not literally exponentials of the total category. However, they may be defined using a combination of the total space exponentials, as well as the subobject classifier [MM92, Chapter IV].

The exception to this pattern is the definition of the elimination structure for modal types (see Eq. (1.9)). This definition uses two modalities together, and so the pseudo-functoriality interferes slightly with the definition.

As before, we begin by forming the following pullback:

$$\begin{tikzcd}
\llbracket \text{🔒}_{\nu} \rrbracket^{*}\widetilde{\mathcal{T}}_o \arrow[dr, dashed, "i"] \arrow[drr, bend left, "\mathbf{mod}_{\nu}"] \arrow[ddr, bend right, "\llbracket \text{🔒}_{\nu} \rrbracket^{*}\tau_o"'] & & \\
& M \arrow[r] \arrow[d, "h"] \arrow[dr, phantom, very near start, "\lrcorner"] & \widetilde{\mathcal{T}}_n \arrow[d, "\tau_n"] \\
& \llbracket \text{🔒}_{\nu} \rrbracket^{*}\mathcal{T}_o \arrow[r, "\mathbf{Mod}_{\nu}"'] & \mathcal{T}_n
\end{tikzcd}$$

Unlike with the definition of a strict model, however, when we apply $\llbracket \text{🔒}_{\mu} \rrbracket^{*}$, however, we do not obtain an equality between $\llbracket \text{🔒}_{\mu\circ\nu} \rrbracket^{*}$ and $(\llbracket \text{🔒}_{\mu} \rrbracket \circ \llbracket \text{🔒}_{\nu} \rrbracket)^{*}$. Instead, we must explicitly insert the chosen isomorphism between these two objects. However, because pullbacks are stable under isomorphism, we obtain the following diagram:

$$\begin{tikzcd}
\llbracket \text{🔒}_{\mu\circ\nu} \rrbracket^{*}\widetilde{\mathcal{T}}_o \arrow[dr, dashed, "\llbracket \text{🔒}_{\mu} \rrbracket^{*} i \circ \iota_0"] \arrow[drr, bend left, "\llbracket \text{🔒}_{\mu} \rrbracket^{*}\mathbf{mod}_{\nu} \circ \iota_0"] \arrow[ddr, bend right, "\llbracket \text{🔒}_{\mu\circ\nu} \rrbracket^{*}\tau_o"'] & & \\
& \llbracket \text{🔒}_{\mu} \rrbracket^{*} M \arrow[r] \arrow[d, "\iota_1^{-1} \circ \llbracket \text{🔒}_{\mu} \rrbracket^{*} h"] \arrow[dr, phantom, very near start, "\lrcorner"] & \llbracket \text{🔒}_{\mu} \rrbracket^{*}\widetilde{\mathcal{T}}_n \arrow[d, "\llbracket \text{🔒}_{\mu} \rrbracket^{*}\tau_n"] \\
& \llbracket \text{🔒}_{\mu\circ\nu} \rrbracket^{*}\mathcal{T}_o \arrow[r, "\llbracket \text{🔒}_{\mu} \rrbracket^{*}\mathbf{Mod}_{\nu} \circ \iota_1"'] & \llbracket \text{🔒}_{\mu} \rrbracket^{*}\mathcal{T}_n
\end{tikzcd} \tag{1.17}$$

Where we have fixed the following isomorphism:

$$\iota_0 : \llbracket \text{🔒}_{\mu\circ\nu} \rrbracket^{*}\widetilde{\mathcal{T}}_o \cong \llbracket \text{🔒}_{\mu} \rrbracket \circ \llbracket \text{🔒}_{\nu} \rrbracket^{*}\widetilde{\mathcal{T}}_o$$
$$\iota_1 : \llbracket \text{🔒}_{\mu\circ\nu} \rrbracket^{*}\mathcal{T}_o \cong \llbracket \text{🔒}_{\mu} \rrbracket \circ \llbracket \text{🔒}_{\nu} \rrbracket^{*}\mathcal{T}_o$$

With these arrows fixed, we may define a weak modal structure:

**Definition 1.6.18** (Weak modal structure)**.** A *weak* modal structure first requires a commuting square for each modality:

$$\begin{tikzcd}
\llbracket \text{🔒}_{\mu} \rrbracket^{*}\widetilde{\mathcal{T}}_n \arrow[r, "\mathbf{mod}_{\mu}"] \arrow[d, "\llbracket \text{🔒}_{\mu} \rrbracket^{*}\tau_n"'] & \widetilde{\mathcal{T}}_m \arrow[d, "\tau_m"] \\
\llbracket \text{🔒}_{\mu} \rrbracket^{*}\mathcal{T}_n \arrow[r, "\mathbf{Mod}_{\mu}"'] & \mathcal{T}_m
\end{tikzcd}$$

Moreover, we further require a left-lifting structure in the slice over $Z = \llbracket \text{🔒}_{\mu\circ\nu} \rrbracket^* \mathcal{T}_o$:

$$\vdash \mathbf{open}^{\mu}_{\nu} : \llbracket \text{🔒}_{\mu} \rrbracket^* i \circ \iota_0 \pitchfork Z^*(\tau_m)$$

### 1.6.4 Coherence

Suppose that we are given a weak model $F$. Using the general 2-categorical result, we may construct $\mathcal{C}[-]$, a strict 2-functor $\mathcal{M}^{\mathsf{coop}} \to \mathbf{Cat}$ such that there is a pseudo-natural equivalence $\alpha_m : \mathcal{C}[m] \simeq F(m)$. We now will show that each of the components of the weak model $F$ can be transferred to strict components of $\mathcal{C}[-]$.

**Lemma 1.6.19.** *If $F$ supports a collection of natural models (Definition 1.6.12), then $\mathcal{C}[-]$ supports a collection of natural models.*

*Proof.* For each $m$, suppose that $\sigma_m : \tilde{V}_m \to V_m$ is a morphism in $\mathbf{PSh}(F(m))$ such that $F(\mu)^*\sigma_m$ is a natural model for all $\mu \in \mathrm{Hom}_{\mathcal{M}}(m,n)$. We choose $\tau_m : \tilde{\mathcal{T}}_m \to \mathcal{T}_m$ to be $\alpha^*_m\sigma_m$.

We must now show that $\llbracket \text{🔒}_{\mu} \rrbracket^*(\alpha^*_m\sigma_m) = (\alpha_m \circ \llbracket \text{🔒}_{\mu} \rrbracket)^*(\sigma_m)$ is natural model. First, we observe that $\alpha_m \circ \llbracket \text{🔒}_{\mu} \rrbracket \cong F(\mu) \circ \alpha_n$. Therefore,

$$\llbracket \text{🔒}_{\mu} \rrbracket^*(\alpha^*_m\sigma_m) \cong \alpha^*_n(F(\mu)^*\sigma_m)$$

By assumption, we know that $F(\mu)^*\sigma_m$ is a natural model, and by Lemma 1.6.6, this tells us that $\alpha^*_n(F(\mu)^*\sigma_m)$ is a natural model. As natural models are stable under isomorphism, this establishes the desired goal. □

**Lemma 1.6.20.** *If $F$ supports dependent products, then so does $\mathcal{C}[-]$.*

*Proof.* Let us fix $\mu \in \mathrm{Hom}_{\mathcal{M}}(n,m)$. By assumption, we have a pullback square:

$$\begin{tikzcd}
\mathbf{P}_{F(\mu)^*\sigma_n}(\tilde{V}_m) \arrow[r] \arrow[d] \arrow[dr, phantom, very near start, "\lrcorner"] & \tilde{V}_m \arrow[d, "\sigma_m"] \\
\mathbf{P}_{F(\mu)^*\sigma_n}(V_m) \arrow[r] & V_m
\end{tikzcd}$$

Now, because precomposition with an equivalence preserves finite limits (Theorem 1.6.4) and preserves polynomial functors (Lemma 1.6.7), we have an induced pullback square in $\mathbf{PSh}(\mathcal{C}[m])$ (setting $\mathfrak{e} = \alpha^*_m$):

$$\begin{tikzcd}
\mathbf{P}_{\mathfrak{e}(F(\mu)^*\sigma_n)}(\mathfrak{e}(\tilde{\mathcal{V}}_m)) \arrow[r] \arrow[d] \arrow[dr, phantom, very near start, "\lrcorner"] & \mathfrak{e}(\tilde{\mathcal{V}}_m) \arrow[d, "\mathfrak{e}(\sigma_m)"] \\
\mathbf{P}_{\mathfrak{e}(F(\mu)^*\sigma_n)}(\mathfrak{e}(\mathcal{V}_m)) \arrow[r] & \mathfrak{e}(\mathcal{V}_m)
\end{tikzcd}$$

Employing the pseudo-naturality of $\alpha$, we observe that $\mathfrak{e}(F(\mu)^*\sigma_n) \cong \llbracket \text{🔒}_{\mu} \rrbracket^*(\alpha^*_n\sigma_n)$. Using the definitions of $\tau_n$ and $\tau_m$ from Lemma 1.6.19, this induces the required pullback square for dependent products. □

**Lemma 1.6.21.** *If $F$ supports dependent sums, then so does $\mathcal{C}[-]$.*

*Proof.* Identical to Lemma 1.6.20. $\square$

**Lemma 1.6.22.** *If $F$ supports booleans, then so does $\mathcal{C}[-]$.*

*Proof.* First, we begin by showing that the formation and introduction rules from $F(m)$ transfer to formation and introduction rules in $\mathcal{C}[m]$. Applying the equivalence to the commuting square from Definition 1.6.15, we obtain the following:

$$\begin{tikzcd} \alpha_m^* 1 \arrow[r, shift left, "\mathbf{tt}"] \arrow[r, shift right, "\mathbf{ff}"'] \arrow[d, equal] & \widetilde{\mathcal{T}}_m \arrow[d, "\tau_m"] \\ \alpha_m^* 1 \arrow[r, "\mathbf{Bool}"'] & \mathcal{T}_m \end{tikzcd}$$

However, because $\alpha_m^*$ preserves finite limits, this is a valid interpretation of booleans in $\mathcal{C}[m]$. Next we consider the elimination principle. Applying Lemma 1.6.8, we obtain a lifting structure of the following type:

$$\alpha_m^*[\mathbf{tt}, \mathbf{ff}] \pitchfork \tau_m[-]$$

Again, however, because $\alpha_m^*$ preserves booleans, this is a valid elimination structure, completing the proof. $\square$

**Lemma 1.6.23.** *If $F$ supports identity types, then so does $\mathcal{C}[-]$.*

*Proof.* Identical to Lemma 1.6.22. $\square$

**Lemma 1.6.24.** *If $F$ supports a universe à la Coquand, then so does $\mathcal{C}[-]$.*

*Proof.* To begin with, we choose the presheaf for the universe of small types in $\mathbf{PSh}(\mathcal{C}[m])$ to be $\alpha_m^*\mathcal{S}_m$ and $\mathbf{lift} = \alpha_m^*\mathbf{lift}$. For clarity, we will write $\mathcal{S}'_m$ for the universe of small types in $\mathbf{PSh}(\mathcal{C}[m])$. We immediately obtain a code for $\mathcal{S}'_m$ by applying $\alpha_m^*$ to $\mathbf{Uni} : 1 \to \mathcal{T}_m$, which we denote $\mathbf{Uni}'$. Similarly, because $\alpha_m^*$ preserves pullbacks, we obtain an isomorphism between $\mathcal{S}'_m$ and $\tau_m^{-1}(\mathbf{Uni}')$.

It remains to show that this universe is closed under the appropriate connectives. All of the cases are similar, so we will show only the case of pi types in detail. We must show the following diagram can be filled in such a way that it commutes:

$$\begin{tikzcd} \sum_{A:[\![\text{🔒}_\mu]\!]^*\mathcal{S}'_n} \mathcal{S}'^{[\![\text{🔒}_\mu]\!]^*\tau_n^{-1}(\mathbf{lift}(A))}_m \arrow[r, dotted] \arrow[d] & \mathcal{S}'_m \arrow[d, "\mathbf{lift}"] \\ \sum_{A:[\![\text{🔒}_\mu]\!]^*\mathcal{T}_n} \mathcal{T}_m^{[\![\text{🔒}_\mu]\!]^*\tau_n^{-1}(A)} \arrow[r, "\prod"'] & \mathcal{T}_m \end{tikzcd}$$

To begin with, we paste in the naturality square which we used when constructing $\prod$ and unfold the definitions of $\mathcal{S}'_m$ and $\mathcal{T}_m$:

$$\begin{tikzcd}
\sum_{A:[\![\text{🔒}_\mu]\!]^*\mathcal{S}'_n} \mathcal{S}'^{[\![\text{🔒}_\mu]\!]^*\tau_n^{-1}(\mathbf{lift}(A))}_m \arrow[r, "\sim"] \arrow[d] & \alpha_m^*\left(\sum_{A:F(\mu)^*\mathcal{S}_n} \mathcal{S}_m^{F(\mu)^*\sigma_n^{-1}(\mathbf{lift}(A))}\right) \arrow[r] \arrow[d] & \alpha_m^*\mathcal{S}_m \arrow[d, "\alpha_m^*\mathbf{lift}"] \\
\sum_{A:[\![\text{🔒}_\mu]\!]^*\mathcal{T}_n} \mathcal{T}_m^{[\![\text{🔒}_\mu]\!]^*\tau_n^{-1}(A)} \arrow[r, "\sim"'] \arrow[rr, bend right, "\prod"'] & \alpha_m^*\left(\sum_{A:F(\mu)^*\mathcal{V}_n} \mathcal{V}_m^{F(\mu)^*\sigma_n^{-1}(A)}\right) \arrow[r] & \alpha_m^*\mathcal{V}_m
\end{tikzcd}$$

The right hand square is the image of the naturality square for $\alpha_m \circ [\![\text{🔒}_\mu]\!] \cong F(\mu) \circ \alpha_n$ together with the canonical isomorphism induced by Lemma 1.6.7 which was used in the construction of $\prod$ in Lemma 1.6.20. The right hand square is the commuting square witnessing that $\mathcal{S}_m$ is closed under pi types from Definition 1.6.17. Therefore, the composite of these two squares is the required commuting square and $\mathcal{S}'_m$ is closed under pi types. □

**Lemma 1.6.25.** *If $F$ supports modal types, then so does $\mathcal{C}[-]$.*

*Proof.* As in Lemmas 1.6.22 and 1.6.23, we interpret the formation rules by applying $\alpha_m^*$. Unlike with these two operations, we also must correct by the natural isomorphism $\alpha_m \circ [\![\text{🔒}_\mu]\!] \cong F(\mu) \circ \alpha_n$:

$$\begin{tikzcd}
{[\![\text{🔒}_\mu]\!]^*\widetilde{\mathcal{T}}_n} \arrow[r, "\sim"'] \arrow[d] \arrow[rr, bend left, "\mathbf{mod}_\mu"] & \alpha_m^*(F(\mu)^*\widetilde{\mathcal{V}}_n) \arrow[r] \arrow[d] & \widetilde{\mathcal{T}}_m \arrow[d] \\
{[\![\text{🔒}_\mu]\!]^*\mathcal{T}_n} \arrow[r, "\sim"] \arrow[rr, bend right, "\mathbf{Mod}_\mu"'] & \alpha_m^*(F(\mu)^*\mathcal{V}_n) \arrow[r] & \mathcal{T}_m
\end{tikzcd}$$

It remains to construct the lifting structure. To begin with, by applying $\alpha_m^*$ to $\mathbf{open}_\mu$ in $\mathbf{PSh}(F(m))$, we obtain a lifting structure for the following:

$$(\alpha_n \circ [\![\text{🔒}_\mu]\!])^* i \circ \alpha_m^* \iota_0 \pitchfork Z^*(\tau_m)$$

Now, using the fact that $\alpha_m^*\iota_0$ must remain an isomorphism and Lemma 1.6.10, we obtain a lifting structure for $(\alpha_n \circ [\![\text{🔒}_\mu]\!])^* i \pitchfork Z^*(\tau_m)$. It therefore suffices to show that $\alpha_n^* i$ is isomorphic in the arrow category to $i$ as defined in $\mathbf{PSh}(\mathcal{C}[m])$. This is a routine verification following the observation that $\alpha_n^*$ preserves pullback together with the naturality properties of $\alpha$.

First, recall the diagram defining $i$ in $\mathbf{PSh}(F(n))$:

$$\begin{tikzcd}
F(\nu)^*\tilde{\mathcal{V}}_o \arrow[dr, dashed, "i"] \arrow[drr, bend left, "\mathbf{mod}_\nu"] \arrow[ddr, bend right, "F(\nu)^*\sigma_o"'] & & \\
& M \arrow[r] \arrow[d, "h"] \arrow[dr, phantom, "\lrcorner", very near start] & \tilde{\mathcal{V}}_n \arrow[d, "\sigma_n"] \\
& F(\nu)^*\mathcal{V}_o \arrow[r, "\mathbf{Mod}_\nu"'] & \mathcal{V}_n
\end{tikzcd} \tag{1.18}$$

Next, we apply $\alpha_n^*$, we then obtain the following diagram in $\mathbf{PSh}(\mathcal{C}[n])$:

$$\begin{tikzcd}
\alpha_n^*(F(\nu)^*\tilde{\mathcal{V}}_o) \arrow[dr, dashed, "\alpha_n^*(i)"] \arrow[drr, bend left, "\alpha_n^*(\mathbf{mod}_\nu)"] \arrow[ddr, bend right, "\alpha_n^*(F(\nu)^*\sigma_o)"'] & & \\
& \alpha_n^*(M) \arrow[r] \arrow[d, "\alpha_n^*(h)"] \arrow[dr, phantom, "\lrcorner", very near start] & \alpha_n^*(\tilde{\mathcal{V}}_n) \arrow[d, "\alpha_n^*(\sigma_n)"] \\
& \alpha_n^*(F(\nu)^*\mathcal{V}_o) \arrow[r, "\alpha_n^*(\mathbf{Mod}_\nu)"'] & \alpha_n^*(\mathcal{V}_n)
\end{tikzcd} \tag{1.19}$$

This remains a pullback because $\alpha_n^*$ preserves pullbacks. We now apply the naturality of $\alpha_n$ to obtain the following (isomorphic) diagram:

$$\begin{tikzcd}
[\![\text{🔒}_\nu]\!]^*\widetilde{\mathcal{T}}_o \arrow[dr, dashed, "i"] \arrow[drr, bend left, "\mathbf{mod}_\nu"] \arrow[ddr, bend right, "[\![\text{🔒}_\nu]\!]^*\tau_o"'] & & \\
& M \arrow[r] \arrow[d, "h"] \arrow[dr, phantom, "\lrcorner", very near start] & \mathcal{T}_n \arrow[d, "\mathcal{T}_n"] \\
& [\![\text{🔒}_\nu]\!]^*\mathcal{T}_o \arrow[r, "\mathbf{Mod}_\nu"'] & \mathcal{T}_n
\end{tikzcd} \tag{1.20}$$

The universality of pullbacks ensure that we do indeed obtain $i$ and $M$ when we apply naturality. This, in turn, gives $i \cong \alpha_n^*(i)$ as required. $\square$

**Theorem 1.6.26.** *Given a pseudo-functor $F : \mathcal{M}^{\mathsf{coop}} \to \mathbf{Cat}$, such that $F(m)$ is a weak model of MTT, there exists a 2-functor $F'$, such that there is a pseudo-natural transformation $F(m) \simeq F'(m)$ and such that $F'$ is a model of MTT with the mode theory $\mathcal{M}$.*

*Proof.* Immediate from the prior lemmas. □

## 1.7 *Additions and Modifications to MTT*

In order to better accommodate the study of certain applications, we might wish to extend MTT with a new base type, or to change some of the rules in order to accord with circumstance. In some cases this will enable an 'apples-to-apples' comparison with existing modal type theories. For example, to faciliate the comparison with the extensional guarded dependent type theory of Bizjak et al. [Biz+16], we ought to replace intensional equality with extensional equality.

Such extensions can be problematic, as they often disrupt the metatheory, or litter the syntax with irrelevant details. Nevertheless, we devote this section to a brief discussion of a number of possibilities of which we will make use later. We refrain however from extending Theorems 1.4.11 and 1.5.7 to cover these extensions and alterations.

### 1.7.1 *Extensional Equality*

We may remove the $\mathsf{Id}_A(M, N)$ type and all the rules associated with it, and replace them with

$$\frac{\Gamma\ \mathsf{ctx} @ m \qquad \Gamma \vdash A\ \mathsf{type}_\ell @ m \qquad \Gamma \vdash M_0, M_1 : {\Uparrow}A @ m}{\Gamma \vdash \mathsf{Eq}_A(M_0, M_1)\ \mathsf{type}_\ell @ m}$$

$$\frac{\Gamma\ \mathsf{ctx} @ m \qquad \Gamma \vdash A\ \mathsf{type}_1 @ m \qquad \Gamma \vdash M : A @ m}{\Gamma \vdash \mathsf{refl}(M) : \mathsf{Eq}_A(M, M) @ m}$$

$$\frac{\Gamma\ \mathsf{ctx} @ m \qquad \Gamma \vdash A\ \mathsf{type}_1 @ m \qquad \Gamma \vdash M_0, M_1 : A @ m \qquad \Gamma \vdash P : \mathsf{Eq}_A(M_0, M_1) @ m}{\Gamma \vdash M_0 = M_1 : A @ m}$$

The model must change as well: following Awodey [Awo18], we ask that the formation and introduction rules form a pullback square

$$\begin{tikzcd}
\sum_{A:\mathcal{T}_m} \tau_m^{-1}(A) \arrow[r, "\mathbf{refl}"] \arrow[d] \arrow[dr, phantom, "\lrcorner", very near start] & \widetilde{\mathcal{T}}_m \arrow[d, "\tau_m"] \\
\sum_{A:\mathcal{T}_m} \tau_m^{-1}(A) \times \tau_m^{-1}(A) \arrow[r, "\mathbf{Eq}"'] & \mathcal{T}_m
\end{tikzcd}$$

### 1.7.2 *Natural Numbers*

These rules are standard:

$$\frac{\Gamma\ \mathsf{ctx} @ m}{\Gamma \vdash \mathsf{Nat}\ \mathsf{type}_\ell @ m} \qquad \frac{\Gamma\ \mathsf{ctx} @ m}{\Gamma \vdash \mathsf{zero} : \mathsf{Nat} @ m} \qquad \frac{\Gamma\ \mathsf{ctx} @ m \qquad \Gamma \vdash M : \mathsf{Nat} @ m}{\Gamma \vdash \mathsf{succ}(M) : \mathsf{Nat} @ m}$$

$$\frac{\begin{array}{ccc} & \Gamma\ \mathsf{ctx} @ m \qquad \Gamma.(1 \mid \mathsf{Nat}) \vdash A\ \mathsf{type}_1 @ m & \\ \Gamma \vdash M_z : A[\mathsf{id}.\mathsf{zero}] @ m & \Gamma.(1 \mid \mathsf{Nat}).(1 \mid A) \vdash M_s : A[\uparrow^2.\mathsf{succ}(\mathbf{v}_1)] @ m & \Gamma \vdash N : \mathsf{Nat} @ m \end{array}}{\Gamma \vdash \mathsf{rec}(A; M_z; M_s; N) : A[\mathsf{id}.N] @ m}$$

The introduction and formation rules are similar to those for booleans: we require two commuting diagrams

$$\begin{tikzcd} & \widetilde{\mathcal{T}}_m \arrow[d, "\tau_m"] \\ 1 \arrow[ur, "\mathbf{zero}"] \arrow[r, "\mathbf{Nat}"'] & \mathcal{T}_m \end{tikzcd} \qquad \begin{tikzcd} & \widetilde{\mathcal{T}}_m \arrow[d, "\tau_m"] \\ \tau_m^{-1}(\mathbf{Nat}) \arrow[ur, "\mathbf{succ}"] \arrow[r] & \mathcal{T}_m \end{tikzcd}$$

Modelling the elimination rule is more complex. We cannot simply use a lifting structure, because there is a recursive component to natural numbers. This is evident in the inductive premise of the elimination rule, as the inductive case $M_s$ depends on the motive $A$. It is therefore impossible to directly express the interpretation of natural numbers as the solution to a lifting problem.

There are a few ways to resolve this. One is to directly use the internal language to write out the data required to model the elimination rule:

$$\mathbf{rec} : \textstyle\prod_{A:\tau_m^{-1}(\mathbf{Nat})\to\mathcal{T}_m} A(\mathbf{zero}) \to (\prod_{n:\tau_m^{-1}(\mathbf{Nat})} A(n) \to A(\mathsf{succ}(n))) \to \prod_{n:\tau_m^{-1}(\mathbf{Nat})} A(n)$$

Whilst correct, this is merely an unsatisfying restatement of the problem.

A more informative solution may be found by tackling the problem of interpreting more general inductive types. Recall that many such types can be captured as initial algebras of endofunctors. Suppose we are given

$$T : \mathbf{PSh}(\mathcal{C}[m]) \to \mathbf{PSh}(\mathcal{C}[m])$$

We ask that $T$ be *enriched*, which ensures that there is a natural map $Y^X \to T(Y)^{T(X)}$ that implements the functorial action of $T$ internally in $\mathbf{PSh}(\mathcal{C}[m])$, and thus enables the use of $T$ in the internal language.[7] Let us further suppose we are given a $T$-algebra for some closed type $A : 1 \to \mathcal{T}_m$, i.e. a morphism

$$\alpha : T(\tau_m^{-1}(A)) \to \tau_m^{-1}(A)$$

Using $\alpha$ we can restrict $T$ to the slice category $\mathbf{PSh}(\mathcal{C}[m])/\tau_m^{-1}(A)$ by sending

$$X \xrightarrow{x} \tau_m^{-1}(A) \;\longmapsto\; T(X) \xrightarrow{T(x)} T(\tau_m^{-1}(A)) \xrightarrow{\alpha} \tau_m^{-1}(A)$$

We may then see $\alpha$ itself in the slice category as a morphism $\alpha : T(1) \to 1$, i.e. a $T$-algebra on the terminal object of $\mathbf{PSh}(\mathcal{C}[m])/\tau_m^{-1}(A)$, and also the unique such map to the terminal object. We may then ask that this map be *naturally weakly initial* among $T$-algebras fibred over $A$, i.e. that there exists a fibred $T$-homomorphism from $\alpha$ to any $T$-algebra. Expressed in the internal language of the slice category $\mathbf{PSh}(\mathcal{C}[m])/\tau_m^{-1}(A)$, this amounts to a structure

$$\textstyle\prod_{C:\mathcal{T}_m} \prod_{c:T(\tau_m^{-1}(C))\to\tau_m^{-1}(C)} \{h : 1 \to \tau_m^{-1}(C) \mid h \circ \alpha = c \circ T(h)\}$$

This generalizes Definition 1.4.7 (take $T$ to be a constant functor), but may also be used to capture many other inductive types. For example, natural numbers are obtained by picking $T(X) \triangleq 1 + X$ and $A \triangleq \mathbf{Nat} : 1 \to \mathcal{T}_m$.

[7]Recall that $\mathbf{PSh}(\mathcal{C}[m])$ is trivially 'self-enriched,' as it has exponentials.

## 1.8 Related Work

Modal type theory has been an active area of research for several decades and, as with any active field, a precise taxonomy of modal type theories would be a paper in and of itself. Accordingly, we have not attempted such a task here, and have chosen instead to focus on separating modal type theories into distinct strands based on the judgmental structures underlying them. Our characterization is slightly artificial in some cases, and these lines of work are not nearly so separate as our description might suggest. We feel, however, that this is the simplest way to place MTT in relation to the current work surrounding the interactions of modalities and dependent types.

### 1.8.1 Dual-Context Modal Calculi

One of the first papers on modal type theory was by Pfenning and Davies [PD01],[8] which constructed a proof theory for S4, i.e. a comonadic modality. The central idea of this approach was to reflect the distinction between the truth and validity of a proposition in the judgmental structure of the system itself, rather than attempting to construct it after the fact. The judgments for this calculus then contained not just a context of true propositions, but rather two contexts: one for true propositions, and one for valid propositions. By structuring the system around this distinction from the beginning, incorporating a $\Box$ comonad is straightforward; $\Box A$ just internalizes an artifact already present in the system:

$$\frac{\Delta;\cdot\vdash A\ \mathsf{true}}{\Delta;\Gamma\vdash\Box A\ \mathsf{true}}\qquad\frac{A\in\Delta\cup\Gamma}{\Delta;\Gamma\vdash A\ \mathsf{true}}\qquad\frac{\Delta;\Gamma\vdash\Box A\ \mathsf{true}\qquad\Delta,A;\Gamma\vdash B\ \mathsf{true}}{\Delta;\Gamma\vdash B\ \mathsf{true}}$$

It was apparent from the beginning that this proof theory could be fruitfully interpreted as a type theory, and Pfenning and Davies [PD01] already begin to develop the metatheory of such a system. Other work picked up where Pfenning and Davies [PD01] had left off and began to develop the theory of dual-context lambda calculi. Recently, Kavvos [Kav17] presented a unified picture of several different modal logics into this proof theory. On the type-theoretic side, it has long been believed that the dual-context calculus should generalize to support full dependent types. This generalization is reflected in both de Paiva and Ritter [dR15] and Shulman [Shu18]. Similarly, contextual modal type theory [NPP08; BP11; BS15; Pie+19] has used a dual-context-like structure in order to give a systematic account of higher-order abstract syntax.

Recent work by Zwanziger [Zwa19] continues this program by formulating a precise categorical semantics based on natural models [Awo18] for a dependent type theory with either an adjunction (**AdjTT**) or comonad (**CoTT**). The categorical semantics of MTT and **AdjTT** are closely related, though with minor differences in the precise definition of the modality. For instance, in MTT only the 🔒 operator is required to act upon the context, while in **AdjTT** the modalities themselves must extend to contexts.[9] These differences arise because Zwanziger [Zwa19] characterizes only a certain, semantically well-behaved, subclass of models, while Section 1.4 describes models which capture not only these situations, but the syntactic model as well as the gluing model from Theorem 1.5.9. Syntactically, **AdjTT** is multimode type theory, including a mode for "both ends" of the adjunction, but it is not multimodal and allows for only one adjunction.

The limitation of this dual-context style is its lack of generality. As the complexity of the modal situation increases, the complexity of the context structure must increase. Moreover, this increased complexity is not linear in the number of distinct modalities, and it quickly becomes unmanageable. Moreover, the structure of a dependent dual-context type theory enforces that a valid type (one belonging to $\Delta$) may not depend on a true type (one belonging to $\Gamma$). This is a reasonable enough restriction

[8] Unfortunately, even this first statement is gross revisionism; the idea of dual-contexts was present well before 2001 [And92; Gir93; Plo93].

[9] This is similar to the relation between a CwF+A and a CwDRA from Birkedal et al. [Bir+20], and we expect a similar relation to exist between the semantics of MTT with a single modality and **AdjTT**.

in the case of $\Box$, but it is already somewhat limiting. For instance, it should be allowed for a valid type to depend on a merely true type if that type is $\Box A$, or equivalent to one of this shape. Making such an adjustment would not only present a typographical problem (with a type occurring to the left of one of its dependencies), it would render the introduction rule for $\Box A$ nonsensical.

This restriction proves even more difficult to manage once there is not merely one modality, but two distinct modalities within the system, say $\mu$ and $\nu$. Should the $\mu$-modified types be allowed to depend on $\nu$-modified types? Vice-versa? Should there be three contexts: $\mu$-modified, $\nu$-modified, unmodified? Or perhaps a fourth, $\mu\nu$-modified? These questions can be addressed for each specific modal situation, indeed both Shulman [Shu18] and Zwanziger [Zwa19] both hand-craft a system for two modalities, but these systems must be constructed for each case specifically.

Indeed, there is very little to complain about for any given dual-context calculi. Many of these type theories satisfy desirable meta-theoretic properties, well-defined semantics, and are reasonable for programming [Vez18]. What is lacking with the dual-context style is the ability to work systematically with a large class of modal situations without reconsidering the properties of the system in each case. Some of MTT's rules can be directly traced to rules in dual-context calculi (in particular, the elimination rule for modal types) but the structure of the context is radically different in order support a wide variety of modal situations out of the box.

### 1.8.2 Modal Type Theories based on Left-Division

Developed concurrently with the dual-context calculi has been a series of modal type theories based on what Nuyts, Vezzosi, and Devriese [NVD17] called "left division". Under this discipline, rather than having a fixed set of contexts, there is a single context consisting of variables annotated with modal annotations. We trace this structure to Pfenning [Pfe01], where the annotations described the relevance of the variable. For instance, a variable could be tagged as *irrelevant*, which could only be used in other irrelevant positions and could be skipped over when checking terms for conversion.

In a non-dependent type system, the distinction between annotations and different contexts is a little artificial: we could simply sort variables by their annotation and obtain different context zones. Once generalized to a dependent type theory, annotated contexts do not require the same fixed discipline for which zones may depend on others. Instead, a type may depend on anything prior in the context, but the nature of that dependence is moderated through their annotations.

The term "left division" is chosen to describe this structure because of the behavior of the introduction rules for modal types. For instance, in Pfenning [Pfe01], there is a rule for introducing a term in an irrelevant context:

$$\frac{\Gamma^{\oplus} \vdash M : A}{\Gamma \vdash M :_{\mathsf{irr}} A}$$

Here $-^{\oplus}$ is a metaoperation defined by traversing the context and modifying the annotations by removing irrelevant annotations. The effect of this is that all the variables in $\Gamma^{\oplus}$ can be used freely, which is sufficient when type-checking $M$ because $M$ itself is irrelevant. If one is sufficiently careful, one can construe this operation as a division operation, we divide all the annotations in $\Gamma$ by $\mathsf{irr}$. The metatheory of a full dependent type theory based on this idea was considered by Abel and Scherer [AS12], which ensures the soundness and decidability of modeling irrelevance in this way.

More recently, Nuyts, Vezzosi, and Devriese [NVD17], and especially Nuyts and Devriese [ND18] have carried this idea to its natural conclusion and incorporated modalities for more modes of use than "irrelevant", "relevant", or "intensional". By including these extra modalities, in addition to modeling irrelevance Nuyts and Devriese [ND18] can prove and internalize the identity extension lemma of relational parametricity [Rey83] even for large types. We will explore the relation between MTT and Nuyts and Devriese [ND18] in greater detail (Section 2.4), but for the moment we content ourselves with discussing the relationship between this left-division style and MTT.

In a related but distinct line of work, the `Granule` Project [Gab+16; OLE19] has exploited a similar structure to give a systematic account of substructurality. There is ongoing work to extend this to a full dependent type theory.

The structure of MTT's contexts is closely related to the contexts of these calculi. Contrasting MTT with Pfenning [Pfe01] in particular, we find that in both of these calculi the contexts are generated by different sorts of variables. For instance, Pfenning [Pfe01] has normal variables (written $x : A$), irrelevant variables ($x \div A$), and valid variables ($x :: A$). Each sort of context entry acts a different modal modifier: $x \div A$ marks $x$ as irrelevant, and prevents us from using it as a normal term and on the opposite end, $x :: A$ marks a variable as treated "intensionally". MTT could accommodate this setup with a single mode that has three endomodalities: irrelevance, extensionality (the identity modality), and intensionality. A composition table for these modalities can be built by sorting out what is the strongest modality under which the composite function can be defined in Pfenning's calculus.

The rules for interacting with the modalities in Pfenning's system traverse the context and modify the binding used for each variable. Suppose, for example, we are typing $M(N)$, where $M$ is a function marked as irrelevant in its first argument, we then typecheck $N$ in a modified context in which all variables $x \div A$ have been replaced with $x : A$. Dually, if $M$ as tagged as being intensional in its argument we replace all the occurrences of $x : A$ with $x \div A$, ensuring that we do not depend on variables which are not themselves intensional.

This bulk operation is different than MTT-style locks, but amounts to the same constraints on variable use. By tagging the context with a lock, every time we use a variable we must ensure that the modality tagging the variable is sufficiently strong. When we bulk-update the context, the same restrictions occur but they are done "eagerly".

The use of "lazy" locks has several advantages over eager bulk updates:

- We do not have to explain what it means to divide one modality by another,
- We can allow non-trivial 2-cells: with the bulk-update strategy there is no place for the user to specify how a variable is extracted from under the modality,
- When interpreting the calculus in a model, we do not have to sort out how a variable by variable modality update affects the interpretation of the entire context (which Nuyts [Nuy18a] found to be a somewhat painful endeavour).

We remark that admissibility of the left division operation on both contexts and substitutions is not that hard to prove for a general mode theory for which left division $(\mu \setminus -)$ is defined on modalities and is left adjoint [Abe06; Abe08] to postcomposition $(\mu \circ -)$. This is in contrast to certain lock or variable removal operations necessary in Fitch-style approaches (Section 1.8.3).

### 1.8.3 Modal Type Theories Based on The Fitch Style

A recent series of paper [BGM17; Bir+20; GSB19] have used a similar judgmental structure to manage a variety of modalities. This judgmental structure, often informally referred to as the "Fitch-style" [Clo18], divides the context into regions of variables separated by locks. Locks are dynamically included or removed throughout the typing derivation of a term.

The central advantage of the Fitch-style is the impressively simple introduction rule for modalities: whenever we wish to introduce a modality we simply append a lock to the context to tag the modal shift and continue. We do not, in particular, ever need to remove variables from the context during the introduction of a modal term, which alleviates a significant sore point of the dual-context calculi. Of course, this style of rule is only sound for a modality which comes equipped with some sort of left adjoint, but this restriction is also present in MTT.

Another desirable property of the Fitch-style calculi are their strong elimination rules for modalities. Rather than the pattern matching-style rules of other systems, Fitch-style calculi have had an *open-scope* elimination rule for their modalities. The stronger rule also often permits a definitional $\eta$-rule for $\Box A$. The elimination rule is generally of the following shape:

$$\frac{F(\Gamma) \vdash M : \Box A}{\Gamma \vdash \mathsf{open}(M) : A}$$

In this rule, $F$ is a meta-theoretic operation on contexts which removes some number of locks and variables from $\Gamma$. For instance, in Birkedal et al. [Bir+20], $F(\Gamma)$ was defined as follows:

$$F(\Gamma, \text{🔓}, \Gamma') = \Gamma \text{ where } \text{🔓} \notin \Gamma'$$

This rule is convenient, and also strictly more powerful than the elimination rule in MTT (see Sections 1.4.2 and 2.6) but rules which remove elements of the context are traditionally problematic in type theory. The source of the trouble in this case is that we must show that substitutions can be commuted past open. For instance, suppose we have some substitution $\gamma : \Delta \to \Gamma, \text{🔓}, \Gamma'$. It is necessary to ensure that this substitution uniquely gives rise to a substitution $F(\gamma) : F(\Delta) \to \Gamma$, and this is not at all guaranteed. For instance, in Birkedal et al. [Bir+20], an induction over the structure of the substitutions is needed to produce $\gamma$, and it cannot be done without knowing both $\Delta$ and $\Gamma'$ in advance. In Gratzer, Sterling, and Birkedal [GSB19], it is laboriously proven that if $\gamma : \Delta \to \Gamma$, then $\gamma : F(\Delta) \to F(\Gamma)$, but at the expense of several complex and artificial typing rules. The situation is in some ways similar to dual-context calculi, where each modal situation requires expert attention in order to show that the elimination rule is syntactically well-behaved.

The other, more serious, issue with the Fitch-style is the difficulty of accounting for multiple distinct modalities. Intuitively, each modality should give rise to a different lock but the structural rules governing their interactions are complex even in relatively simple cases. For instance, it is well-understood how to model the $\blacktriangleright$ modality in a Fitch-style type theory, and Gratzer, Sterling, and Birkedal [GSB19] developed an extensive account of the $\Box$ modality, but it is exceptionally difficult to combine the two. There is work to this effect in a simple type theory [BGM19], but even in this case there are restrictions on $\Box$ and $\blacktriangleright$ which prevent recovering, e.g. Gratzer, Sterling, and Birkedal [GSB19] as a subsystem.

The root of the issue seems to mirror the problems in Section 1.8.1, having a rule which removes elements of the context is difficult to account for in more complex situations. Drawing on this intuition, MTT has adopted the simple introduction rules from Fitch-style calculi, but not the elimination rules. The result is that MTT has a less powerful elimination rule, and a weaker definitional equality than Fitch-style calculi. In particular, there is no equivalent of the definitional $\eta$-equality. In return for these weaker rules, MTT can smoothly incorporate multiple interacting modalities.

### *1.8.4 Other General Modal Frameworks*

A recent line of work [LS16; LSR17] has tackled the same question as MTT: how to design one calculus which can be systematically adapted to many different modal situations. Currently, there is ongoing work to extend Licata, Shulman, and Riley [LSR17] to a full dependent type theory, as of April 2020 this work remains unpublished.

In fact, this line of research (which we refer to as LSR after the authors), is more general still. LSR is designed to handle a wide variety of modal situations *as well as* a variety of different structural principles. This is an axis of generalization entirely outside the scope of MTT, promising to address the some of the major shortcomings in the interaction between dependent types and substructural logics. Owing, however, to the fact that MTT is a firmly structural type theory, we will focus on the modal aspects of LSR.

The very idea of parametrizing a type theory by a mode theory, as we have done with MTT, originates with LSR [LS16]. Indeed, the modal situations that can be handled by MTT are a strict subset of those which can be handled by LSR; LSR not only includes a modal connective for the right adjoint described in the mode theory, but the left adjoint as well. For instance, in Section 2.8, we will discuss how MTT can model an adjunction of modalities, but this situation is limited to the case where the left adjoint has a further left adjoint. LSR has no such restrictions, and can freely talk about arbitrary adjunctions inside the type theory.

The contribution of MTT is not increased generality of LSR. Instead, we have focused on ensuring that MTT is a simpler type theory which still accounts for some of the interesting modal situations that LSR handles. In particular, by explicitly avoiding substructurality, MTT has a simpler syntax which is amenable to current proof and implementation techniques. This is reflected in our proof of canonicity (Theorem 1.5.9), and our experimental implementation efforts [Nuy19]. We therefore believe that MTT is a natural halfway point between current modal type theories (which are custom-fitted for each modal situation) and the full generality of LSR.

# 2 *Applications of MTT*

The slogan is "Adjoint functors arise everywhere."

Saunders Mac Lane
*Categories for the Working Mathematician*

Thus far we have only studied properties of MTT that remain invariant under a change of mode theory. These general theorems constitute a strong justification of our programme, as we do not need to repeat a proof of—say—canonicity for each modal situation that MTT is called to express. Instead, we prove it once and for all for an arbitrary mode theory. Likewise, we have no need to change the syntax, or to describe a specialised kind of model each time we want to study a new modality.

Such general theorems come at a price: this good behaviour is contingent on our limiting the modal expressivitity of our system. In particular, our modal types are a form of weak dependent right adjoint, and more general modalities—such as arbitrary functors or left adjoints—are beyond our reach. The purpose of this chapter is to demonstrate that this restriction is not as severe as it might first appear. By working through a series of important examples, we argue that MTT can be used to reason about a variety of modal situations. Moreover, we argue that programming in MTT is not just tractable, but also a simpler alternative to many existing modal type theories.

## 2.1 Constructing Dependent Right Adjoints

Most of our examples of models of MTT will factor through Theorem 1.4.11: we will first construct models of Martin-Löf Type Theory (MLTT) at each mode $m$, and then relate them by presenting dependent right adjoints (DRAs) between them. In many cases we will work with well-known models of MLTT—e.g. presheaf categories—so that the only hard work will pertain to the construction of the relevant DRAs.

In this section we present a general result that allows us to lift a general right adjoint to a dependent right adjoint. This lemma, versions of which have appeared before in the work of Birkedal et al. [Bir+20] and Nuyts [Nuy18a], will simplify many of our later constructions. We will demonstrate its use by proving that the adjunctions between left Kan extension, precomposition, and right Kan extension induce a DRA structure on the latter two.

**Remark 2.1.1.** Recall that a dependent right adjoint is stronger than what is required for a model of MTT: we only need an action on terms and types, and an appropriate lifting structure. However, all the models we actually consider come with a full dependent right adjoint, and the weaker concept appears to be an artifact of the syntax. ◁

Recall from Section 1.4 and Definition 1.4.13 the notion of *morphism of natural models*. Using the same notation we define the following weaker notion, for which see also [New18, §§2.3.9].

**Definition 2.1.2.** A *weak morphism of natural models* $(\mathcal{C}, \mathcal{T}_c) \to (\mathcal{D}, \mathcal{T}_d)$ consists of a functor $F : \mathcal{C} \to \mathcal{D}$, and a commuting square

$$\begin{tikzcd} \widetilde{\mathcal{T}}_c \arrow[r, "\widetilde{\varphi}"] \arrow[d, "\tau_c"'] & F^*\widetilde{\mathcal{T}}_d \arrow[d, "F^*\tau_d"] \\ \mathcal{T}_c \arrow[r, "\varphi"'] & F^*\mathcal{T}_d \end{tikzcd}$$

such that $F(\mathbf{1}) = \mathbf{1}$ and the canonical morphism $F(\Gamma.A) \to F\Gamma.\varphi(A)$ is an isomorphism. We say that this morphism of natural models *preserves size* whenever there is a commuting square

$$\begin{tikzcd} \mathcal{S}_c \arrow[r, "\mathcal{S}_\varphi"] \arrow[d, "\mathbf{lift}"'] & F^*\mathcal{S}_d \arrow[d, "F^*\mathbf{lift}"] \\ \mathcal{T}_c \arrow[r, "\varphi"'] & F^*\mathcal{T}_d \end{tikzcd}$$

We show that a right adjoint that is also a morphism of natural models can be lifted to a DRA.

**Lemma 2.1.3.** *Suppose that $(\mathcal{C}, \tau_{\mathcal{C}})$ and $(\mathcal{D}, \tau_{\mathcal{D}})$ are natural models, and that $L \dashv R$ is an adjunction between $\mathcal{C}$ and $\mathcal{D}$. If the right adjoint $R : \mathcal{C} \to \mathcal{D}$ extends to a weak morphism of natural models then it gives rise to a dependent right adjoint. Moreover, the resulting DRA is size-preserving whenever $R$ is.*

*Proof.* We first fix some notation: we write $L : \mathcal{D} \to \mathcal{C}$ for the left adjoint, and $\eta : \mathsf{Id} \Rightarrow RL$ for the unit of the adjunction $L \dashv R$. We assume a commuting square

$$\begin{tikzcd} \widetilde{\mathcal{T}_{\mathcal{C}}} \arrow[r, "\mathsf{r}"] \arrow[d, "\tau_{\mathcal{C}}"'] & R^*\widetilde{\mathcal{T}_{\mathcal{D}}} \arrow[d, "R^*\tau_{\mathcal{D}}"] \\ \mathcal{T}_{\mathcal{C}} \arrow[r, "\mathsf{R}"'] & R^*\mathcal{T}_{\mathcal{D}} \end{tikzcd} \tag{2.1}$$

that witnesses the weak natural model morphism structure of $R$, and write

$$\nu_{\Gamma,A} : R\Gamma.\mathsf{R}(A) \xrightarrow{\cong} R(\Gamma.A)$$

for the canonical isomorphism corresponding to $\lfloor A \rfloor : \mathbf{y}(\Gamma) \Rightarrow \mathcal{T}_{\mathcal{C}}$. We then define the square

$$\begin{tikzcd} L^*\widetilde{\mathcal{T}_{\mathcal{C}}} \arrow[r, "L^*\mathsf{r}"] \arrow[d, "L^*\tau_{\mathcal{C}}"'] & L^*R^*\widetilde{\mathcal{T}_{\mathcal{D}}} \arrow[r, "\eta^*_{\widetilde{\mathcal{T}_{\mathcal{D}}}}"] \arrow[d, "L^*R^*\tau_{\mathcal{D}}"] & \widetilde{\mathcal{T}_{\mathcal{D}}} \arrow[d, "\tau_{\mathcal{D}}"] \\ L^*\mathcal{T}_{\mathcal{C}} \arrow[r, "L^*\mathsf{R}"'] & L^*R^*\mathcal{T}_{\mathcal{D}} \arrow[r, "\eta^*_{\mathcal{T}_{\mathcal{D}}}"'] & \mathcal{T}_{\mathcal{D}} \end{tikzcd}$$

The left part of the square is the image of (2.1) under the $L^*$ functor, and the right part is a naturality square for the natural transformation $\eta^* : L^*R^* = (RL)^* \Rightarrow \mathsf{Id}$ induced by the unit of the adjunction.

We must show that this defines a pullback, and it suffices to do so on the representables. Assume we have $\lfloor A \rfloor : \mathbf{y}(\Delta) \Rightarrow L^*\mathcal{T}_{\mathcal{C}}$ and a $\lfloor M \rfloor : \mathbf{y}(\Delta) \Rightarrow \mathcal{T}_{\mathcal{D}}$ such that the diagram commutes. Switching to type-theoretic notation, this amounts to a type $L(\Delta) \vdash A$ type—which gives rise to a type $R(L(\Delta)) \vdash \mathsf{R}(A)$ type by applying $\mathsf{R}$—and a term $\Delta \vdash M : \mathsf{R}(A)[\eta_\Delta]$. The universal property of the pullback dictates that we must show the existence of a unique term $L(\Delta) \vdash N : A$ such that

$$RL(\Delta) \vdash \mathsf{r}(N)[\eta_\Delta] = M : \mathsf{R}(A)[\eta_\Delta] \tag{2.2}$$

We do this as follows. First, observe that we can form the substitution $\eta_\Delta.M : \Delta \to RL\Delta.\mathsf{R}(A)$. We can then postcompose the isomorphism $\nu_{\Delta,A}$ to obtain a morphism of type $\Delta \to R(L\Delta.A)$. To this we can apply $L$ and postcompose the counit $\epsilon_{L\Delta.A}$ to obtain a substitution

$$k \triangleq \epsilon_{L\Delta.A} \circ L(\nu_{\Delta,A} \circ \eta_\Delta.M) : L\Delta \to L\Delta.A$$

Using naturality of the counit and the equations satisfied by the canonical isomorphism $\nu_{\Delta,A}$, it is easy to show that $\mathbf{p} \circ k = \mathsf{id} : L\Delta \to L\Delta$, and hence that we can extract a term

$$L(\Delta) \vdash N \triangleq \mathbf{q}[k] : A$$

from $k$. Using naturality of $\mathsf{r}(-)$, naturality of the unit, and the one of the triangle identities for the adjunction, we can then calculate that this satisfies equation (2.2). Finally, we can prove this choice is unique by calculating that any such $N$ necessarily implies that $k = \mathsf{id}.N$, and hence that $\mathbf{q}[k] = N$.

It is routine to show that this is size-preserving, using the fact that $\mathsf{R}$ preserves size. □

**Lemma 2.1.4.** *Given a functor $\mu : \mathcal{C} \to \mathcal{D}$, the precomposition functor $\mu^* : \mathbf{PSh}(\mathcal{D}) \to \mathbf{PSh}(\mathcal{C})$ induces a dependent right adjoint. Moreover, $\mu^*$ is size-preserving for any Grothendieck universe.*

*Proof.* It is well-known that this functor has a left adjoint (via Kan extension). It therefore suffices to show that it satisfies the other requirements of a dependent right adjoint, namely that it induces a natural action on types and terms and preserves context extension and the empty context up to isomorphism.

Let us recall that the standard CwF structure on $\mathbf{PSh}(\mathcal{D})$ defines types in context $\Gamma$ as the objects $\mathbf{PSh}(\int \Gamma)$ and the terms as the global sections of these objects. We can lift the action of $\mu^*$ to these terms as follows:

$$\mu^* A \in \mathbf{PSh}(\textstyle\int \mu^*\Gamma) = (C : \mathcal{C}, a \in \Gamma(\mu(C))) \mapsto A(\mu(C), a)$$
$$\mu^* M \in \mathrm{Hom}(1, \mu^* A) = (C : \mathcal{C}, a \in \Gamma(\mu(C))) \mapsto M(\mu(C), a)$$

It is routine to check that these are functorial (respectively natural) using the fact that $A$ and $M$ both satisfy the corresponding condition. In order to show that these commute with substitution, we use the functoriality of $\mu$. For instance, suppose that we have $\delta : \Delta \to \Gamma$ and $A \in \mathbf{PSh}(\int \Gamma)$.

$$\begin{aligned}
(\mu^* A)[\mu^*\delta] &= ((C, a) \mapsto A(\mu(C), a))[\mu^*\delta] \\
&= (C, a) \mapsto A(\mu(C), (\mu^*\delta)_C(a)) \\
&= (C, a) \mapsto A(\mu(C), \delta_{\mu(C)}(a)) \\
&= (C, a) \mapsto (A[\delta])(\mu(C), a) \\
&= \mu^*(A[\delta])
\end{aligned}$$

The proof for terms is similar.

The preservation of the terminal context is immediate: a terminal object is always preserved by a right adjoint. Context extension is preserved up to isomorphism by a simple calculation:

$$\begin{aligned}
\mu^*(\Gamma.A) &= \mu^*(D \mapsto (\gamma \in \Gamma(D)) \times A(D, \gamma)) \\
&= C \mapsto (\gamma \in \mu^*\Gamma(C)) \times A(\mu(C), \gamma) \\
&\cong \mu^*\Gamma.\mu^* A
\end{aligned}$$ □

In order to show size preservation, we recall that $A$ is said to be $\mathcal{V}$-small if each fiber of $A$ is $\mathcal{V}$-small. Since, however, the fibers of $\mu^* A$ are a subset of those of $A$, it is immediate that $\mu^* A$ is $\mathcal{V}$-small whenever $A$ satisfies this condition.

**Lemma 2.1.5.** *Given a functor $\mu : \mathcal{C} \to \mathcal{D}$, the right adjoint to precomposition, $\mu_* : \mathbf{PSh}(\mathcal{C}) \to \mathbf{PSh}(\mathcal{D})$ induces a dependent right adjoint. Moreover, $\mu_*$ is size-preserving for any Grothendieck universe.*

*Proof.* At this point we have a left adjoint to $\mu_*$, namely $\mu^*$, so it again suffices to show that this functor lifts to a natural action on types and terms and that the functor respects context extension.

We define the lifting of $\mu_*$ as follows:

$$\mu_* A \in \mathbf{PSh}(\textstyle\int \mu_*\Gamma) = (D : \mathcal{D}, a \in \mu_*\Gamma(D)) \mapsto \mathrm{Hom}_{\mathbf{PSh}(\int \mu^*(\mathbf{y}(D)))}(1, A\left[\widehat{\lfloor a \rceil}\right])$$
$$\mu_* M \in \mathrm{Hom}(1, \mu_* A) = (D : \mathcal{D}, a \in \mu_*\Gamma(D)) \mapsto (\widehat{\lfloor a \rceil})^* M$$

In these definitions, $\widehat{\lfloor a \rceil} \in \mathrm{Hom}(\mu^*(\mathbf{y}(D)), \Gamma)$.

*The presheaf action.* The fact that $\mu_* A$ results in a presheaf is subtle, and we take a moment to specify its action. Given $f : \mathrm{Hom}_{\mathcal{D}}(D', D)$, $a \in \mu_*\Gamma(D)$, $A \in \mathbf{PSh}(\int \Gamma)$, and $x \in \mu_* A(D, a)$, we define $x \cdot f$ as follows:

$$x \cdot f : \mu_* A(D', a \cdot f) \triangleq (\mu^* \mathbf{y}(f))^* x$$

In order to see that this typechecks, recall that $(\mu^* \mathbf{y}(f))^* x$ is an element of the following set:

$$\mathrm{Hom}_{\mathbf{PSh}(\int \mu^*(\mathbf{y}(D')))}((\mu^* \mathbf{y}(f))^* 1, A\left[\widehat{\lfloor a \rfloor} \circ \mu^* \mathbf{y}(f)\right])$$

We can immediately see that $(\mu^* \mathbf{y}(f))^* x = 1$, since reindexing is a right adjoint. Moreover, we have the following chain of equalities:

$$\widehat{\lfloor a \rfloor} \circ \mu^* \mathbf{y}(f) = \widehat{\lfloor a \rfloor \circ \mathbf{y}(f)} = \widehat{a \cdot f}$$

Therefore, this term is an element of $\mu_* A(D', a \cdot f)$ as required. The fact that this action respects composition and identity follows from the fact that reindexing respects composition and identity.

*Naturality.* We must show that both of these definitions are natural with respect to substitutions. That is, $(\mu_* \gamma)^* (\mu_* A) = \mu_* (\gamma^* A)$ and similarly for terms. Again, we will show only the case for types, which is representative for the case on terms.

Suppose we are given $\gamma : \Gamma' \to \Gamma$ and $A \in \mathbf{PSh}(\int \Gamma)$. We wish to show that the following sets are equal:

$$\left(\mathrm{Hom}_{\mathbf{PSh}(\int \mu^* \mathbf{y}(-))}(1, A\left[\widehat{\lfloor - \rfloor}\right])\right)[\mu_* \gamma] = \mathrm{Hom}_{\mathbf{PSh}(\int \mu^* \mathbf{y}(-))}(1, A\left[\widehat{\gamma \circ \lfloor - \rfloor}\right])$$

Unfolding the definition of reindexing, we see that the left-hand side of this equation is the following:

$$\mathrm{Hom}_{\mathbf{PSh}(\int \mu^* \mathbf{y}(-))}(1, A\left[\widehat{\lfloor \gamma(-) \rfloor}\right])$$

However, the naturality of Yoneda means that $\gamma \circ \lfloor x \rfloor$ is identical to $\lfloor \gamma(x) \rfloor$. Therefore, these two sides are identical.

*Context Extension.* We now must show that this functor preserves context extension up to isomorphism. Let us consider the following morphism:

$$\mu_*(\Gamma.A) \xrightarrow{\langle \mu_* \mathbf{p}, \mu_* \mathbf{q} \rangle} \mu_* \Gamma . \mu_* A$$

We wish to show that this is invertible. To start with, we consider a global element $\mathbf{y}(D) \xrightarrow{e} \mu_*(\Gamma.A)$. We can transpose and then decompose $e$ to obtain a pair of maps:

$$\mu^* \mathbf{y}(D) \xrightarrow{e_0} \Gamma \qquad\qquad 1 \xrightarrow{e_1} e_0^* A$$

Here $e_1$ is a morphism in $\mathbf{PSh}(\int \mu^* \mathbf{y}(D))$. We can unfold definitions to see that this is equivalent to $e_1 \in \mu_*(A)(D, \lceil e_0 \rceil)$. Next, we can compute the action of $\langle \mu_* \mathbf{p}, \mu_* \mathbf{q} \rangle$ on this $e$, to observe that it reduces to $\langle \widehat{e}_0, e_1 \rangle$:

$$\begin{aligned}
\langle \mu_* \mathbf{p}, \mu_* \mathbf{q} \rangle \circ e &= \langle \mu_*(\mathbf{p}) \circ e, e^*(\mu_* \mathbf{q}) \rangle \\
&= \langle \widehat{\mathbf{p} \circ \langle e_0, e_1 \rangle}, \widehat{e}^* \mathbf{q} \rangle \\
&= \langle \widehat{e}_0, \langle e_0, e_1 \rangle^* \mathbf{q} \rangle \\
&= \langle \widehat{e}_0, e_1 \rangle
\end{aligned}$$

We can now define an inverse to this map at the level of global elements, sending $\langle \gamma, M \rangle$ to the transpose of $\langle \widehat{\gamma}, M \rangle$.

Size preservation is an easy result of the definition: if $A$ is small, so are its reindexings and so are the collection of its points in each slice. $\square$

## *2.2 Multiple Presheaf Categories*

As a warm up to some of the more interesting applications of MTT, we will start by showing how MTT can model the internal type theories of multiple presheaf categories, interconnected by a graph of essential geometric morphisms. Unlike the rest of §2, this section is not motivated by a particular modal situation in the literature. Accordingly, it is perhaps less interesting than the remaining applications. It does, however, give a good overview of how to apply MTT in a setting where there are relatively few irrelevant details obscuring the main ideas.

Suppose we have some small 2-category $\mathcal{I}$ and a functor $J : \mathcal{I} \to \mathbf{Cat}$. We will construct a model of MTT for reasoning about the internal type theories of $\mathbf{PSh}(J(i))$, for each $i : \mathcal{I}$. More interestingly, we will allow these type theories to interact through the functors $J(f)_*$, for each $f \in \mathrm{Hom}_{\mathcal{I}}(i_0, i_1)$.

Now that we have an intended model, we must construct a mode theory that allows us to capture it with MTT. Our mode theory in this case is easy to find in the case, it is $\mathcal{I}$. Instantiating MTT with $\mathcal{I}$ immediately gives us a type theory, but it remains to show that we can interpret this type theory in the situation we described above. We want to show that there is an interpretation of MTT with $\mathcal{C}[i]$ being sent to $\mathbf{PSh}(J(i))$ and with $[\![\text{🔒}_f]\!]$ being $J(f)^*$. We need to show that this is a contravariant functor, but this is immediate because $J(f)^* \circ J(g)^* = (J(g) \circ J(f))^* = J(g \circ f)^*$.

**Remark 2.2.1.** Already there is a point worth discussing: why should $[\![\text{🔒}_f]\!]$ to be $J(f)^*$, when we want a modality corresponding to $J(f)_*$? Recall from Section 1.4.1 that a model of MTT is determined by a map $\mathcal{M}^{\mathsf{coop}} \to \mathbf{Cat}$. In this case, we have $\mathcal{M} = \mathcal{I}$ (which has discrete 1-cells, so the $-^{\mathsf{co}}$ has no effect), so if we define $\mathcal{C}[i]$ to be $J(i)_*$ then $J(f)_*$ would point the wrong way.

One might wonder why specify a model as a functor out of $\mathcal{M}^{\mathsf{coop}}$ in the first place, if it only leads to this contravariance. This choice, however, is forced: recall that the modalities in MTT do *not* necessarily have to have an action on contexts. Modalities are only required to be defined on types and terms while their adjoints twins, $\text{🔒}_\mu$, only act on contexts. This is why Section 1.4 requires that the functor interpreting the mode theory picks out the interpretation of $\text{🔒}_\mu$, not $\langle \mu \mid - \rangle$: asking for the latter would not always be possible. The end result of these technicalities is that our interpretation of $\text{🔒}_f$ should pick out the left adjoint of $J(f)^*$, being $J(f)_!$.

This is particularly confusing in this instance because in this model $J(f)_*$ *does* have an action on contexts, not just types and terms. In fact, this will be the case in many models because many of our models are democratic [CD14]. Moreover, $J(f)_*$ is a dependent right adjoint, and so it uniquely determines the interpretation of $\text{🔒}_f$ because adjoints are unique. Therefore, in this particular instance we could write a description of the interpretation of the modalities, not the locks, and deduce from it the input required by Theorem 1.4.11. When this occurs in applications going forward, we will skip straight to this more natural description and illustrate our interpretation with a diagram like the following:

$$
\begin{tikzcd}
 & j & & & & \mathbf{PSh}(J(j)) \\
i \arrow[ur] \arrow[dr] & & \arrow[r, mapsto] & \, & \mathbf{PSh}(J(i)) \arrow[ur, "J(f)^*"] \arrow[dr, "J(g)^*"'] & \\
 & k & & & & \mathbf{PSh}(J(k))
\end{tikzcd}
$$

$\lhd$

Now that we have chosen a collection of categories and morphisms between them, we must show two more facts in order to apply Theorem 1.4.11.

1. We must show that each $\mathcal{C}[i]$, $\mathbf{PSh}(J(i))$, supports a model of Martin-Löf Type Theory.

2. We must show that $[\![\text{🔒}_f]\!]$ is left adjoint to a dependent right adjoint.

For the first point, recall that a presheaf topos always gives rise to a quite rich model of type theory, supporting dependent sums and products, extensional identity types (which are sufficient to model intensional identity types), and general indexed inductive types. A standard reference for this model can be found in Hofmann [Hof97] and a description of the interpretation of universes can be found in Coquand [Coq13] or Hofmann and Streicher [HS97]. Again, to make this paper more self-contained, we briefly recall the interpretation of sorts, types, and terms here.

Contexts $\Gamma$ are interpreted as objects of the presheaf category $\mathbf{PSh}(\mathcal{C})$. A type is not interpreted as an object of $\mathbf{PSh}(\mathcal{C})/\Gamma$, as this would lead to strictness issues with substitution. If we were working in an arbitrary locally cartesian closed category, we would be forced to interpret types in this manner, and then apply a strictification theorem [Hof94; LW15]. In the particular case of presheaves, we have a richer model and do not need to resort to such contortions. Instead, types are interpreted as objects of $\mathbf{PSh}(\int \Gamma)$, which is justified by the equivalence $\mathbf{PSh}(\mathcal{C})/\Gamma \simeq \mathbf{PSh}(\int \Gamma)$. A term of type $A$ is then a section of $A$ in $\mathbf{PSh}(\int \Gamma)$, a morphism $\mathrm{Hom}_{\mathbf{PSh}(\int \Gamma)}(1, A)$.

The crucial fact for this model is the interpretation of substitution. Any morphism $\gamma : \Delta \to \Gamma$ gives rise to a functor $\gamma^* : \mathbf{PSh}(\int \Gamma) \to \mathbf{PSh}(\int \Delta)$, defined by essentially precomposition. These functors are then necessarily strictly functorial, $\gamma^* \circ \delta^* = (\delta \circ \gamma)^*$, justifying the interpretation of substitutions by $\gamma^*$.

**Remark 2.2.2** (Size Issues)**.** There is a small issue of size in considering $\mathbf{PSh}(\mathcal{C})$ as a model of type theory in general, and MTT in particular. We have defined a category of contexts to be a *small* category so that the category of models can be formulated without issue. $\mathbf{PSh}(\mathcal{C})$, however, is certainly not small. This obstacle can be avoided with the introduction of Grothendieck universes into our metatheory. Instead of considering presheaves valued in all of $\mathbf{Set}$, we consider presheaves valued in $\mathcal{V}$. There is no loss of expressivity, because $\mathcal{V}$ is closed under all set-theoretic operations. With this restriction, $\mathbf{PSh}(\mathcal{C})$ is small.

These maneuvers with Grothendieck universes are technically necessary, but fundamentally uninteresting. To a type theorist, this is just the standard technique of "bumping a universe level" when using $\mathbf{PSh}(\mathcal{C})$ as a model. With this justification, we shall not remark further on issues of size and will assume the Grothendieck universe axiom to ensure an ample supply of universes. For instance, the interpretation of universes will require that the choice of Grothendieck universe for $\mathbf{PSh}(\mathcal{C})$ be large enough to contain an inner universe of its own. $\lhd$

Now that we have addressed the "mode local" models of type theory, we must show that $[\![\text{🔒}_f]\!] = L(f)^*$ forms the left half of a dependent right adjoint. This follows from Lemma 2.1.5 and the standard fact that $L(f)^* \dashv L(f)_*$. All that remains to complete the construction of this model is to apply Theorem 1.4.11.

This instantiation of MTT gives us a way to reason simultaneously in multiple presheaf categories at once, passing back and forth using the modal types. It also illustrates the standard process for constructing a model of MTT: picking a mode theory and interpretation, constructing mode-local models of type theory, and applying Theorem 1.4.11. This example also would easily scale up to incorporate non-trivial 2-cells, or modalities of other shapes to this type theory. All that is needed is to add these to $\mathcal{I}$ and explain how the new components are interpreted.

**Remark 2.2.3** (Interpreting the Modality as Precomposition)**.** In this example we have interpreted the modality as the direct image of a functor, $F_*$, and the lock is interpreted as the left adjoint to this functor, $F^*$. One might wish to instead interpret the modality as $F^*$, this is a right adjoint as well as left adjoint with $F_! \dashv F^*$.

The construction of this model is not quite as simple as the model using $F_*$. The issue is a strictness mismatch: we have $F_! \circ G_! \cong (G \circ F)_!$, but this isomorphism not an equality as required by Theorem 1.4.11. On the other hand, if we use Theorem 1.6.26, then this mismatch can be papered over by replacing the pseudo-functor interpreting one-cells with $F_!$ with an equivalent strict functor. In fact, assuming this conjecture we can freely mix modalities which are interpreted using both $-^*$ and $-_*$ in the same type theory. $\lhd$

## 2.3 Guarded Recursion

In this application we wish to study the simplest form of guarded recursion. Let's start by picking the following mode theory, $\mathcal{M}$:

$$\begin{tikzcd} t \arrow[loop above, "\ell"] \arrow[d, bend left, "\gamma"] \\ s \arrow[u, bend left, "\delta"] \end{tikzcd}$$

*The Adjoint Bowling Pin*

$$\delta \circ \gamma \leq 1 \qquad 1 = \gamma \circ \delta$$
$$1 \leq \ell \qquad \gamma = \gamma \circ \ell$$

We wish to construct an interpretation of this mode theory with $s$ being interpreted as $\mathbf{Set}$ and $t$ being interpreted as $\mathbf{PSh}(\omega)$. Moreover, we will want the interpretation of $\langle \ell \mid A \rangle$ to be induced by the adjunction $\blacktriangleleft \dashv \blacktriangleright$[1] from Birkedal et al. [Bir+12], $\langle \delta \mid A \rangle$ by $\Pi_0 \dashv \Delta$, and $\langle \gamma \mid A \rangle$ by $\Delta \dashv \Gamma$. To begin with, we must construct a 2-functor, $L$, from $\mathcal{M}^{\mathsf{coop}}$ to $\mathbf{Cat}$.

Rather than constructing $L$ directly, it proves simpler to factor it through the 2-functor $\mathbf{PSh}(-) : \mathbf{Pos}^{\mathsf{coop}} \to \mathbf{Cat}$. The advantage of this factorization is that $\mathbf{Pos}$ is poset-enriched, like $\mathcal{M}$, and so checking the enrichment of the factorization is far simpler. We will write the factorization $L = \mathbf{PSh}(-) \circ L'$ and define $L'$ as follows on the 0- and 1-cells:

$$L'(s) = \star \qquad\qquad L'(\ell) = n \mapsto n + 1$$
$$L'(t) = \omega \qquad\qquad L'(\delta) = \star \mapsto 0$$
$$L'(\gamma) = n \mapsto \star$$

What remains is to show that there exist the required (in)equalities between the interpretation of the 1-cells. There is no extra data to be conveyed here: there is at most one inequality between maps between posets. We must show the following (in)equalities

$$\delta \circ \gamma \leq 1 \qquad\qquad 1 = \gamma \circ \delta$$
$$1 \leq \ell \qquad\qquad \gamma = \gamma \circ \ell$$

Each of these can be checked calculation. We will show $\delta \circ \gamma \leq 1$ for a representative example. We must show for all $n$, that $L'(\delta \circ \gamma)(n) \leq n$. We can unfold $L'(\delta \circ \gamma)(n)$ as follows:

$$L'(\delta \circ \gamma)(n) = L'(\delta)(L'(\gamma)(n)) = L'(\delta)(\star) = 0$$

Since $0 \leq n$ for any $n$, this inequality holds. Having constructed with $L'$, we post-compose it with $\mathbf{PSh}(-)$ to construct the required 2-functor $M^{\mathsf{coop}} \to \mathbf{Cat}$. We can unfold the definitions to see that we have interpreted the 0- and 1-cells in the expected way:

$$L(s) = \mathbf{Set} \qquad\qquad L(\ell) = \blacktriangleleft$$
$$L(t) = \mathbf{PSh}(\omega) \qquad\qquad L(\delta) = \Pi_0$$
$$L(\gamma) = \Delta$$

These computations tell us that $L$ is a valid 2-functor $M^{\mathsf{coop}} \to \mathbf{Cat}$. Additionally, because $\mathbf{Set}$ and $\mathbf{PSh}(\omega)$ are both presheaf categories, we have a standard model of Martin-Löf Type Theory in both. Finally, we may apply Lemma 2.1.5 to the adjunctions $\blacktriangleleft \dashv \blacktriangleright$, $\Delta \dashv \Gamma$, and $\Pi_0 \dashv \Delta$ to see that each of these adjunctions gives rise to a dependent right adjoint. This is all the data needed to apply Theorem 1.4.11 and so we have the following.

[1] $\blacktriangleleft$ is induced by precomposition with $n \mapsto n + 1$, this adjunction is proven in Birkedal et al. [Bir+12].

**Theorem 2.3.1.** *There is a model of MTT with mode theory $\mathcal{M}$ interpreting $s$ as $\mathbf{Set}$, $t$ as $\mathbf{PSh}(\omega)$. Furthermore, this model interprets $\delta$ by the dependent right adjoint arising from $\Pi_0 \dashv \Delta$, $\gamma$ by $\Delta \dashv \Gamma$, and $\ell$ by $\blacktriangleleft \dashv \blacktriangleright$.*

We can summarize this theorem diagrammatically, by saying that we can interpret MTT into the following diagram:

$$\begin{tikzcd}
\mathbf{PSh}(\omega) \arrow[loop above, "\blacktriangleright"] \arrow[d, bend left, "\Gamma"] \\
\mathbf{Set} \arrow[u, bend left, "\Delta"]
\end{tikzcd}$$

Of course, this simple diagram is capturing a great deal of information. It is asserting the existence of model of type theory in both $\mathbf{Set}$ and $\mathbf{PSh}(\omega)$, as well as a left adjoint to each of the three functors, an extension of each of the functors as *dependent* right adjoints, not merely right adjoints, as well as the validity of all the equalities and 2-cells of $\mathcal{M}$.

**Remark 2.3.2** (Key Substitutions)**.** This mode theory is merely poset-enriched, as opposed to being a proper 2-category. As a result, the key substitutions for navigating between $\Gamma.\text{🔒}_\mu$ and $\Gamma.\text{🔒}_\nu$ must be considerably simpler. In particular, for any $\mu, \nu$, there is at most one such key substitution $\Gamma.\text{🔒}_\mu \vdash \text{🗝}^{\mu \geq \nu}_{\Gamma} : \Gamma.\text{🔒}_\nu \mathrel{@} m$. This property means that we can (without any ambiguity) elide key substitutions entirely in our terms: they can always be uniquely inferred.

This, however, leads to terms that are more difficult to type-check, so we adopt a compromise in what follows. We will write $A^{\nu \leq \mu}$ or $M^{\nu \leq \mu}$ for the application of the unique key substitution in context $\Gamma.\text{🔒}_\mu$ induced by $\mu \geq \nu$. For instance, given a type $\Gamma \vdash A\ \mathsf{type}_\ell \mathrel{@} t$, we could form $\Gamma \vdash \langle \ell \mid A^{1 \leq \ell} \rangle\ \mathsf{type}_\ell \mathrel{@} t$. ◁

### 2.3.1 Specializing MTT

Now that we have specified our mode theory and explained the intended model, we will specialize our notation and syntax for this application. We fix the following shorthands:

$$\begin{aligned}
b &= \delta \circ \gamma \\
\square A &= \langle b \mid A \rangle \\
\blacktriangleright A &= \langle \ell \mid A \rangle \\
\Gamma A &= \langle \gamma \mid A \rangle \\
\Delta A &= \langle \delta \mid A \rangle
\end{aligned}$$

The first is a definition in $\mathcal{M}$, while all the rest are definitions of types. We would like to establish that MTT with this mode theory specializes in two important ways:

1. The modalities on mode $t$ should give rise to the standard modalities and operations from Guarded Type Theory [Biz+16] inside the type theory.[2]

[2] These facts are certainly true in our intended model, but we wish to go a step further and show that this structure can be constructed inside MTT.

More explicitly, we will show that $\Box$ is an idempotent comonad by constructing a map $\Box A \to A$, and $\Box A \to \Box\Box A$ and showing that they satisfy the expected laws. Additionally, we will construct the operations and proofs to demonstrate that $\blacktriangleright$ is an applicative functor [MP08] and that $\Box \blacktriangleright A \simeq \Box A$.

2. The type theory when restricted to mode $s$ is standard Martin-Löf Type Theory.

First, we can show that $\Box$ is an idempotent comonad using the following operations (using terms from Section 1.3):

$$\begin{aligned}
&\mathsf{dup}_A &&: \Box\Box A \to \Box A \\
&\mathsf{dup}_A(x) &&\triangleq \mathbf{comp}^{-1}_{b,b}(x) \\[1ex]
&\mathsf{extract}_A &&: \Box A \to A^{b\le 1} \\
&\mathsf{extract}_A(x) &&\triangleq \mathbf{triv}^{-1}(\mathbf{coe}[b \le 1](x))
\end{aligned}$$

We still have the K operator from Section 1.3.3: $f \circledast_b a$. We wish to show that these operations together give us an (idempotent) comonad. We must show the following equalities:

$$(x : \Box A) \to \mathsf{Id}_{\Box A}(x, \mathsf{box}(\mathsf{extract}) \circledast \mathsf{dup}(x)) \tag{2.3}$$

$$(x : \Box A) \to \mathsf{Id}_{\Box A}(x, \mathsf{extract}(\mathsf{dup}(x))) \tag{2.4}$$

$$(x : \Box A) \to \mathsf{Id}_{\Box\Box\Box A}(\mathsf{dup}(\mathsf{dup}(x)), \mathsf{box}(\mathsf{dup}) \circledast \mathsf{dup}(x)) \tag{2.5}$$

These terms can be constructed essentially by induction and unfolding. In order to make the proofs slightly more accessible, we have presented them not as terms, but as a series of equational steps. In what follows, understand that $=$ denotes mere internal equality, *not* judgmental equality. First, for 2.3

$$\begin{aligned}
&\mathsf{box}(\mathsf{extract}) \circledast \mathsf{dup}(x) \\
&= \mathsf{mod}_b(\lambda x.\, \mathbf{triv}^{-1}(\mathbf{coe}[b \le 1](x))) \circledast \mathbf{comp}^{-1}_{b,b}(x) \\
&\qquad\qquad \text{Use induction to consider the case where } x = \mathsf{mod}_b(y) \\
&= \mathsf{mod}_b(\lambda x.\, \mathbf{triv}^{-1}(\mathbf{coe}[b \le 1](x))) \circledast \mathbf{comp}^{-1}_{b,b}(\mathsf{mod}_b(y)) \\
&= \mathsf{mod}_b(\lambda x.\, \mathbf{triv}^{-1}(\mathbf{coe}[b \le 1](x))) \circledast \mathsf{mod}_b(\mathsf{mod}_b(y)) \\
&= \mathsf{mod}_b((\lambda x.\, \mathbf{triv}^{-1}(\mathbf{coe}[b \le 1](x)))(\mathsf{mod}_b(y))) \\
&= \mathsf{mod}_b(\mathbf{triv}^{-1}(\mathbf{coe}[b \le 1](\mathsf{mod}_b(y)))) \\
&= \mathsf{mod}_b(y) \\
&= x
\end{aligned}$$

The calculation for 2.4 is similar, proceeding by expanding all relevant definitions and performing induction.

$$\begin{aligned}
&\mathsf{extract}(\mathsf{dup}(x)) \\
&= \mathbf{triv}^{-1}(\mathbf{coe}[b \le 1](\mathbf{comp}^{-1}_{b,b}(x))) &&\qquad \text{replace } x \text{ with } \mathsf{mod}_b(y) \\
&= \mathbf{triv}^{-1}(\mathbf{coe}[b \le 1](\mathbf{comp}^{-1}_{b,b}(\mathsf{mod}_b(y)))) \\
&= \mathbf{triv}^{-1}(\mathbf{coe}[b \le 1](\mathsf{mod}_b(\mathsf{mod}_b(y)))) \\
&= \mathbf{triv}^{-1}(\mathsf{mod}_1(\mathsf{mod}_b(y))) \\
&= \mathsf{mod}_b(y) \\
&= x
\end{aligned}$$

The proof of 2.5 is more of the same, and thus we have elided it.

We immediately have that $\blacktriangleright$ is an applicative functor: this is guaranteed by Section 1.3.3. We additionally have that $\blacktriangleright$ has an appropriate point: $\mathsf{next}(x) = \mathbf{coe}[1 \leq \ell](\mathbf{triv}(x))$.

The importance of the inclusion of $\square$ into our type theory is the interaction between $\square$ and $\blacktriangleright$, namely that $\square \blacktriangleright A \simeq \square A$. We can recover this interaction inside our type theory, with the function $\square \blacktriangleright A \to \square A$ coming from Section 1.3.1: $\mathsf{now}(x) = \mathbf{comp}^{-1}_{b,\ell}(x)$.

As a final check, we can ensure that the following transformation is the identity:

$$\square A \xrightarrow{\mathsf{box}(\mathsf{next}) \circledast -} \square \blacktriangleright A \xrightarrow{\mathsf{box}(\mathsf{now}) \circledast -} \square A$$

The calculation is as follows:

$$\begin{aligned}
&\mathbf{comp}_{b,\ell}(\mathsf{mod}_b(\mathbf{coe}[1 \leq \ell](\mathbf{triv}(-))) \circledast x) && \text{By induction, suppose } x = \mathsf{mod}_b(y)\\
&= \mathbf{comp}_{b,\ell}(\mathsf{mod}_b(\mathbf{coe}[1 \leq \ell](\mathbf{triv}(-))) \circledast \mathsf{mod}_b(y))\\
&= \mathbf{comp}_{b,\ell}(\mathsf{mod}_b(\mathsf{mod}_\ell(y)))\\
&= \mathbf{comp}_{b,\ell}(\mathsf{mod}_b(\mathsf{mod}_\ell(y)))\\
&= \mathsf{mod}_b(y)\\
&= x
\end{aligned}$$

For the second point, we wish to show that if we restrict our attention to only types of mode $s$ and endomodalities $\mu \in \mathrm{Hom}(s,s)$, the result is Martin-Löf Type Theory. A routine calculation shows that any such modality $\mu$ must be equal to 1, using induction and the fact that $\gamma \circ \ell^n \circ \delta = 1$. This implies that $\langle \mu \mid A \rangle$ is also equivalent to $A$, using $\mathbf{triv}(-)$. Finally, we can specialize our variable rule further, as there is no non-trivial 2-cell $1 \Rightarrow 1$:

$$\frac{\mu \in \mathrm{Hom}(s,s) \qquad \Gamma\ \mathsf{ctx} \mathrel{@} s \qquad \Gamma \vdash A\ \mathsf{type}_\ell \mathrel{@} s \qquad (x : (\mu \mid A)) \in \Gamma}{\Gamma \vdash x : A \mathrel{@} s}$$

**Remark 2.3.3.** This same rule does not hold true if we assume that $\mu \in \mathrm{Hom}(s,t)$. Consider $\mu = \ell \circ \delta$, if we think in terms of the semantics with $\mathbf{PSh}(\omega)$, we then have $\mathbf{Mod}_\mu = \blacktriangleright \circ \Delta$. However, we then can see that $\mathbf{Mod}_\mu(\emptyset)$ is *not* just $\_ \mapsto \emptyset$, because it is represented by $\blacktriangleright 0$ which is locally non-zero. $\lhd$

### 2.3.2 *Reasoning about Guarded Streams*

We now turn to putting MTT to work. Specifically, we wish to use this modal situation to reason about infinite streams, the canonical coinductive data type.

**Remark 2.3.4** (Historical Context)**.** Using guarded type theories to reason about coinduction is a long running program, and in this section we will draw on examples presented first in Bizjak et al. [Biz+16]. This type theory was similar to MTT, using the later modality and Löb induction to construct guarded fixed-points, which could then be refined to true coinductive definitions. Unlike MTT, however, this second step used *clocks* [AM13]. In essence, Bizjak et al. [Biz+16] does not have a single $\blacktriangleright$ modality, but rather an entire collection of them, each indexed by a clock name. There is a quantifier which allows clock names to be bound inside a particular type, and a crucial isomorphism:

$$\forall \kappa.\ \blacktriangleright^\kappa A \cong \forall \kappa.\ A \tag{$*$}$$

This work, however, suffered from several technical complications. For instance, Bizjak et al. [Biz+16] is forced to use *delayed substitutions* in order to handle the combination of dependent types and modalities. Delayed substitutions pollute the equational theory and are well-known to be obstacles to providing an implementation of **gDTT**. This issue was later resolved by Clocked Type Theory (CloTT) [BGM17], which introduced a judgmental structure to capture the later modality and proved a normalization result. It is conjectured that type-checking is decidable for CloTT and there is ongoing implementation work.

The other issue, however, is the inherent complexity in using clocks to obtain Eq. $(*)$. This complexity is reflected in the syntax, but it is more serious in the models. Rather than just being interpreted into $\mathbf{PSh}(\omega)$, CloTT is modeled in a collection of different presheaf categories, with functors navigating between them [MM18]. These models are well-studied, but it was hoped that some of the complexity could be circumvented by introducing a second modality rather than clocks. In Clouston et al. [Clo+15], for instance, rather than using clock quantifiers a new modality is introduced to capture the same phenomenon in a simple type theory. In particular, Eq. $(*)$ is replaced by the following:

$$\square \blacktriangleright A \cong \square A \tag{$\dagger$}$$

The main advantage of $\square$ is that $\mathbf{PSh}(\omega)$ is once again a valid model. On the other hand, the interactions between $\square$ and $\blacktriangleright$ have proven difficult to capture inside a dependent type theory; indeed, merely adding $\square$ to a dependent type theory has proven to be a significant technical challenge. Recently, Gratzer, Sterling, and Birkedal [GSB19] constructed a complete story for the addition of $\square$ to a full dependent type theory, building on previous work [BGM17; Bir+20; Shu18]. Despite this effort, however, there are still serious technical obstacles to the addition of $\blacktriangleright$ to Gratzer, Sterling, and Birkedal's [GSB19] type theory. This instantiation of MTT continues this line of research by eschewing clocks in favor of $\square$, but by providing a sufficiently flexible syntax to incorporate both $\square$, $\blacktriangleright$, and Eq. $(\dagger)$. $\lhd$

We will demonstrate that MTT with this mode theory is sufficiently expressive to carry out the coinductive constructions from the clocked setting by reproducing an example from Bizjak et al. [Biz+16]: we will show that $\mathsf{zipWith}(f)$ on a coinductive stream is commutative if $f$ itself is commutative. Prior to constructing these programs, however, we will alter MTT in a few ways:

1. We replace the intensional equality $\mathsf{Id}_A(M_0, M_1)$ with extensional equality $\mathsf{Eq}_A(M_0, M_1)$.

2. We add Löb induction as an axiom.

The first change is easily accommodated by our intended model in $\mathbf{PSh}(\omega)$ and $\mathbf{Set}$: the interpretation of intensional identity was already a valid interpretation of the stronger extensional version. The switch to extensional equality is not strictly necessary: we could carry out the following examples in an intensional identity setting. Doing so, however, would make the proof terms more verbose (much more so than is pleasant on paper!) and we would have to add a functional extensionality axiom. Moreover, Bizjak et al. [Biz+16] is an extensional type theory, and we will copy this decision to better facilitate a comparison between MTT and **gDTT**.

The second addition is more subtle, but Löb induction is a crucial *modal specific* operator which cannot be captured directly by a framework like MTT. To be more precise, we add the following rules to MTT:

$$\frac{\Gamma\ \mathsf{ctx} \mathbin{@} t \qquad \Gamma \vdash A\ \mathsf{type}_1 \mathbin{@} t}{\Gamma \vdash \mathsf{löb} : (\blacktriangleright A^{1 \le \ell} \to A) \to A \mathbin{@} t} \qquad \frac{\Gamma\ \mathsf{ctx} \mathbin{@} t \qquad \Gamma \vdash A\ \mathsf{type}_1 \mathbin{@} t \qquad \Gamma \vdash M : \blacktriangleright A^{1 \le \ell} \to A \mathbin{@} t}{\Gamma \vdash \mathsf{löb}(M) = M(\mathsf{next}(\mathsf{löb}(M))) : (\blacktriangleright A^{1 \le \ell} \to A) \to A \mathbin{@} t}$$

Notice that these rules are not added at both $s$ and $t$, these rules admit only a sensible interpretation in mode $t$. This new operation admits a sound interpretation in $\mathbf{PSh}(\omega)$, which justifies adding them to the theory. We have also added a definitional unfolding for Löb (which is additionally validated by $\mathbf{PSh}(\omega)$), this certainly disrupts normalization but also provides a pleasant experience in actually working with guarded fixed points.

**Theorem 2.3.5.** $\mathsf{löb}(M)$ *is (internally) the unique fixed point of $M$, i.e. there is a term of the following type:*

$$(A : \mathsf{U})(x : \mathsf{El}(A)) \to \mathsf{Eq}_{\mathsf{El}(A)}(M(\mathsf{next}(x)), x) \to \mathsf{Eq}_{\mathsf{El}(A)}(\mathsf{löb}(M), x)$$

*Proof.* The term witnessing this construction is just $\lambda A.\ \text{löb}(\lambda f.\ \lambda x.\ \lambda p.\ \mathsf{refl}(x))$. This type-checks thanks to the equality reflection rule, though it is less than informative!

We will instead present the equational reasoning that leads to this term being well-typed. Let us suppose that we have $A : \mathsf{U}$, $f : \blacktriangleright((x : \mathsf{El}(A)) \to \mathsf{Eq}_{\mathsf{El}(A)}(M(\mathsf{next}(x)), x) \to \mathsf{Eq}_{\mathsf{El}(A)}(\text{löb}(M), x))$, $x : A$ and $p : \mathsf{Eq}_{\mathsf{El}(A)}(M(\mathsf{next}(x)), x)$.

$$
\begin{aligned}
\text{löb}(M) &= M(\mathsf{next}(\text{löb}(M))) && \text{Unfolding rule for löb} \\
&= M(\mathsf{next}(x)) && \text{Reflecting } f \circledast \mathsf{next}(x) \circledast \mathsf{next}(p) : \blacktriangleright\mathsf{Eq}_A(\text{löb}(M), x) \\
&= x && \text{Reflecting } p : \mathsf{Eq}_A(M(\mathsf{next}(x)), x)
\end{aligned}
$$
□

This proof uses a general technique that we will note explicitly here: when we have an equality under a $\blacktriangleright$, we can use it to rewrite terms which appear next (more generally, in a locked context).

We can use Löb operator to form guarded recursive types. We will now fix the primary object of study in the rest of this section:

$$
\begin{aligned}
&\mathsf{Str}' \ : \ (\delta \mid \mathsf{U}) \to \mathsf{U} \ @\ t \\
&\mathsf{Str}' \triangleq \lambda A.\ \text{löb}(\lambda x.\ (\Delta A) \times \mathsf{let}\ \mathsf{mod}_{\ell}(x') \leftarrow x\ \mathsf{in}\ \blacktriangleright x') \\
&\mathsf{Str} \ \ : \ \mathsf{U} \to \mathsf{U} \ @\ s \\
&\mathsf{Str} \ \triangleq \lambda A.\ \Gamma(\mathsf{Str}'(A))
\end{aligned}
$$

We have separated this definition of coinductive streams into two halves: $\mathsf{Str}'$ uses löb to form a guarded fixed-point on $\mathsf{U}$ which defines infinite streams, while $\mathsf{Str}$ does the required modal plumbing to move this definition from mode $t$ to mode $s$. These definition is the first example of mixing two modalities, so we will take a moment to type-check $\mathsf{Str}(A)$ more explicitly. We end up needing to show the following:

$$\cdot, A : (1 \mid \mathsf{U}).\text{🔒}_{\gamma} \vdash \mathsf{Str}'(A) : \mathsf{U} \ @\ t$$

It sufficies to check that $\cdot, A : (1 \mid \mathsf{U}).\text{🔒}_{\gamma}.\text{🔒}_{\delta} \vdash A : \mathsf{U} \ @\ s$. This is not entirely obvious, $A$ is under two locks but only modified by 1, so it may not be accessible. However, our mode theory tells us that $\gamma \circ \delta = 1$, and so our context is equal to $\cdot, A : (1 \mid \mathsf{U}).\text{🔒}_{1}$, and in this context we can use $A$.

With this definition of streams in hand, we can start by defining a few operations for constructing and deconstructing streams.

$$
\begin{aligned}
&\mathsf{cons} && : (A : \mathsf{U}) \to \mathsf{El}(A) \to \mathsf{El}(\mathsf{Str}(A)) \to \mathsf{El}(\mathsf{Str}(A)) \\
&\mathsf{cons}_A(h, t) && \triangleq \mathsf{let}\ \mathsf{mod}_{\gamma}(t') \leftarrow t\ \mathsf{in}\ \mathsf{mod}_{\gamma}((\mathsf{mod}_{\delta}(h), \mathsf{next}(t'))) \\[1ex]
&\mathsf{head} && : (A : \mathsf{U}) \to \mathsf{El}(\mathsf{Str}(A)) \to \mathsf{El}(A) \\
&\mathsf{head}_A(s) && \triangleq \mathsf{let}\ \mathsf{mod}_{\gamma}(s') \leftarrow s\ \mathsf{in}\ \mathbf{triv}^{-1}(\mathbf{comp}_{\delta,\gamma}(\mathsf{mod}_{\gamma}(\mathsf{pr}_0(s')))) \\[1ex]
&\mathsf{tail} && : (A : \mathsf{U}) \to \mathsf{El}(\mathsf{Str}(A)) \to \mathsf{El}(\mathsf{Str}(A)) \\
&\mathsf{tail}_A(s) && \triangleq \mathsf{let}\ \mathsf{mod}_{\gamma}(s') \leftarrow s\ \mathsf{in}\ \mathbf{comp}_{\ell,\gamma}(\mathsf{mod}_{\gamma}(\mathsf{pr}_1(s')))
\end{aligned}
$$

Those familiar with prior work on guarded streams may be surprised by the type of tail. The typical definition of guarded streams would have a type more like $\mathsf{Str}(A) \to \blacktriangleright\mathsf{Str}(A)$, owing the fact that Str is defined by a guarded fixed-point. In our case, however, the $\Gamma$ modality is sufficiently strong to "absorb" this extra $\blacktriangleright$, similar to Eq. (†). In fact, the "obvious" definition of tail would be the following:

$$\mathsf{tail}_A(s) \stackrel{?}{=} \mathsf{let}\ \mathsf{mod}_{\gamma}(s') \leftarrow s\ \mathsf{in}\ \mathsf{mod}_{\gamma}(\mathsf{pr}_1(s'))$$

This definition has the type $\mathsf{El}(\mathsf{Str}(A)) \to \Gamma(\blacktriangleright\mathsf{El}(\mathsf{Str}'(A)))$. However, we can now make use the equality $\gamma \circ \ell = \gamma$ in $\mathcal{M}$ and use Section 1.3.1 to adjust by the isomorphism $\Gamma \cong \Gamma \circ \blacktriangleright$ to obtain the proper version of tail. This small difference is the crucial difference which will make $\mathsf{Str}(A)$ a final coalgebra, as we shall demonstrate presently.

**Lemma 2.3.6.** *These operations satisfy the expected $\beta$ and $\eta$ laws. That is, there are terms of the following types:*

1. $(h : \mathsf{El}(A))(t : \mathsf{El}(\mathsf{Str}(A))) \to \mathsf{Eq}_{\mathsf{El}(A)}(\mathsf{head}_A(\mathsf{cons}_A(h, t)), h)$
2. $(h : \mathsf{El}(A))(t : \mathsf{El}(\mathsf{Str}(A))) \to \mathsf{Eq}_{\mathsf{El}(\mathsf{Str}(A))}(\mathsf{tail}_A(\mathsf{cons}_A(h, t)), t)$
3. $(h : \mathsf{El}(A))(t : \mathsf{El}(\mathsf{Str}(A))) \to \mathsf{Eq}_{\mathsf{El}(\mathsf{Str}(A))}(s, \mathsf{cons}_A(\mathsf{head}_A(s), \mathsf{tail}_A(s)))$

*Proof.* We will prove these case by case:

1. First, let us suppose we have some $h : \mathsf{El}(A)$ and $t : \mathsf{El}(\mathsf{Str}(A))$. We will show that $\mathsf{refl}(h)$ has the appropriate type. In order to do this, we must reduce $\mathsf{head}_A(\mathsf{cons}_A(h, t))$:

$$\begin{aligned}
\mathsf{head}_A(\mathsf{cons}_A(h, t)) &= \mathsf{head}_A(\mathsf{cons}_A(h, \mathsf{mod}_\gamma(t'))) \\
&\qquad \text{Using induction to replace } t \text{ with } \mathsf{mod}_\gamma(t') \\
&= \mathsf{head}_A(\mathsf{let}\ \mathsf{mod}_\gamma(x) \leftarrow \mathsf{mod}_\gamma(t')\ \mathsf{in}\ \mathsf{mod}_\gamma((\mathsf{mod}_\delta(h), \mathsf{next}(x)))) \\
&= \mathsf{head}_A(\mathsf{mod}_\gamma((\mathsf{mod}_\delta(h), \mathsf{next}(t')))) \\
&= \mathsf{let}\ \mathsf{mod}_\gamma(s') \leftarrow \mathsf{mod}_\gamma((\mathsf{mod}_\delta(h), \mathsf{next}(t')))\ \mathsf{in}\ \mathbf{triv}^{-1}(\mathbf{comp}_{\delta,\gamma}(\mathsf{mod}_\gamma(\mathsf{pr}_0(s')))) \\
&= \mathbf{triv}^{-1}(\mathbf{comp}_{\delta,\gamma}(\mathsf{mod}_\gamma(\mathsf{pr}_0((\mathsf{mod}_\delta(h), \mathsf{next}(t')))))) \\
&= \mathbf{triv}^{-1}(\mathbf{comp}_{\delta,\gamma}(\mathsf{mod}_\gamma(\mathsf{mod}_\delta(h)))) \\
&= \mathbf{triv}^{-1}(\mathsf{mod}_1(h)) \\
&= h
\end{aligned}$$

2. Again, assuming that we have an appropriate $h$ and $t$, we claim that $\mathsf{refl}(t)$ has the appropriate type. In order to show this, we will reduce $\mathsf{tail}_A(\mathsf{cons}_A(h, t))$:

$$\begin{aligned}
\mathsf{tail}_A(\mathsf{cons}_A(h, t)) &= \mathsf{tail}_A(\mathsf{cons}_A(h, \mathsf{mod}_\gamma(t'))) \\
&\qquad \text{Using induction to replace } t \text{ with } \mathsf{mod}_\gamma(t') \\
&= \mathsf{tail}_A(\mathsf{let}\ \mathsf{mod}_\gamma(x) \leftarrow \mathsf{mod}_\gamma(t')\ \mathsf{in}\ \mathsf{mod}_\gamma((\mathsf{mod}_\delta(h), \mathsf{next}(x)))) \\
&= \mathsf{tail}_A(\mathsf{mod}_\gamma((\mathsf{mod}_\delta(h), \mathsf{next}(t')))) \\
&= \mathsf{let}\ \mathsf{mod}_\gamma(s') \leftarrow \mathsf{mod}_\gamma((\mathsf{mod}_\delta(h), \mathsf{next}(t')))\ \mathsf{in}\ \mathbf{comp}_{\ell,\gamma}(\mathsf{mod}_\gamma(\mathsf{pr}_1(s'))) \\
&= \mathbf{comp}_{\ell,\gamma}(\mathsf{mod}_\gamma(\mathsf{pr}_1((\mathsf{mod}_\delta(h), \mathsf{next}(t'))))) \\
&= \mathbf{comp}_{\ell,\gamma}(\mathsf{mod}_\gamma(\mathsf{next}(t'))) \\
&= \mathsf{mod}_\gamma(t') \\
&= t
\end{aligned}$$

3. Finally, we assume that we have $s : \mathsf{El}(\mathsf{Str}(A))$. We will show that $\mathsf{refl}(s)$ has the appropriate type. For this, we must calculate on $\mathsf{cons}_A(\mathsf{head}_A(s), \mathsf{tail}_A(s))$:

$$\begin{aligned}
&\mathsf{cons}_A(\mathsf{head}_A(s), \mathsf{tail}_A(s)) \\
&\quad = \mathsf{cons}_A(\mathsf{head}_A(\mathsf{mod}_\gamma(s')), \mathsf{tail}_A(\mathsf{mod}_\gamma(s'))) \\
&\qquad \text{Using induction to replace } s \text{ with } \mathsf{mod}_\gamma(s') \\
&\quad = \mathsf{cons}_A(\mathsf{head}_A(\mathsf{mod}_\gamma(s')), \mathsf{tail}_A(\mathsf{mod}_\gamma(s'))) \\
&\quad = \mathsf{cons}_A(\mathsf{let}\ \mathsf{mod}_\gamma(s') \leftarrow \mathsf{mod}_\gamma(s')\ \mathsf{in}\ \mathbf{triv}^{-1}(\mathbf{comp}_{\delta,\gamma}(\mathsf{mod}_\gamma(\mathsf{pr}_0(s')))), \mathsf{tail}_A(\mathsf{mod}_\gamma(s'))) \\
&\quad = \mathsf{cons}_A(\mathbf{triv}^{-1}(\mathbf{comp}_{\delta,\gamma}(\mathsf{mod}_\gamma(\mathsf{pr}_0(s')))), \mathsf{tail}_A(\mathsf{mod}_\gamma(s'))) \\
&\quad = \mathsf{cons}_A(\mathbf{triv}^{-1}(\mathbf{comp}_{\delta,\gamma}(\mathsf{mod}_\gamma(\mathsf{pr}_0(s')))), \mathbf{comp}_{\ell,\gamma}(\mathsf{mod}_\gamma(\mathsf{pr}_1(s'))))
\end{aligned}$$

$$
\begin{aligned}
&\quad \text{Write } s' = (h, t) \\
&= \mathsf{cons}_A(\mathbf{triv}^{-1}(\mathbf{comp}_{\delta,\gamma}(\mathsf{mod}_\gamma(h))), \mathbf{comp}_{\ell,\gamma}(\mathsf{mod}_\gamma(t))) \\
&\quad \text{Use induction to replace } h \text{ with } \mathsf{mod}_\delta(h') \text{ and } t \text{ with } \mathsf{mod}_\ell(t'). \\
&= \mathsf{cons}_A(\mathbf{triv}^{-1}(\mathbf{comp}_{\delta,\gamma}(\mathsf{mod}_\gamma(\mathsf{mod}_\delta(h')))), \mathbf{comp}_{\ell,\gamma}(\mathsf{mod}_\gamma(\mathsf{mod}_\ell(t')))) \\
&= \mathsf{cons}_A(h', \mathsf{mod}_\gamma(t')) \\
&= \mathsf{mod}_\gamma((\mathsf{mod}_\delta(h'), \mathsf{next}(t'))) \\
&= \mathsf{mod}_\gamma((\mathsf{mod}_\delta(h'), \mathsf{mod}_\ell(t'))) \\
&= \mathsf{mod}_\gamma((h, t)) \\
&= \mathsf{mod}_\gamma(s') \\
&= s
\end{aligned}
$$

□

**Lemma 2.3.7.** *Given an element $A : \mathsf{U}$, the function $\lambda B.\, A \times B$ of elements of $\mathsf{U}$ is an internal endofunctor (considering the category of elements $A : \mathsf{U}$ and maps as functions $\mathsf{El}(A) \to \mathsf{El}(B)$).*

*Proof.* □

**Theorem 2.3.8.** $\mathsf{Str}(A)$ *is the final coalgebra for $B \mapsto A \times B$.*

*Proof.* First, given some $A : \mathsf{U}$, we must construct a map $\mathsf{uncons} : \mathsf{Str}(A) \to (A \times \mathsf{Str}(A))$. We can construct this map as follows:

$$\mathsf{uncons}(s) \triangleq (\mathsf{head}_A(s), \mathsf{tail}_A(s))$$

We must now show that this coalgebra is final. For this, suppose that we have some $B$ and $b : B \to \mathsf{El}(A) \times B$. We start by constructing a map of coalgebras $b \to \mathsf{uncons}$. This must be a function $f : B \to \mathsf{Str}(A)$ which satisfies the following equation:

$$\mathsf{uncons}(f(x)) = (\mathsf{pr}_0(b(x)), f(\mathsf{pr}_1(b(x))))$$

We will define $f$ as follows:

$$
\begin{aligned}
&f' &&: \Delta(B \to \mathsf{El}(A) \times B) \to \Delta B \to \mathsf{El}(\mathsf{Str}'(A)) \\
&f'(b) &&\triangleq \mathsf{let}\ \mathsf{mod}_\delta(b') \leftarrow b \ \mathsf{in}\ \mathsf{l\ddot{o}b}(\lambda f', x.\ (h, t)) \\
& && \quad \text{where } h = \mathsf{let}\ \mathsf{mod}_\delta(x') \leftarrow x \ \mathsf{in}\ \mathsf{mod}_\delta(\mathsf{pr}_0(b'(x))) \\
& && \quad\ \text{and } t = \mathsf{let}\ \mathsf{mod}_\delta(x') \leftarrow x \ \mathsf{in}\ f' \circledast_\ell \mathsf{next}(\mathsf{mod}_\delta(\mathsf{pr}_1(b'(x)))) \\
&f &&: B \to \mathsf{El}(\mathsf{Str}(A)) \\
&f(x) &&\triangleq \mathsf{mod}_\gamma(f'(\mathsf{mod}_\delta(b), \mathsf{mod}_\delta(x)))
\end{aligned}
$$

This can be shown to satisfy the equation for a morphism of coalgebras by more or less direct computation. Suppose that we have some $x : B$:

$$
\begin{aligned}
\mathsf{uncons}(f(x)) &= (\mathsf{head}_A(f(x)), \mathsf{tail}_A(f(x))) \\
&= (\mathsf{pr}_0(b(x)), \mathsf{tail}_A(f(x))) \\
&= (\mathsf{pr}_0(b(x)), \mathbf{comp}_{\ell,\gamma}(\mathsf{next}(\mathsf{l\ddot{o}b}(\ldots)) \circledast_\ell \mathsf{next}(\mathsf{mod}_\delta(\mathsf{pr}_1(b'(x)))))) \\
&= (\mathsf{pr}_0(b(x)), \mathbf{comp}_{\ell,\gamma}(\mathsf{next}(f(\mathsf{pr}_1(b(x)))))) \\
&= (\mathsf{pr}_0(b(x)), f(\mathsf{pr}_1(b(x))))
\end{aligned}
$$

Finally, we must show that $f$ is unique with this property. This is essentially a corollary of Theorem 2.3.5 because we can phrase the property of being a coalgebra as being a solution to a guarded fixed point. □

We conclude this section by showing how we can actually use these mechanisms to prove properties of coinductive programs. Specifically, we will replicate proof from Bizjak et al. [Biz+16] which shows that the `zipWith` operator on streams preserves commutativity. We start by defining the `zipWith` function:

$$\begin{aligned}
&\mathsf{zipWith}' && : \Delta(\mathsf{El}(A) \to \mathsf{El}(B) \to \mathsf{El}(C)) \to \mathsf{El}(\mathsf{Str}'(A)) \to \mathsf{El}(\mathsf{Str}'(B)) \to \mathsf{El}(\mathsf{Str}'(C)) \\
&\mathsf{zipWith}'(f) && \triangleq \text{löb}(\lambda r.\ \lambda x, y.\ (f \circledast_\delta \mathsf{pr}_0(x) \circledast_\delta \mathsf{pr}_0(y), r \circledast_\ell \mathsf{pr}_1(x) \circledast_\ell \mathsf{pr}_1(y)))
\end{aligned}$$

$$\begin{aligned}
&\mathsf{zipWith} && : (\mathsf{El}(A) \to \mathsf{El}(B) \to \mathsf{El}(C)) \to \mathsf{El}(\mathsf{Str}(A)) \to \mathsf{El}(\mathsf{Str}(B)) \to \mathsf{El}(\mathsf{Str}(C)) \\
&\mathsf{zipWith}(f) && \triangleq \lambda x, y.\ \mathsf{mod}_\gamma(\mathsf{zipWith}'(\mathsf{mod}_\delta(f))) \circledast_\gamma x \circledast_\gamma y
\end{aligned}$$

This is a common pattern when programming in this implementation of guarded recursion: we have a helper function which lives in mode $t$, uses Löb induction, and performs the majority of the work. Then, on top of this, the main function is just a thin wrapper which takes care of the modal plumbing.

**Theorem 2.3.9.** *If $f$ is commutative then $\mathsf{zipWith}(f)$ is commutative. That is, given $A, B : \mathsf{U}$ and $f : \mathsf{El}(A) \to \mathsf{El}(A) \to \mathsf{El}(B)$ there is a term of the following type:*

$$\begin{aligned}
&((a_0, a_1 : \mathsf{El}(A)) \to \mathsf{Eq}_{\mathsf{El}(B)}(f(a_0, a_1), f(a_1, a_0))) \to \\
&\qquad (s_0, s_1 : \mathsf{El}(\mathsf{Str}(A))) \to \mathsf{Eq}_{\mathsf{El}(\mathsf{Str}(B))}(\mathsf{zipWith}(f, s_0, s_1), \mathsf{zipWith}(f, s_1, s_0))
\end{aligned}$$

*Proof.* Let us suppose that we have $e : (a_0, a_1 : \mathsf{El}(A)) \to \mathsf{Eq}_{\mathsf{El}(B)}(f(a_0, a_1), f(a_1, a_0))$ as well as $s_0, s_1 : \mathsf{El}(\mathsf{Str}(A))$. We wish to show that the term $\mathsf{refl}(\mathsf{zipWith}(f, s_0, s_1))$ is the desired proof. We will do this by showing that $\mathsf{zipWith}(f, s_0, s_1)$ is convertible with $\mathsf{zipWith}(f, s_1, s_0)$. We will freely make use of equality reflection throughout this proof.

For a first step, we note is clearly sufficient to construct a term of the following type (and then to reflect it):

$$(t_0, t_1 : \mathsf{El}(\mathsf{Str}'(A))) \to \mathsf{Eq}_{\mathsf{El}(\mathsf{Str}(B))}(\mathsf{zipWith}'(\mathsf{mod}_\delta(f), t_0, t_1), \mathsf{zipWith}'(\mathsf{mod}_\delta(f), t_1, t_0))$$

In order to construct this term, it is sufficient to prove the following equality:

$$\begin{aligned}
&\text{löb}(\lambda r.\ \lambda x, y.\ (f \circledast_\delta \mathsf{pr}_0(x) \circledast_\delta \mathsf{pr}_0(y), r \circledast_\ell \mathsf{pr}_1(x) \circledast_\ell \mathsf{pr}_1(y))) = \\
&\qquad\qquad \text{löb}(\lambda r.\ \lambda x, y.\ (f \circledast_\delta \mathsf{pr}_0(y) \circledast_\delta \mathsf{pr}_0(x), r \circledast_\ell \mathsf{pr}_1(y) \circledast_\ell \mathsf{pr}_1(x)))
\end{aligned}$$

We will write $\text{löb}(F_0)$ for the left hand side of this equation and $\text{löb}(F_1)$ for the right.

Now, using Theorem 2.3.5, it then suffices to show that $\text{löb}(F_1)$ a fixed-point of $F_0$::

$$\text{löb}(F_1) = F_0(\mathsf{next}(\text{löb}(F_1))) \tag{2.6}$$

We will now use löb to construct a term in $\mathsf{Eq}(\text{löb}(F_1), F_0(\mathsf{next}(\text{löb}(F_1))))$.

$$\begin{aligned}
F_0(\mathsf{next}(\text{löb}(F_1))) &= \lambda x, y.\ (f \circledast_\delta \mathsf{pr}_0(x) \circledast_\delta \mathsf{pr}_0(y), \mathsf{next}(\text{löb}(F_1)) \circledast_\ell \mathsf{pr}_1(x) \circledast_\ell \mathsf{pr}_1(y)) \\
&= \lambda x, y.\ (f \circledast_\delta \mathsf{pr}_0(y) \circledast_\delta \mathsf{pr}_0(x), \mathsf{next}(\text{löb}(F_1)) \circledast_\ell \mathsf{pr}_1(x) \circledast_\ell \mathsf{pr}_1(y)) \\
&= \lambda x, y.\ (f \circledast_\delta \mathsf{pr}_0(y) \circledast_\delta \mathsf{pr}_0(x), \mathsf{next}(F_0(\mathsf{next}(\text{löb}(F_1)))) \circledast_\ell \mathsf{pr}_1(x) \circledast_\ell \mathsf{pr}_1(y)) \\
&\qquad\qquad \text{Using induction to replace } \mathsf{pr}_1(x) \text{ and } \mathsf{pr}_1(y) \text{ with } \mathsf{mod}_\ell(t_0') \text{ and } \mathsf{mod}_\ell(t_1') \\
&= \lambda x, y.\ (f \circledast_\delta \mathsf{pr}_0(y) \circledast_\delta \mathsf{pr}_0(x), \mathsf{next}(F_0(\mathsf{next}(\text{löb}(F_1)))) \circledast_\ell \mathsf{mod}_\ell(t_0') \circledast_\ell \mathsf{mod}_\ell(t_1')) \\
&= \lambda x, y.\ (f \circledast_\delta \mathsf{pr}_0(y) \circledast_\delta \mathsf{pr}_0(x), \mathsf{next}(F_0(\mathsf{next}(\text{löb}(F_1)), t_0', t_1'))) \\
&= \lambda x, y.\ (f \circledast_\delta \mathsf{pr}_0(y) \circledast_\delta \mathsf{pr}_0(x), \mathsf{next}(F_1(\mathsf{next}(\text{löb}(F_1)), t_1', t_0'))) \\
&= \lambda x, y.\ (f \circledast_\delta \mathsf{pr}_0(y) \circledast_\delta \mathsf{pr}_0(x), \mathsf{next}(F_1(\mathsf{next}(\text{löb}(F_1)))) \circledast_\ell \mathsf{mod}_\ell(t_1') \circledast_\ell \mathsf{mod}_\ell(t_0')) \\
&= \lambda x, y.\ (f \circledast_\delta \mathsf{pr}_0(y) \circledast_\delta \mathsf{pr}_0(x), \mathsf{next}(F_1) \circledast_\ell \mathsf{pr}_1(y) \circledast_\ell \mathsf{pr}_1(x)) \\
&= \text{löb}(F_1)
\end{aligned}$$

□

## 2.4 *Degrees of Relatedness*

Of all type systems present in the literature, the most similar to MTT is probably that of Degrees of Relatedness [ND18]. In Section 2.4.1, we discuss at a conceptual level how Reynolds' original formulation of parametricity [Rey83] was gradually generalized to dependent types. In Section 2.4.2, we explain how modalities can help to validate the identity extension lemma for large types [NVD17]. In Section 2.4.3, we discuss Degrees of Relatedness proper, and in Section 2.4.4, we consider how MTT can serve as an internal language in which one could build a model of Degrees of Relatedness.

### 2.4.1 *Parametricity, from System F to dependent types*

We discuss parametricity in System F [Rey83], System F$\omega$ [Atk12], and dependent type theory [AGJ14; BCM15; Mou16].

#### *System F*

System F is relationally parametric [Rey83]. If we think of proof-irrelevant relations $R : \mathrm{Rel}(A, B)$ as notions of heterogeneous equality between elements of $A$ and elements of $B$, and write $a \asymp_R b$ for $R(a, b)$ in order to emphasize this perspective, then we can conceptually describe proof-relevant relational parametricity as follows:

- Type-level operations $F : * \to *$ preserve (meta-theoretical) equality,[3]
- Type-level operations $F : * \to *$ preserve relations,
  - sending the equality relation on $X$ to the equality relation on $FX$ (this is the identity extension lemma),
- Parametric functions $f : \forall X.FX$ send relations to proofs of heterogeneous equality,
- Term-level operations $hX : FX \to GX$ preserve heterogeneous equality.

The identity extension lemma asserts that our use of the name 'heterogeneous equality' is sensible: in the homogeneous case, it boils down to mathematical equality.

We can represent this diagrammatically as follows:

$$\begin{tikzcd}
A = B \arrow[r, "F_="] \arrow[d, "\mathrm{Eq}"'] & FA = FB \arrow[d, "\mathrm{Eq}"] \\
\mathrm{Rel}(A, B) \arrow[r, "F_{\mathrm{Rel}}"'] & \mathrm{Rel}(FA, FB)
\end{tikzcd}
\qquad
\begin{tikzcd}
A = B \arrow[d, "\mathrm{Eq}"'] & & fA \asymp_{F_{\mathrm{Rel}} R} fB \\
R : \mathrm{Rel}(A, B) \arrow[urr, "f_{\mathrm{Rel}}"] & &
\end{tikzcd}$$

$$a \asymp_{F_{\mathrm{Rel}} R} b \xrightarrow{h_{\mathrm{Rel}} R} hAa \asymp_{G_{\mathrm{Rel}} R} hBb$$

These diagrams are a bit awkward in the sense that some of their nodes are meta-theoretic propositions whereas others are meta-theoretic sets. For example, the arrow $\mathrm{Eq} : A = B \to \mathrm{Rel}(A, B)$ is to be read as: if $A$ and $B$ are really the same thing, then $\mathrm{Eq}$ will pick out a relation $\mathrm{Eq}_A$ between $A$ and $B$, namely

[3] A sane meta-theory will not allow the creation of anything that doesn't.

the identity relation. Set theorists who do not balk at dealing with large objects, may prefer to write this as $\mathrm{Eq} \in \prod_A \mathrm{Rel}(A, A)$. Commutativity of the diagram simply means that $F_{\mathrm{Rel}}\mathrm{Eq}_A = \mathrm{Eq}_{FA}$.

The arrow $f_{\mathrm{Rel}} : (R : \mathrm{Rel}(A, B)) \to fA \asymp_{F_{\mathrm{Rel}}R} fB$ means: for any relation $R : \mathrm{Rel}(A, B)$, the relation $F_{\mathrm{Rel}}R$ will relate $fA$ and $fB$. This would be more typically written as $\forall(R \in \mathrm{Rel}(A, B)).(fA, fB) \in F_{\mathrm{Rel}}R$.

An alternative way to make sense of the diagram is by translating every proposition $P$ to the subsingleton $\{*|P\}$.

### *System Fω*

Atkey [Atk12] extends Reynolds' ideas to System F$\omega$. Every kind $\kappa$ is equipped with a 'native' proof-relevant relation $\frown_\kappa : \kappa \times \kappa \to \mathbf{Set}$, such that $\frown_* = \mathrm{Rel}$.[4] We say that $K_1, K_2 : \kappa$ are **related** if we can give an element of $K_1 \frown_\kappa K_2$.[5] Similarly, for every $K : \kappa$, we get a proof $\mathrm{refl}(K) : K \frown_\kappa K$ such that for $X : *$ we get $\mathrm{refl}(X) = \mathrm{Eq}_X : \mathrm{Rel}(X, X)$. We can then generalize our description of relational parametricity:

- Type-level operations $F : \theta \to \kappa$ preserve equality,
- Type-level operations $F : \theta \to \kappa$ preserve relatedness,
  - sending $\mathrm{refl}(X)$ to $\mathrm{refl}(FX)$ (this is the identity extension lemma),
- Parametric functions $f : \forall(X : \kappa).FX$ send related types to heterogeneously equal terms,
- Term-level operations $hX : FX \to GX$ preserve heterogeneous equality.

Diagrammatically (the same interpretation remarks apply as for the System F diagrams above):

$$\begin{tikzcd}
A =_\theta B \arrow[r, "F_="] \arrow[d, "\mathrm{refl}"'] & FA =_\kappa FB \arrow[d, "\mathrm{refl}"] \\
A \frown_\theta B \arrow[r, "F_\frown"'] & FA \frown_\kappa FB
\end{tikzcd}
\qquad
\begin{tikzcd}
A =_\kappa B \arrow[d, "\mathrm{refl}"'] & fA \asymp_{F_\frown R} fB \\
R : A \frown_\kappa B \arrow[ur, "f_\frown"] &
\end{tikzcd}$$

$$a \asymp_{F_\frown R} b \xrightarrow{h_\frown R} hAa \asymp_{G_\frown R} hBb$$

Following Robinson and Rosolini [RR94] and Hasegawa [Has94a; Has94b], Atkey structured all of this in a reflexive graph model. A reflexive graph $\Gamma$ is a (contravariant) presheaf over the category $\mathbf{RG}$ generated by the following diagram, subject to the following equations:[6]

$$\begin{tikzcd}
\mathbf{n} \arrow[r, bend left, "\mathbf{s}"] \arrow[r, bend right, "\mathbf{t}"'] & \mathbf{e} \arrow[l, "\mathbf{r}" description]
\end{tikzcd}
\qquad
\begin{aligned}
\mathbf{r} \circ \mathbf{s} &= 1_{\mathbf{n}}, \\
\mathbf{r} \circ \mathbf{s} &= 1_{\mathbf{n}}.
\end{aligned}$$

[4] We ignore size issues in this introductory exposition.

[5] Note that any two types $T_1, T_2 : *$ are related. However, as $\frown_*$ is a proof-relevant relation, we care not only for the truth value (whether types are related) but also for the particular proof we choose to give (the relation $R : T_1 \frown_* T_2$ that we consider between $T_1$ and $T_2$).

[6] Readers who expected the opposite of $\mathbf{RG}$ are likely thinking of *covariant* functors to $\mathbf{Set}$, whereas we take presheaves to be contravariant functors to $\mathbf{Set}$.

The idea is that $\Gamma\mathbf{n}$ is the set of nodes, $\Gamma\mathbf{e}$ is the set of edges, and that $(-)\mathbf{s}, (-)\mathbf{t} : \Gamma\mathbf{e} \to \Gamma\mathbf{n}$ extract the source and target of an edge, whereas $(-)\mathbf{r} : \Gamma\mathbf{n} \to \Gamma\mathbf{e}$ produces the reflexive edge on a node. The equations assert that the edge $x\mathbf{r}$ really goes from $x$ to $x$.

In this reflexive graph model, kinds $\kappa$ are interpreted as large reflexive graphs $[\![\kappa]\!]$. The nodes in $[\![\kappa]\!]\mathbf{n}$ are the semantic elements of $\kappa$, whereas the edges in $[\![\kappa]\!]\mathbf{e}$ can be seen as a triple of two elements $K_1, K_2 : \kappa$ wrapped up with a proof of $K_1 \frown_\kappa K_2$. The kind $*$ specifically is interpreted as the reflexive graph $[\![*]\!]$ whose nodes are small sets and whose edges are proof-irrelevant relations, the reflexive edges being the equality relations. An open type $\Gamma \vdash K : \kappa$ is then a reflexive graph morphism (i.e. a presheaf morphism) $[\![K]\!] : [\![\Gamma]\!] \to [\![\kappa]\!]$. The fact that these preserve reflexive edges (for $*$ this means the equality relation), expresses the identity extension lemma.

This means that a closed type $\cdot \vdash T : *$ is essentially a small discrete reflexive graph, i.e. a small reflexive graph whose only edges are the reflexive ones. To see this, note that the empty context is interpreted as the terminal reflexive graph $[\![\cdot]\!]$, having a single node $\bullet$ and a single reflexive edge $\bullet\mathbf{r}$. This node $\bullet$ is then mapped to a small set $[\![T]\!]\bullet$, and the edge $\bullet\mathbf{r}$ to a relation $[\![T]\!](\bullet\mathbf{r})$ on that set. However, since graph morphisms map reflexive edges to reflexive edges, and reflexive edges in $[\![*]\!]$ are the equality relations, we see that

$$[\![T]\!](\bullet\mathbf{r}) = ([\![T]\!]\bullet)\mathbf{r} = \mathrm{Eq}_{[\![T]\!]\bullet}$$

i.e. $[\![T]\!]$ is a set equipped with its equality relation.

A general (open) type $\Gamma \vdash T : *$ can be reorganized to be seen as a reflexive graph $[\![\Gamma|T]\!] \to [\![\Gamma]\!]$ over $[\![\Gamma]\!]$ that lifts reflexive edges (i.e. edges over reflexive edges are reflexive, this is again the identity extension lemma), and equality of edges (i.e. edges over the same edge in $[\![\Gamma]\!]$ are necessarily equal, expressing prove irrelevance). A term $\Gamma \mid \Theta \vdash t : T$ is then interpreted as a morphism from $[\![\Gamma|\Theta]\!] \to [\![\Gamma|T]\!]$ in the slice category over $[\![\Gamma]\!]$; in particular a closed term is a section.

*Dependent type theory*

As dependent type theory is not just a programming language but also a logic, we can distinguish *three* approaches to parametricity:

- In the external approach, we state *and* prove parametricity theorems in the meta-theory. This is the only possible approach in System F and F$\omega$.
- In the admissible approach, we state the parametricity theorems in some very similar (ideally the same) type system, and we give a metatheoretic proof that every program is parametric. That is, we give a meta-theoretic function that maps program derivation trees to derivation trees of proofs of the statement that the program is parametric.
- In the internal approach, we have an internal operator that essentially inhabits the theorem "every program is parametric". This operator will again have type dependencies, and self-application should prove that it is parametric. This phenomenon is called iterated parametricity, and generally needs to be modelled in higher-dimensional reflexive graphs, i.e. cubical sets.

*External parametricity, with identity extension only for small types* Atkey, Ghani, and Johann [AGJ14] have reorganized Atkey's [Atk12] model to a model of dependent type theory. Essentially, they start from the standard presheaf model of dependent type theory in reflexive graphs [Hof97, Ch. 4] (see Section 2.2 for a summary). The idea is that nodes of large types (kinds) represent their elements, whereas edges represent proofs of relatedness ($\frown_\kappa$). For small types, nodes are again elements, but edges should be proofs of heterogeneous equality ($\asymp_R$, where $R$ is the corresponding edge between the types).

The desired identity extension lemma can now be rephrased as: homogeneous edges (edges living above reflexive edges in the context) should be reflexive. In order to validate this lemma, we could naively require all internal types to be discrete, i.e. to satisfy this condition. However, the problem is that the universe does *not* satisfy it. Indeed, a homogeneous edge in the universe is like a proof of $A \frown_* B$ in System F$\omega$, which is essentially a relation between $A$ and $B$. Surely, the existence of a relation $R : \mathrm{Rel}(A, B)$ does not imply that $A = B$ and $R = \mathrm{Eq}_A$. So the universe is not discrete as it has non-reflexive homogeneous edges.

For this reason, Atkey, Ghani, and Johann [AGJ14] only adapt the Hofmann-Streicher universe of small types [HS97] by restricting it to small discrete proof-irrelevant types. Types in general are allowed to be non-discrete, and hence identity extension is only proven for small types. Proof-irrelevance is required in order to model function types: for function types to be discrete, we either need to work in proof-irrelevant graphs, or we need higher-dimensional structure (cubical sets) in order to reason about equality of functions' actions on edges.

Writing $e : x \div_A y$ for a homogeneous edge in type $A$ (which generalizes both $e : x \frown_A y$ and $e : x \asymp_A y$), and $e : x \div_R y$ for a heterogeneous edge where $R : A \div_{\mathsf{U}} B$, we can summarize the behaviour of dependent functions $f : (x : A) \to B(x)$ in Atkey, Ghani, and Johann [AGJ14] in a single diagram:[7]

$$\begin{tikzcd} x =_A y \arrow[r, "f_="] \arrow[d, "(-)\mathbf{r}"'] & f(x) =_{B(x)} f(y) \arrow[d, "(-)\mathbf{r}"] \\ e : x \div_A y \arrow[r, "f_\div"'] & f(x) \div_{B_\div(e)} f(y) \end{tikzcd}$$

In this diagram, $=_A$ denotes mathematical equality. E.g. the arrow $(-)\mathbf{r} : x =_A y \to x \div_A y$ means: if $x$ and $y$ are really the same node of $A$, then $x\mathbf{r}$ is an edge of $A$ whose source $x\mathbf{rs}$ equals $x$ and whose target $x\mathbf{rt}$ equals $y$.

*Admissible parametricity* The work on admissible parametricity generally uses different techniques and is in this sense much less relevant in this historical resume. We cite some important works for completeness:

- Takeuti [Tak01] gives a parametric translation from every system in the Lambda Cube to a richer system in the Lambda Cube, and proves soundness of identity extension (calling it the "axiom of parametricity") for small types.
- Bernardy, Jansson, and Paterson [BJP12] give a parametric translation from a general pure type system to (in general) a different pure type system. Identity extension is not considered.
- Keller and Lasson [KL12] give a parametric translation from a variation of the calculus of inductive constructions to itself. They use this as a basis to implement the `paramcoq` plugin for Coq.[8] Identity extension is not considered.

*Internal parametricity, without identity extension* Bernardy, Coquand, and Moulin [BCM15] and Moulin [Mou16] have introduced internal operators that allow the creation of proof terms for parametricity theorems, and provide a model in (unary) cubical sets.

[7] The bottom arrow in this diagram can be generalized to act on heterogeneous edges (by replacing $A$ with an edge in the universe); however then the left side of the diagram would be ill-typed. Dependent diagrams are always a bit awkward.

[8] https://github.com/coq-community/paramcoq

Their system is about unary parametricity and hence cannot feature identity extension, but it can be converted to a binary system straightforwardly in which we could either postulate the identity extension lemma for small types as an axiom[9], or create a universe of types that satisfy the lemma and is closed under small type formers.

(Binary) cubical sets can be seen as higher-dimensional graphs, which feature not just nodes and edges, but also squares, cubes, and higher-dimensional cubes. This higher-dimensional structure is necessary to model iterated parametricity (see above), as well as to prove the identity extension lemma for the function type if you want to allow parametricity to be applied to proof-relevant relations.

### 2.4.2 Parametric Quantifiers: internal parametricity with identity extension

*Motivation*

In order to validate the identity extension lemma for all types, rather than just small types, Nuyts, Vezzosi, and Devriese [NVD17] create a type system ParamDTT that uses modalities to distinguish between parametric, continuous and pointwise functions. These modalities differ in how they act on different flavours of edges:

- **Paths** $p : x \asymp y$ generalize equality of types and heterogeneous equality of terms in System F$\omega$,
- **Bridges** $b : x \frown y$ generalize relatedness of types.

One might hope to give a model in bridge/path reflexive graphs, which would be presheaves over the category $\mathbf{BPRG}$ generated by the following diagram and equations:

$$\begin{tikzcd}
\mathbf{n} \arrow[rr, bend left, "\mathbf{s}"] \arrow[rr, bend right, "\mathbf{t}"'] & \mathbf{p} \arrow[l, "\mathbf{r}"'] & \mathbf{b} \arrow[l, "\mathbf{u}"']
\end{tikzcd}$$

$$\begin{aligned}
\mathbf{r} \circ \mathbf{u} \circ \mathbf{s} &= 1_{\mathbf{n}}, \\
\mathbf{r} \circ \mathbf{u} \circ \mathbf{t} &= 1_{\mathbf{n}}.
\end{aligned}$$

Here, $\mathbf{r}$ expresses that every node is path-equal to itself, $\mathbf{u}$ expresses that when things are path-equal, they are also bridge-related, and $\mathbf{s}$ and $\mathbf{t}$ extract source and target from a bridge. By composing with $\mathbf{u}$, we can extract source and target of a path, or obtain reflexive bridges.

However, because the bridges in the universe — which will be relations between types — are inherently proof-relevant, we need a model that accommodates proof-relevant parametricity. Furthermore, because the aim is to provide internal parametricity operators, it is desirable to accommodate iterated parametricity. For these two reasons, we need a cubical model. Indeed, ParamDTT is modelled in bridge/path cubical sets, which are presheaves over the category $\mathbf{BPCube}$ which is the free cartesian monoidal category over $\mathbf{BPRG}$ with the same terminal object $\mathbf{n}$. In other words, the objects of $\mathbf{BPCube}$ are finite products of $\mathbf{b}$ and $\mathbf{p}$ and the morphisms are generated by weakening ($\mathbf{r} : \mathbf{p} \to ()$), exchange ($v \times w \to w \times v$), contraction ($w \to w \times w$) and $\mathbf{u} : \mathbf{b} \to \mathbf{p}$.

Functions $f : (x : A) \to B(x)$ in ParamDTT are then classified according to how they act on bridges and paths.

[9] This axiom would be partly justified by a cubical generalization of Atkey, Ghani, and Johann's [AGJ14] model, but Moulin's [Mou16] model is more subtle in that it uses *refined* presheaves (based on $I$-sets) to strictify certain isomorphisms related to the internal parametricity operators. To our knowledge no one has created a refined presheaf model of identity extension.

A **pointwise** function $f : (x : (\mathbf{ptw} \mid A)) \to B(x)$ maps path-connected inputs $p : x \asymp y$ to path-connected outputs $f_{\asymp}(p) : f(x) \asymp f(y)$, witnessing that it preserves heterogeneous equality. However, it has no action on bridges $b : x \frown y$, meaning that bridge-related inputs may be mapped to arbitrary outputs. In particular, pointwise quantification over types has no action on relations between types. The only way to assert bridge-connected outputs from a pointwise function, is by feeding it path-connected inputs; then $f_{\asymp}(p)\mathbf{u}$ is the desired bridge. The pointwise modality may be used to soundly assume the law of excluded middle:

$$(X : (\mathbf{ptw} \mid \mathsf{U})) \to X \uplus (X \to \bot)$$

Stating it with the parametric modality would imply that either all types are inhabited or all types are empty.

A **continuous** function $f : (x : (\mathbf{con} \mid A)) \to B(x)$ sends path-connected inputs $p : x \asymp y$ to path-connected outputs $f_{\asymp}(p) : f(x) \asymp f(y)$, and bridge-connected inputs $b : x \frown y$ to bridge-connected outputs $f_{\frown}(b) : f(x) \frown f(y)$. Thus, it preserves heterogeneous equality *and* relatedness. This corresponds to the behaviour of a type-level operation in System F$\omega$.

A **parametric** function $f : (x : (\mathbf{par} \mid A)) \to B(x)$ sends bridge-connected inputs $b : x \frown y$ to path-connected outputs $f_{\frown}(b) : f(x) \asymp f(y)$. Hence, it also sends paths $p : x \asymp y$ to paths $f_{\frown}(p\mathbf{u}) : f(x) \asymp f(y)$ and bridges $b : x \frown y$ to bridges $f_{\frown}(b)\mathbf{u} : f(x) \frown f(y)$. In particular, a function $f : (X : (\mathbf{par} \mid \mathsf{U})) \to T(X)$ sends a relation $B : X \frown Y$ to a proof $f_{\frown}(B) : f(X) \asymp f(Y)$ that the instantiations $f(X)$ and $f(Y)$ are heterogeneously equal according to the relation $T_{\frown}(B) : T(X) \frown T(Y)$.[10]

$$\begin{tikzcd}
x \asymp y \arrow[r, "f_{\asymp}"] \arrow[d, "(-)\mathbf{u}"'] & f(x) \asymp f(y) \arrow[d, "(-)\mathbf{u}"] \\
x \frown y & f(x) \frown f(y)
\end{tikzcd}$$

$$\boxed{\mathbf{ptw}}$$

$$\begin{tikzcd}
x \asymp y \arrow[r] \arrow[d, "(-)\mathbf{u}"'] & f(x) \asymp f(y) \arrow[d, "(-)\mathbf{u}"] \\
x \frown y \arrow[r] & f(x) \frown f(y)
\end{tikzcd}$$

$$\boxed{\mathbf{con}}$$

$$\begin{tikzcd}
x \asymp y \arrow[d, "(-)\mathbf{u}"'] & f(x) \asymp f(y) \arrow[d, "(-)\mathbf{u}"] \\
x \frown y \arrow[ur] & f(x) \frown f(y)
\end{tikzcd}$$

$$\boxed{\mathbf{par}}$$

**Remark 2.4.1.** We remark that Vezzosi's ParamDTT implementation `agda-parametric` [NVD17] features three additional and at the time experimental modalities, for which we need to include a trivially satisfied relation sending $x$ and $y$ to the singleton $\top$: irrelevance ($\mathbf{irr}$), shape-irrelevance ($\mathbf{shi}$),

[10] The codomain $T$ is required to be continuous for the parametric function type to be well-formed.

and the join of shape-irrelevance and parametricity (**shi** $\vee$ **par**):

$$\begin{tikzcd}
x \asymp y \arrow[d, "(-)\mathbf{u}"'] & f(x) \asymp f(y) \arrow[d, "(-)\mathbf{u}"] \\
x \frown y \arrow[d] & f(x) \frown f(y) \arrow[d] \\
\top \arrow[uur] & \top
\end{tikzcd}
\qquad
\begin{tikzcd}
x \asymp y \arrow[r] \arrow[d, "(-)\mathbf{u}"'] & f(x) \asymp f(y) \arrow[d, "(-)\mathbf{u}"] \\
x \frown y \arrow[d] & f(x) \frown f(y) \arrow[d] \\
\top \arrow[ur] & \top
\end{tikzcd}
\qquad
\begin{tikzcd}
x \asymp y \arrow[d, "(-)\mathbf{u}"'] & f(x) \asymp f(y) \arrow[d, "(-)\mathbf{u}"] \\
x \frown y \arrow[d] \arrow[ur] & f(x) \frown f(y) \arrow[d] \\
\top \arrow[ur] & \top
\end{tikzcd}$$

$\boxed{\mathbf{irr}}$ $\qquad$ $\boxed{\mathbf{shi}}$ $\qquad$ $\boxed{\mathbf{shi} \vee \mathbf{par}}$

$\lhd$

*The Mode Theory and the Corresponding Instance of MTT*

**Definition 2.4.2.** The mode theory for ParamDTT is the poset-enriched category

- that has a single object $*$,
- such that $\mathrm{Hom}(*, *) = \{\mathbf{ptw} < \mathbf{con} < \mathbf{par}\}$,
- where **con** is the identity and composition is given by

| $\downarrow \circ \rightarrow$ | **ptw** | **con** | **par** |
|---|---|---|---|
| **ptw** | **ptw** | **ptw** | **par** |
| **con** | **ptw** | **con** | **par** |
| **par** | **ptw** | **par** | **par**. |

It is clear that the identity function is continuous. The modality of a composite function, can be found by pasting together the above diagrams, which yields the above composition table.

Note also that, using $(-)\mathbf{u}$, we can prove that all parametric functions are continuous. All continuous functions are clearly also pointwise (as we can forget the action on bridges), which confirms the postulated order on modalities.

**Theorem 2.4.3.** *The instantiation of MTT with the mode theory for ParamDTT yields a type system ParamMTT which can be modelled in the category* $\mathbf{PSh}(\mathbf{BPCube})$ *as an instance of Section 2.2. ParamMTT is* not *the system ParamDTT [NVD17].*

**Remark 2.4.4.** ParamDTT deviates from MTT in two important respects:

- It uses eager left division $\mu \setminus \Gamma$, rather than lazy locks $\Gamma, 🔒_\mu$ (see Section 1.8.2),
- It features a parametric type decoding rule

$$\frac{\mathbf{par} \setminus \Gamma \vdash T : \mathsf{U} \mathbin{@} *}{\Gamma \vdash T \ \mathsf{type}_\ell \mathbin{@} *} \tag{2.7}$$

which has the effect of making variables available in a term and its type (or more precisely its type's code) by a different modality (e.g. parametric functions have continuous type).

Furthermore, it lacks all system-specific features, such as internal parametricity operators. $\triangleleft$

**Lemma 2.4.5.** *We have three adjoint functors* $\int \dashv \flat \dashv \sharp : \mathbf{BPCube} \to \mathbf{BPCube}$ *which are the cartesian monoidal functors such that:*

$$\begin{array}{lll} \int\mathbf{p} = (), & \flat\mathbf{p} = \mathbf{b}, & \sharp\mathbf{p} = \mathbf{p}, \\ \int\mathbf{b} = \mathbf{b}, & \flat\mathbf{b} = \mathbf{b}, & \sharp\mathbf{b} = \mathbf{p}. \end{array}$$

*Proof.* Left as an exercise to the reader, but note that these functors are definable on $\mathbf{BPRG}$ and that the adjunctions can be proven there and carry over. $\square$

*Proof of Theorem 2.4.3.* We need to find a functor $J : \mathcal{M} \to \mathbf{Cat}$ where $\mathcal{M}$ is the mode theory for ParamDTT, such that $J(\mu)_*$ is a good interpretation of the dependent right adjoint. Clearly, we will take $J(*) = \mathbf{BPCube}$.

Before we define the action of $J$ on morphisms, we will define $K : \mathcal{M}^{\mathsf{coop}} \to \mathbf{Cat}$, and then we will construct $J$ so that $J(\mu) \dashv K(\mu)$. This means that $K(\mu)^*$ will be naturally isomorphic to $J(\mu)_*$. Of course all of this is only well typed assuming we take $K(*) = J(*) = \mathbf{BPCube}$.

In general, $K(\mu)\mathbf{b}$ should be the weakest relation (represented by an object of $\mathbf{BPCube}$) such that a $\mu$-modal function will send $K(\mu)\mathbf{b}$-related inputs to bridge-related outputs. Similarly, $K(\mu)\mathbf{p}$ should be the weakest relation such that a $\mu$-modal function will send $K(\mu)\mathbf{p}$-related inputs to path-equal outputs.

For $\mathbf{con}$, which is the identity modality, this means $K(\mathbf{con}) = 1$. For parametricity, a bridge in the domain is sufficient to guarantee either a path or a bridge in the codomain, so we take $K(\mathbf{par}) = \flat$. For pointwise functions, we need a path in the domain to guarantee either a path or a bridge in the codomain, so we take $K(\mathbf{ptw}) = \sharp$. This is immediately seen to reverse 2-cells.

For $J$ then, we simply take the left adjoints:

$$J(\mathbf{con}) = 1, \qquad J(\mathbf{par}) = \int, \qquad J(\mathbf{ptw}) = \flat.$$ $\square$

Let us now map concepts from System F$\omega$ to those of ParamDTT by looking for similarities between the corresponding diagrams. Type level operators in F$\omega$ become continuous functions in ParamDTT. Parametric functions in System F$\omega$ become parametric functions in ParamDTT. One can imagine a modal extension of System F$\omega$ that allows ad hoc polymorphism, so that we can have a `typecase` operator or postulate a non-parametric law of excluded middle. The latter is sound in ParamDTT.

When we consider term level functions in System F$\omega$, we notice an awkward aspect of the model of ParamDTT, namely that small types, too, come equipped with a path ($\asymp$) and a bridge ($\frown$) relation. In System F$\omega$ on the other hand, we could only consider heterogeneous equality ($\asymp$) for elements of small types. In fact, we have no need for these two relations, and unless we allow HITs with bridge constructors, all small closed types will be bridge-discrete, meaning essentially that $(-)\mathbf{u}$ is an isomorphism. An immediate consequence is that if a function's domain is a small closed type, then its modality does not matter. However, the type system does distinguish between the corresponding function types and has no way of coercing upstream against the order on the modality monoid. This shortcoming is addressed in Nuyts and Devriese [ND18] (Section 2.4.3) by having a separate mode for types that have no bridge relation, thus conflating the different modalities.

**Remark 2.4.6.** If we add $\mathbf{shi}$, $\mathbf{irr}$ and $\mathbf{shi} \vee \mathbf{par}$ (Remark 2.4.1), then the inequality relation is given by

$$\mathbf{ptw} < \mathbf{con} < \begin{array}{c}\mathbf{par}\\ \mathbf{shi}\end{array} < (\mathbf{shi} \vee \mathbf{par}) < \mathbf{irr}, \tag{2.8}$$

and **shi** and **par** are incomparable. Composition is given by

| $\downarrow \circ \rightarrow$ | ptw | con | par | shi | shi $\vee$ par | irr |
|---|---|---|---|---|---|---|
| ptw | ptw | ptw | par | ptw | par | irr |
| con | ptw | con | par | shi | shi $\vee$ par | irr |
| par | ptw | par | par | irr | irr | irr |
| shi | shi | shi | shi $\vee$ par | shi | shi $\vee$ par | irr |
| shi $\vee$ par | shi | shi $\vee$ par | shi $\vee$ par | irr | irr | irr |
| irr | irr | irr | irr | irr | irr | irr |

Since $\mathbf{ptw} \circ \mathbf{shi} = \mathbf{ptw} < \mathbf{con}$ and $\mathbf{con} < \mathbf{shi} = \mathbf{shi} \circ \mathbf{ptw}$, we see that $\mathbf{ptw} \dashv \mathbf{shi}$. Furthermore, $\mathbf{shi} \vee \mathbf{par} = \mathbf{shi} \circ \mathbf{par}$ and $\mathbf{irr} = \mathbf{par} \circ \mathbf{shi}$. These observations inspire us to extend the semantics from Theorem 2.4.3 with:

$$J(\mathbf{shi}) = \sharp, \qquad J(\mathbf{shi} \vee \mathbf{par}) = \sharp \circ \textstyle\int, \qquad J(\mathbf{irr}) = \textstyle\int \circ \sharp.$$

Together, these are all 6 'relation shifting' modalities whose modal functions still preserve path-equality. If we want to also classify functions that do not preserve path-equality, then we get 4 more modalities, but their locks cannot be interpreted as inverse images, so we would have to rely on Remark 2.2.3 and Theorem 1.6.26 to build a model. ◁

*Extending the MTT instance to ParamDTT🔒*

While ParamMTT is not ParamDTT, we can extend it soundly and come pretty close. The main remaining differences will be:

- The use of locks,
- That face restrictions on the context will have a modality annotation, which we consider an improvement over ParamDTT proper.

**Theorem 2.4.7.** *We can soundly extend ParamMTT to a system ParamDTT🔒 by adding:*

1. *Bridge interval variables, face propositions, Glue- and Weld-types [NVD17],*
2. *A judgement form for discrete types $\Gamma \vdash T\ \mathsf{dtype}_\ell \mathrel{@} *$ which is closed under discreteness-preserving type formers with modality annotations as in MTT and such that*

$$\frac{\Gamma \vdash T\ \mathsf{dtype}_\ell \mathrel{@} *}{\Gamma \vdash T\ \mathsf{type}_\ell \mathrel{@} *} \tag{2.9}$$

3. *The degeneracy axiom, stating that homogeneous paths in discrete types are constant,*[11]
4. *Parametric existential quantifiers,*
5. *A universe $\vdash \mathsf{U}^{\mathsf{DD}}\ \mathsf{dtype}_\ell \mathrel{@} *$ which is closed under discreteness-preserving type formers with modality annotations as in ParamDTT, which features a parametric decoding rule*

$$\frac{\Gamma, 🔒_{\mathbf{par}} \vdash T : \mathsf{U}^{\mathsf{DD}} \mathrel{@} *}{\Gamma \vdash \mathsf{El}(T)\ \mathsf{dtype}_\ell \mathrel{@} *}. \tag{2.10}$$

[11] This is the internalization of the identity extension lemma.

Note that discreteness-preserving type formers most notably exclude the Hofmann-Streicher universe and $\langle \mathbf{par} \mid - \rangle$,[12] which is why parametric existentials are explicitly listed as a separate addition.

*Proof.* 1. The bridge interval is simply interpreted by $\mathbf{y}(\mathbf{b})$. Glue and Weld exist in any presheaf category. We refer to the original work [NVD17; Nuy18a] for details.

2. The semantics of this judgement is simply a type which satisfies the degeneracy axiom.

3. This is then trivial.

4. Because discreteness is a robust notion of fibrancy [Nuy18b; Nuy18a], the discrete replacement commutes with substitution. Thus, we can simply take the $\Sigma$-type over $\langle \mathbf{par} \mid A \rangle$ and then take the discrete replacement of that.

5. Using standard techniques, we obtain a closed type $\mathsf{U}^{\mathsf{NDD}}$ that is a classifier for the discrete typing judgement but is itself not discrete and still has a continuous decoding rule [Nuy18a]. It's symbol stands for 'non-discrete universe of discrete types'. Then we define $\mathsf{U}^{\mathsf{DD}} \triangleq (\sharp\smallint)^* \mathsf{U}^{\mathsf{NDD}}$, as motivated below. □

**Remark 2.4.8** (Construction of $\mathsf{U}^{\mathsf{DD}}$)**.** The non-discrete universe $\mathsf{U}^{\mathsf{NDD}}$ behaves like the Hofmann-Streicher universe, only it classifies discrete types.

A bridge $B : X \frown_{\mathsf{U}^{\mathsf{NDD}}} Y$ then encodes a notion of heterogeneous bridges $(x : X) \frown_B (y : Y)$, and path $P : X \asymp_{\mathsf{U}^{\mathsf{NDD}}} Y$ encodes notions of paths $(x : X) \asymp_P (y : Y)$ *and* also a notion of bridges $(x : X) \frown_{P\mathbf{u}} (y : Y)$. The latter is inevitable, as $P$ must contain all information needed to form the bridge $P\mathbf{u} : X \frown_{\mathsf{U}^{\mathsf{NDD}}} Y$. It is immediately clear that the existence of a $P$ does not assert that $X = Y$, i.e. $\mathsf{U}^{\mathsf{NDD}}$ is not itself discrete.

In the desired discrete universe of discrete types $\mathsf{U}^{\mathsf{DD}}$, all paths are reflexive, i.e. we would like to have $\mathsf{U}^{\mathsf{DD}}\mathbf{p} = \mathsf{U}^{\mathsf{NDD}}()$.

Meanwhile, we want to model the parametric decoding rule by making sure that there exists a function $\mathsf{El}(-) : \langle \mathbf{par} \mid \mathsf{U}^{\mathsf{DD}} \rangle \to \mathsf{U}^{\mathsf{NDD}}$. This means that a bridge in $\mathsf{U}^{\mathsf{DD}}$ must be (at least) a path in $\mathsf{U}^{\mathsf{NDD}}$. We can achieve this if $\mathsf{U}^{\mathsf{DD}}\mathbf{b} = \mathsf{U}^{\mathsf{NDD}}\mathbf{p}$:

$$\begin{tikzcd}
X \asymp_{\mathsf{U}^{\mathsf{DD}}} Y \arrow[r, equal] \arrow[d, "(-)\mathbf{u}"'] & X =_{\mathsf{U}^{\mathsf{NDD}}} Y \arrow[d, "(-)\mathbf{r}"'] & X \asymp_{\mathsf{U}^{\mathsf{NDD}}} Y \arrow[d, "(-)\mathbf{u}"] \\
X \frown_{\mathsf{U}^{\mathsf{DD}}} Y \arrow[r, equal] & X \asymp_{\mathsf{U}^{\mathsf{NDD}}} Y \arrow[ur, "\mathsf{El}_{\frown}"] & X \frown_{\mathsf{U}^{\mathsf{NDD}}} Y
\end{tikzcd}$$

$$\boxed{\mathbf{par}}$$

Both equations are satisfied by taking $\mathsf{U}^{\mathsf{DD}} = (\sharp\smallint)^* \mathsf{U}^{\mathsf{NDD}}$, since

$$\sharp\smallint\mathbf{p} = \sharp() = (), \qquad \sharp\smallint\mathbf{b} = \sharp\mathbf{b} = \mathbf{p}, \tag{2.11}$$

(and $\sharp\smallint$ is the unique cartesian monoidal functor respecting these equations). ◁

The attentive reader might wonder why we found it appropriate to discard the bridge relation $\frown_{\mathsf{U}^{\mathsf{NDD}}}$ when building $\mathsf{U}^{\mathsf{DD}}$. The unsatisfactory answer is that the path relation $\asymp_{\mathsf{U}^{\mathsf{NDD}}}$ contains more information and that we ran out of slots so we had to discard something. An issue that can be traced back to this discarding, is that the internal parametricity operators of ParamDTT have an extremely contagious pointwise dependency that essentially renders proofs of parametricity theorems non-parametric themselves, getting in the way of iterated parametricity despite having a cubical model.

[12] as well as $\langle \mathbf{shi} \mid - \rangle$, $\langle \mathbf{shi} \vee \mathbf{par} \mid - \rangle$ and $\langle \mathbf{irr} \mid - \rangle$.

*Wrapping up*

In ParamDTT, ParamMTT and ParamDTT🔒, we see two important causes of discomfort: we have too many relation slots in small types (which feature an unnecessary bridge relation), and we have one too few in the universe. In Degrees of Relatedness (Section 2.4.3), small types are equipped with just a single relation, and every universe has one relation slot more than the types that it classifies.

### 2.4.3 Degrees of Relatedness

In comparison to ParamDTT [NVD17], the type system RelDTT [ND18] makes two improvements:

- It officially contains modalities that interact with the trivially and uniquely provable relation $\top$ (which were already available in `agda-parametric` but not in ParamDTT or its model),
- It addresses the aforementioned shortcomings of ParamDTT by moving to a multimode system in which types come equipped with a different number of relations, depending on their mode.

We focus on the second improvement. The modes of RelDTT (called **depths**) are integers starting from $-1$ and types of mode $m$ are equipped with $m+1$ relations called $\frown_0$ through $\frown_m$. The mode $m$ segment of the type system is modelled in 'depth $m$ cubical sets', which are presheaves over $\mathbf{Cube}_m$, the free cartesian monoidal category (with same terminal object) over $\mathbf{RG}_m$, which is generated by:

$$\begin{tikzcd}
\mathbf{n} \arrow[rrrr, bend left, "\mathbf{s}"] \arrow[rrrr, bend right, "\mathbf{t}"'] & \mathbf{e}_0 \arrow[l, "\mathbf{r}"'] & \mathbf{e}_1 \arrow[l, "\mathbf{u}_0^1"'] & \cdots \arrow[l, "\mathbf{u}_1^2"'] & \mathbf{e}_m \arrow[l, "\mathbf{u}_{m-1}^m"']
\end{tikzcd}$$

$$\begin{aligned}
\mathbf{r} \circ \mathbf{u}_0^1 \circ \ldots \circ \mathbf{u}_{m-1}^m \circ \mathbf{s} &= 1_{\mathbf{n}}, \\
\mathbf{r} \circ \mathbf{u}_0^1 \circ \ldots \circ \mathbf{u}_{m-1}^m \circ \mathbf{t} &= 1_{\mathbf{n}}.
\end{aligned}$$

By convention, $\mathbf{Cube}_{-1}$ is the point category.

The modalities $\mu : \mathrm{Hom}_{\mathcal{M}}(m, n)$ will be, essentially, all diagrams from $m$ ordered relations (and $\top$) to $n$ ordered relations (and $\top$) such that a 0-edge in the domain always gives rise to a 0-edge in the codomain, and such that we can also map from $\top$ to $\top$.

A succinct way to denote such a diagram is by answering, for all $i = 0 \ldots n$, the question: how related do the arguments need to be, if I want the results to be $i$-related? This gives rise to an increasing function $\{0 < 1 < \ldots < n\} \to \{0 < 1 < \ldots < m < \top\}$. Hence, we define:

**Definition 2.4.9.** The mode theory for RelDTT is the poset-enriched category

- whose objects are integers starting from $-1$,
- such that $\mathrm{Hom}(m, n)$ is the set of increasing functions
$$\mu : \{0 < 1 < \ldots < n\} \to \{0 < 1 < \ldots < m < \top\} : i \mapsto i \cdot \mu,$$
also denoted $\langle 0 \cdot \mu, \ldots, n \cdot \mu \rangle$,
- where the identity modality $\mathbf{con}$ is given by $i \cdot \mathbf{con} = i$ and composition is given by
$$i \cdot (\nu \circ \mu) = \begin{cases} (i \cdot \nu) \cdot \mu & \text{if } i \cdot \nu \neq \top, \\ \top & \text{if } i \cdot \nu = \top, \end{cases}$$

where $\mu \leq \nu$ whenever $i \cdot \mu \leq_1 \cdot \nu$ for all $i$.

**Example 2.4.10.** We refer to Nuyts and Devriese [ND18] for a compendium of interesting modalities. Here, we just mention parametricity $\mathbf{par} : \mathrm{Hom}_{\mathcal{M}}(m+1, m)$ for which $i \cdot \mathbf{par} = i + 1$ and its right adjoint structurality $\mathbf{str} : \mathrm{Hom}_{\mathcal{M}}(m, m+1)$ for which $0 \cdot \mathbf{str} = 0$ and $(i+1) \cdot \mathbf{str} = i$. They have the following form:

$$\begin{tikzcd}
x \frown_0 y \arrow[d] & f(x) \frown_0 f(y) \arrow[d] & x \frown_0 y \arrow[r] \arrow[d] & f(x) \frown_0 f(y) \arrow[d] \\
x \frown_1 y \arrow[d] \arrow[ur] & f(x) \frown_1 f(y) \arrow[d] & x \frown_1 y \arrow[d] \arrow[dr] & f(x) \frown_1 f(y) \arrow[d] \\
x \frown_2 y \arrow[d] \arrow[ur] & \cdots \arrow[d] & \cdots \arrow[d] & f(x) \frown_2 f(y) \arrow[d] \\
\cdots \arrow[d] & f(x) \frown_m f(y) \arrow[d] & x \frown_m y \arrow[d] \arrow[dr] & \cdots \arrow[d] \\
x \frown_{m+1} y \arrow[d] \arrow[ur] & \top & \top \arrow[dr] & f(x) \frown_{m+1} f(y) \arrow[d] \\
\top \arrow[ur] & & & \top
\end{tikzcd}$$

$\lhd$

**Theorem 2.4.11.** *The instantiation of MTT with the mode theory for RelDTT yields a type system RelMTT which can be modelled in the categories $\mathbf{PSh}(\mathbf{Cube}_m)$ as an instance of Section 2.2. RelMTT is* not *the system RelDTT [ND18].*

Remark 2.4.4 applies also for RelMTT vs. RelDTT.

**Lemma 2.4.12.** *The modalities $\mu : \mathrm{Hom}(m, n)$ are, by Galois connection ($\kappa \dashv \mu$), in 1-1 correspondence with increasing functions*

$$\kappa : \{0 < 1 < \ldots < m\} \to \{(=) < 0 < 1 < \ldots < n\} : j \mapsto j \cdot \kappa,$$

*which are called **contramodalities**.* $\square$

*Proof of Theorem 2.4.11.* We define the 2-functor $J : \mathcal{M} \to \mathbf{Cat}$ that sends modes to base categories. Of course, we need $J(m) = \mathbf{Cube}_m$. In order to define $J(\mu)$, let $\kappa \dashv \mu$ be the corresponding contramodality. Since $J(\mu)_*$ is going to be the interpretation of the DRA of $\mu$, we can think of $J(\mu)^*$ as the interpretation of $\kappa$. Hence, we define $J(\mu)$ to be the cartesian monoidal functor such that $J(\mu)\mathbf{e}_i = \mathbf{e}_{i \cdot \kappa}$ if $i \cdot \kappa \neq (=)$, and to the terminal object otherwise. $\square$

**Proposition 2.4.13.** *ParamMTT is a subsystem of RelMTT with the same semantics. Concretely, we have a functor* $I : \mathcal{M}_{\mathsf{ParamDTT}} \to \mathcal{M}_{\mathsf{RelDTT}}$*, invertible on Hom-posets, from the mode theory of ParamDTT to the mode theory of RelDTT such that the following diagram commutes if we identify* $\mathbf{BPCube} = \mathbf{Cube}_1$*:*

$$\begin{tikzcd} \mathcal{M}_{\mathsf{ParamDTT}} \arrow[r, "I"] \arrow[dr, "J"'] & \mathcal{M}_{\mathsf{RelDTT}} \arrow[d, "J"] \\ & \mathbf{Cat} \end{tikzcd}$$

*This still works if we include the modalities from Remark 2.4.6.*

*Proof.* The following definition of $I$ does the job:

$$I(*) = 1, \qquad \begin{array}{ccc} I(\mathbf{ptw}) = \langle 0, 0\rangle\,, & I(\mathbf{con}) = \langle 0, 1\rangle\,, & I(\mathbf{par}) = \langle 1, 1\rangle\,, \\ I(\mathbf{shi}) = \langle 0, \top\rangle\,, & I(\mathbf{shi} \vee \mathbf{par}) = \langle 1, \top\rangle\,, & I(\mathbf{irr}) = \langle \top, \top\rangle\,. \end{array} \qquad \square$$

Again, we can extend RelMTT to something that differs from RelDTT mainly in the use of locks vs. left division:

**Theorem 2.4.14.** *We can soundly extend RelMTT to a system* $\mathsf{RelDTT}_{🔒}$ *by adding:*

1. *Interval variables, face propositions, Glue- and Weld-types,*
2. *A judgement form for discrete types* $\Gamma \vdash T\ \mathsf{dtype}_\ell\ @\ m$ *which is closed under discreteness-preserving type formers with modality annotations as in MTT and such that*

$$\frac{\Gamma \vdash T\ \mathsf{dtype}_\ell\ @\ m}{\Gamma \vdash T\ \mathsf{type}_\ell\ @\ m} \tag{2.12}$$

3. *The degeneracy axiom, stating that homogeneous 0-edges in discrete types are constant,*[13] *or at mode* $-1$ *that elements of the same type are equal,*
4. *Modal existential quantifiers for modalities* $\mu$ *such that* $0 \cdot \mu \neq 0$*,*
5. *A universe* $\vdash \mathsf{U}^{\mathsf{DD}}\ \mathsf{dtype}_\ell\ @\ m+1$ *which is closed under discreteness-preserving type formers with modality annotations as in RelDTT, which features a parametric decoding rule*

$$\frac{\Gamma, 🔒_{\mathbf{par}} \vdash T : \mathsf{U}^{\mathsf{DD}}\ @\ m+1}{\Gamma \vdash \mathsf{El}(T)\ \mathsf{dtype}_\ell\ @\ m}. \tag{2.13}$$

Note that discreteness-preserving type-formers most notably exclude the Hofmann-Streicher universe and $\langle \mu \mid - \rangle$ when $0 \cdot \mu \neq 0$, which is why existentials for those modalities are explicitly listed as a separate addition.

*Proof.* All points but the last are proved as in Theorem 2.4.7.

Using standard techniques, we obtain a non-discrete universe of discrete types $\mathsf{U}^{\mathsf{NDD}}$ at every mode $m$. Then we define $\mathsf{U}^{\mathsf{DD}} \triangleq J(\mathbf{par})^* \mathsf{U}^{\mathsf{NDD}}$ (where $J$ is defined as in the proof of Theorem 2.4.3), which lives at mode $m+1$. Note that $J(\mathbf{par})$ sends $\mathbf{e}_0$ to the terminal object and $\mathbf{e}_{i+1}$ to $\mathbf{e}_i$. Hence, the 0-edges of $\mathsf{U}^{\mathsf{DD}}$ are the points of $\mathsf{U}^{\mathsf{NDD}}$ (so it is discrete) and we shove aside all other relations so that the $(i+1)$-edges of $\mathsf{U}^{\mathsf{DD}}$ are the $i$-edges of $\mathsf{U}^{\mathsf{NDD}}$. This indexation shift allows for a parametric function $\langle \mathbf{par} \mid \mathsf{U}^{\mathsf{DD}} \rangle \to \mathsf{U}^{\mathsf{NDD}}$. $\square$

In this system, we can think of terms at mode $-1$ as proofs, at mode $0$ as programs, at mode $1$ as types, at mode $2$ as kinds, etc.

[13] This is the internalization of the identity extension lemma.

### *2.4.4 MTT as an internal language of the model*

As mentioned, neither ParamDTT nor RelDTT are themselves instances of MTT, their most stark deviation being the parametric type decoding rule which causes both systems to enforce different modalities for terms and their types (e.g. parametric functions have continuous types and irrelevant functions have shape-irrelevant types).

Fleshing out the semantics of both systems was a major effort and produced a technical report [Nuy18a], some parts of which could be classified as 'write-only'. It would have been desirable to carry out these proofs in a proof-assistant, i.e. internal to another type system. For the authors, this has the advantage that a lot of tedious bookkeeping could be done automatically, and RelDTT's end users would of course have more confidence in the system.

As both models start with an instantiation of Section 2.2, MTT seems quite well-suited as a metatheory in which discreteness, $\mathsf{U}^{\mathsf{NDD}}$ and $\mathsf{U}^{\mathsf{DD}}$ could be defined, and ParamDTT and RelDTT could be shallowly embedded. This is in fact one of the central motivations behind the Menkar project [Nuy19].

There is one important difficulty, namely that the creation of $\mathsf{U}^{\mathsf{DD}}$ out of $\mathsf{U}^{\mathsf{NDD}}$ needs to insert an equality relation in relation slot 0, which the internal modalities of ParamDTT and RelDTT are unable to do. A contramodality $\kappa \dashv \mu$ has the capacity to do so, however. If in the above construction, we set $[\![\text{🔒}_\kappa]\!] = J(\mu)_!$, then an $i$-edge in $\langle \kappa \mid A \rangle$ is an $(i \cdot \kappa)$-edge in $A$, or an equality proof if $i \cdot \kappa = (=)$.

On the other hand, modalities such as irrelevance and shape-irrelevance interact with $\top$ and as such are not contramodalities. So an ideal metatheory in which to embed RelDTT, has as its dependent right adjoints both inverse and direct images, which may compose to DRAs that are neither. As such, it becomes difficult to provide semantics that are strictly functorial on locks, and we need to invoke Remark 2.2.3 and Theorem 1.6.26 to model an appropriate metatheory.

**Definition 2.4.15.** The mode theory for the model of RelDTT is the poset-enriched category

- whose objects are integers starting from $-1$,
- such that $\mathrm{Hom}(m, n)$ is the set of increasing functions

$$\mu : \{0 < 1 < \ldots < n\} \to \{(=) < 0 < 1 < \ldots < m < \top\} : i \mapsto i \cdot \mu,$$

- where the identity modality **con** is given by $i \cdot \mathbf{con} = i$ and composition is given by

$$i \cdot (\nu \circ \mu) = \begin{cases} (i \cdot \nu) \cdot \mu & \text{if } i \cdot \nu \notin \{=, \top\}, \\ (=) & \text{if } i \cdot \nu = (=), \\ \top & \text{if } i \cdot \nu = \top, \end{cases}$$

where $\mu \leq \nu$ whenever $i \cdot \mu \leq i \cdot \nu$ for all $i$.

**Theorem 2.4.16.** *Using Theorem 1.6.26, there is a model in categories equivalent to $\mathbf{Cube}_m$ for MTT over the mode theory for the model of RelDTT.*

*Proof.* Every modality of this mode theory can be written as a composite of a modality $\mu$ of RelDTT and a contramodality $\kappa$ of RelDTT. We can interpret $[\![\text{🔒}_\mu]\!] = J(\mu)^*$ and $[\![\text{🔒}_\kappa]\!] = J(\nu)_!$ where $\kappa \dashv \nu$. One can show that this constitutes a pseudofunctor.[14] As inverse and direct images are always DRAs, we can invoke Theorem 1.6.26. □

[14]E.g. by noting that the category of presheaves over $\mathbf{Cube}_m$ is equivalent to the category of 0-discrete sheaves over $\mathbf{Cube}_{m+1}$ and interpreting the modalities strictly functorially in the sheaf categories.

## 2.5 *Idempotent S4*

One of the most studied modal logics is **S4** [PD01; Shu18; GSB19; Zwa19]. This system includes a single comonad, traditionally written $\square$. Here, we consider an instantiation of MTT which models an idempotent version of **S4**. The mode theory, $\mathcal{M}$, consists of a single mode $m$, and a single idempotent endomorphism $\mu$:

$$\begin{tikzcd} m \arrow[loop above, "\mu"] \end{tikzcd}$$

We have required $\mu \circ \mu = \mu$, so $\mathrm{Hom}_{\mathcal{M}}(m, m) = \{1, \mu\}$. We further specify a single inequality between modalities: $\mu \leq 1$. This mode theory is merely poset enriched, but if we wished to model a non-strictly idempotent comonad we would need to use a non-posetal 2-category.

**Notation 2.5.1.** We will write $\square A$ for $\langle \mu \mid A \rangle$ in keeping with more traditional calculi for **S4**.

**Theorem 2.5.2.** *In MTT with $\mathcal{M}$, $\square A$ is an idempotent comonad internally to the type theory.*

*Proof.* In order to show this, we must exhibit a function $\square A \to A$ and $\square A \to \square\square A$ which satisfy the comonad equations. Both of these functions can be taken wholesale from Section 1.3.1. The first is $\mathbf{coe}[\mu \leq 1](-)$, and the second is $\mathbf{triv}^{-1}(\mathbf{comp}_{\mu,\mu}(-))$. The equations hold up to internal equality and follow from a straightforward calculations. $\square$

We can do better than merely showing that $\square$ behaves like a comonad, this instantiation of MTT has a very similar flavor to a dependent version of Pfenning and Davies [PD01] (such as Shulman [Shu18]). In particular, because there are precisely two modalities in the system, there are two variable rules:

$$\frac{\Gamma_0, x : (\mu \mid A), \Gamma_1 \ \mathsf{ctx} \mathbin{@} m}{\Gamma_0, x : (\mu \mid A), \Gamma_1 \vdash x : A \mathbin{@} m} \qquad \frac{\Gamma_0, x : (1 \mid A), \Gamma_1 \ \mathsf{ctx} \mathbin{@} m \qquad \mathsf{locks}(\Gamma_1) = 1}{\Gamma_0, x : (1 \mid A), \Gamma_1 \vdash x : A \mathbin{@} m}$$

One should contrast this with the variable rule for accessing a valid variable and the rule for accessing an ephemeral variable. One major difference between our system and a dual-context approach is that our style of context management is based on locks, rather than a pair of static zones. This allows for valid types to depend on ephemeral variables, in a limited way of course.

## 2.6 Dependent Right Adjoints

A closely related modal type theory is the calculus of *dependent right adjoint*, developed in Birkedal et al. [Bir+20]. We have already discussed some of the relation between dependent right adjoints and MTT's notion of modalities (e.g. Section 1.4.2). In this section, we attempt to compare the expressivity of both systems.

Birkedal et al. [Bir+20] has a single modality (written $\square$) which encodes the rules of a dependent right adjoint. In order to represent this syntax for MTT, we instantiate $\mathcal{M}$ to a category with a single mode, $m$, and one generator for $\mathrm{Hom}_{\mathcal{M}}(m, m)$:

$$\begin{tikzcd} m \arrow[loop above, "\mu"] \end{tikzcd}$$

We impose no further equations on this category (so in particular, $\mu \circ \mu \neq \mu$).

**Theorem 2.6.1.** *Any model of Birkedal et al. [Bir+20] is a model of this instantiation of MTT*

*Proof.* This is an immediate corollary of Theorem 1.4.11. ☐

This result tells us our syntax is certainly sound with respect to the calculus of dependent right adjoints. At a more intuitive level, we can encode the contexts from MTT as contexts in the calculus of dependent right adjoints as follows:

$$\begin{aligned} [\![\cdot]\!] &= \cdot \\ [\![\Gamma.(\mu^n \mid A)]\!] &= [\![\Gamma]\!].\square^n A \\ [\![\Gamma.\text{🔒}_{\mu^n}]\!] &= [\![\Gamma]\!].\text{🔓}^n \end{aligned}$$

MTT is certainly not complete for the calculus of dependent right adjoints. The central issue is precisely the mismatch described in Theorem 1.4.11: our calculus does not require that the same strong elimination rule as Birkedal et al. [Bir+20]. Moreover, we cannot encode the open-scope eliminator for $\square$, **open**, in MTT.

To what extent does this matter? It is not evident that the loss of this stronger elimination rule is as significant as it may appear. For instance, we are certainly still capable of proving the dependent axiom K (function application under $\square$).

Moreover, while it is difficult to prove without a normalization result, it is reasonable to conjecture that MTT is complete for *closed* terms. That is, given any closed term in the DRA calculus, there is a (non-compositional!) translation of it to MTT. Such a result would allow us to definitively prove that it is sufficient to work with MTT, even though it may be less convenient in some circumstances. With the addition of appropriate commuting conversions for $\mathsf{let}_\nu\ \mathsf{mod}_\mu(\_) \leftarrow M_0\ \mathsf{in}\ M_1$, this result may even to extend to open terms.

This mismatch is clearly related to the distinction between the Fitch-style calculi and the dual-context calculi for $\square$ [PD01; Kav17; Shu18]. In order to further crystallize this divide, let us suppose that we have a substition inverse to $\uparrow.\mathsf{mod}_\mu(\mathbf{v}_0)$:

$$\begin{tikzcd} \Gamma.(\mu \mid A) \arrow[rr, bend left, "\uparrow.\mathsf{mod}_\mu(\mathbf{v}_0)"] && \Gamma.(1 \mid \langle \mu \mid A \rangle) \arrow[ll, bend left, "\sigma"] \end{tikzcd}$$

This inverse substitution can be defined with the stronger, open-scope version of $\mathbf{open}$: $\sigma = \uparrow.\mathbf{open}(\mathbf{v}_0)$ However, the existence of $\sigma$ is also *sufficient* to define the stronger elimination rule:

$$\frac{\Gamma \vdash M : \Box A}{\Gamma.\text{🔒} \vdash \mathsf{open}(M) : A} \triangleq$$

$$\dfrac{\dfrac{}{\Gamma.(\mu \mid A).\text{🔒}_\mu \vdash \mathbf{v}_0 : A \mathbin{\color{gray}@\, m}} \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash \mathsf{id} : \Gamma \mathbin{\color{gray}@\, m}} \qquad \Gamma \vdash M : \langle \mu \mid A \rangle \mathbin{\color{gray}@\, m}}{\Gamma \vdash \mathsf{id}.M : \Gamma.(1 \mid \langle \mu \mid A \rangle) \mathbin{\color{gray}@\, m}} \qquad \dfrac{}{\Gamma.(1 \mid \langle \mu \mid A \rangle) \vdash \sigma : \Gamma.(\mu \mid A) \mathbin{\color{gray}@\, m}}}{\Gamma \vdash \sigma \circ (\mathsf{id}.M) : \Gamma.(\mu \mid A) \mathbin{\color{gray}@\, m}}}{\Gamma.\text{🔒}_\mu \vdash (\sigma \circ (\mathsf{id}.M)).\text{🔒}_\mu : \Gamma.(\mu \mid A).\text{🔒}_\mu \mathbin{\color{gray}@\, m}}}{\Gamma.\text{🔒}_\mu \vdash \mathbf{v}_0[(\sigma \circ (\mathsf{id}.M)).\text{🔒}_\mu] : A \mathbin{\color{gray}@\, m}}$$

With the introduction of this primitive substition $\sigma$, we can no longer trivially resolve all explicit substitutions by just pushing them in towards variables and this stronger version of $\mathbf{open}$ is an example of such a stuck term. Substitution is still likely admissible, but it would no longer be automatic and must be established quite carefully [GSB19].

Therefore, while MTT is certainly sound for the calculus of dependent right adjoints, it is not complete, and the failure of completeness is precisely the lack of an inverse to the substitution $\uparrow.\mathsf{mod}_\mu(\mathbf{v}_0)$. It is not clear whether this loss of power is truly problematic, and it is reasonable to conjecture that it is unimportant overall.

## 2.7 Warps

Nakano's [Nak00] later modality, written $\blacktriangleright$, marks a type as producing information one timestep later. By capturing this information in the types, Nakano's [Nak00] system is able to provide an abstract characterization of productive definitions; one that is immune to refactoring or restructuring the definitions.

This system has proven to be a tantalizing formulation of coinduction, but in order to capture coinductive types it is necessary to add more modalities. In particular, Bizjak et al. [Biz+16] showed how a combination of $\blacktriangleright$ and $\square$ could capture coinductive types, while $\blacktriangleright$ alone could not. There are technical complications from combining these two modalities, however, and the interactions between $\square$ and $\blacktriangleright$ have proven to be a major challenge for guarded type theories. In Section 2.3, we demonstrated that MTT could reconcile these two modalities and thus provide a type theory which can smoothly model coinduction.

Another line of research [Gua18] proposes a way to avoid these challenges by capturing $\square$ and $\blacktriangleright$ as instantiations of the same parametrized modality. Therefore, rather than dealing with the actions of two a priori unrelated modalities, Guatto [Gua18] can just provide a type theory with one modality; one which is sufficiently flexible to capture both $\square$ and $\blacktriangleright$ as instances. In particular, Guatto defines this übermodality, $*_p$, which is parameterized by a warp: an abstract description of the rate at which the computation produces information. To be precise, we will define a warp to be a monotone function from $\omega + 1 \to \omega + 1$ which preserves all joins and sends $\omega$ to $\omega$.

**Remark 2.7.1.** This is a deviation from Guatto [Gua18] which only required that only that $p$ preserves all joins, a weaker condition allowing $\alpha \mapsto 0$ as a valid warp. We have chosen to restrict warps in this way because it allows us to work with a semantics in $\mathbf{PSh}(\omega)$ without undue effort, and it still includes both $\square$ and $\blacktriangleright$. $\lhd$

A program of type $\langle *_p \mid A\rangle$ then describes a computation which at stage $n$ has produced the information required by $A$ at stage $p(n)$. For instance, $\blacktriangleright$ would be modeled as $*_{n \mapsto n-1}$:[15] at stage $n$ this type has only produced the information for step $n-1$. On the other hand, $\square$ can be defined as $*_{n \mapsto \omega}$, because at stage $n$ the information for stage $\omega$ is already available. Monotonicity ensures that a type cannot suddenly lose information that it had previously produced, and requiring that $\omega$ be sent to $\omega$ ensures that globally a warp does not cause a computation to lose information.

It was challenging to explain how $\blacktriangleright$ and $\square$ should interact when they were separate modalities, however, there is a simple method for combining them when viewed as particular warps: $*_p \circ *_q \cong *_{q \circ p}$. Simple computation assures us that our encodings of $\square$ and $\blacktriangleright$ combine as expected, e.g. $\square \circ \blacktriangleright \cong \square$. Even though the generality of $*_p$ can be motivated by just two instances ($\square$ and $\blacktriangleright$), there are many, many warps beyond just these two. This extra flexibility turns out to be useful in capturing more complex guarded programs, in which information is produced at various rates. We refer the reader to Guatto [Gua18] for further details and examples.

Despite the advantages provided by the warp modalities, the calculus is still complex and Guatto's [Gua18] system is unsuitable for generalizing to a dependent type theory. The central issue is, as always, the management of the modal context: the proposed warp calculus does not satisfy a general substitution principle. With the machinery of MTT, however, there is a simple way to recover this calculus.

For a mode theory, we consider the poset-enriched category with a single object $m$ and a 1-cell, $\bar{p}$ for each cocontinuous function $p : \omega+1 \to \omega+1$. We order these 1-cells such that $\bar{p} \geq \bar{q}$ if $p(\alpha) \leq q(\alpha)$ for every $\alpha$.

The induced calculus is equipped with a modality for each warp, and Section 1.3 ensures that the subtyping rules of Guatto [Gua18] become natural transformations in MTT. There is no term corre-

[15] Like Guatto, we only give a warp's action on finite non-zero ordinals; its action on 0 and $\omega$ is forced.

sponding to Löb induction, nor can there be: MTT does not ensure the existence of any "modal-specific" operations and only provides the operations enforced by the mode theory. One can add Löb induction as an axiom, but showing this is sound requires constructing a model of MTT which satisfies Löb induction.

We now turn to constructing a model of MTT in $\mathbf{PSh}(\omega)$, with $*_{-1}$ and $*_{\omega}$ being sent to the familiar $\blacktriangleright$ and $\square$ in this model [Bir+12], thereby showing the soundness of Löb induction. More generally, we shall arrange matters so that $[\![\text{🔒}_{*_p}]\!]$ is $\widehat{p}^*$, where $\widehat{p}$ is the restriction of the left Galois connection for a warp $p$. This Galois connection can be constructed as follows:

$$\widehat{p} : \omega + 1 \to \omega + 1$$
$$\widehat{p}(\alpha) = \bigwedge \{n \in \omega \mid \alpha \leq p(n)\}$$

Moreover, using the requirement that $p(\omega) = \omega$ and the fact that $p$ is monotone, we can conclude that the only situation when $p(\alpha) = \omega$ is $\alpha = \omega$. To see this, observe that if $\alpha = n < \omega$, then $p(\omega) = \omega > n$. We have required that $p(\omega) = \bigvee_m p(m)$, so there must exist some $m$ such that $p(m) \geq n$ because $n$ is compact. Therefore, $m \in \{n \mid n \leq p(n)\}$ and so $\widehat{p}(n) \leq m < \omega$.

Next, let us observe that $\widehat{p}$ is cocontinuous (as a left adjoint), and therefore fully determined by its restriction to $\omega$. We have concluded that $\widehat{p}(n) \in \omega$ for all $n \in \omega$, and so $\widehat{p}$ is fully determined as a map $\omega \to \omega$.

**Lemma 2.7.2.** *There is a 2-equivalence between $\mathcal{M}^{\mathsf{coop}}$ and $\mathcal{M}'$. Here $\mathcal{M}'$ is the poset-enriched category with one object, an endomorphism for each monotone function $\omega \to \omega$ which preserves zero, and with $f \leq g$ when $f(n) \geq g(n)$ for all $n$.*

*Proof.* This proof is a standard application of some adjoint calculus. First, we observe that since adjoints (Galois connections) are unique up to isomorphism, the procedure sending $p : \mathrm{Hom}_{\mathcal{M}}(m, m)$ to $\widehat{p}$ is injective. Moreover, it is bijective because every monotone and 0-preserving function $q : \omega \to \omega$ extends uniquely to a cocontinuous function $q^+ : \omega + 1 \to \omega + 1$, moreover, transposing this function gives $p : \omega + 1 \to \omega + 1$ such that $\widehat{p} = q$. Here, $p$ is determined by the following formula:

$$p(\alpha) = \bigvee \{n \in \omega \mid q(n) \leq \alpha\}$$

In this case, we must have that $p(\omega) = \bigvee \{n \in \omega \mid q(n) \leq \omega\}$, and since $q(n) < \omega$ by definition, we must have that $p(\omega) \geq n$, for all $n$. Therefore, $p(\omega) = \omega$ and so $p$ constitutes a valid 1-cell in $\mathcal{M}$.

This shows that this functor is full and faithful, so it only remains to show that it respects the ordering of 1-cells. To show this, it suffices to recall the standard fact that if $p \leq p'$, then $\widehat{p} \geq \widehat{p}'$. □

**Lemma 2.7.3.** *$\mathcal{M}'$ is the full subcategory of $\mathbf{Pos}^{\mathsf{coop}}$ when we restrict to precisely one object: $\omega$.*

*Proof.* This follows immediately by unfolding the definitions of the 1- and 2-cells in $\mathcal{M}'$. □

With this lemma in hand, we will construct our desired model:

**Theorem 2.7.4.** *There exists a model of MTT with $\mathcal{M}$ where the mode is interpreted as the standard model of type theory on $\mathbf{PSh}(\omega)$, and each modality $p$ is interpreted by a dependent right adjoint extending $\widehat{p}^* \dashv (\widehat{p})_*$. In particular, $-1$ is interpreted by the adjunction $\blacktriangleleft \dashv \blacktriangleright$ and $\_ \mapsto \omega$ is interpreted by $\blacklozenge \dashv \square$.*

*Proof.* We will use Theorem 1.4.11 to construct this model. First, we must define a 2-functor $L : \mathcal{M}^{\mathsf{coop}} \to \mathbf{Cat}$. We will define this functor by factoring through the functor $\mathbf{PSh}(-) : \mathbf{Pos}^{\mathsf{coop}} \to \mathbf{Cat}$. In particular, we define the functor $M^{\mathsf{coop}} \to \mathbf{Pos}^{\mathsf{coop}}$ by composing with the equivalence of Lemma 2.7.2 and the inclusion of Lemma 2.7.3.

This gives a 2-functor $\mathcal{M}^{\mathsf{coop}} \to \mathbf{Cat}$ which sends the unique object to $\mathbf{PSh}(\omega)$ and interprets each lock as an appropriate inverse image functor. We may then apply Lemma 2.1.5 to conclude that each lock is then part of an adjunction which induces a dependent right adjoint. Therefore, Theorem 1.4.11 gives an appropriate model. For the final part of the theorem, we can simply calculate the behavior of $-1$ and $\_ \mapsto \omega$ as they are fed through various equivalences to see that they indeed come out to the desired values.

For instance:

$$\widehat{-1}(n) = \bigwedge\{m \mid n \leq m - 1\} = n + 1$$

Therefore, $[\![🔒_{-1}]\!] = (n \mapsto n + 1)^* = ◀$, and therefore $\mathbf{Mod}_{-1} = ▶$ as expected. $\square$

## 2.8 *Internal Adjoints*

In this section we consider two modalities which are adjoint *to each other*. We require that all modalities are a weak form of dependent right adjoint, so they all must have a left adjoint on the context. In this section, however, we wish to internalize one of these left adjoints so that it can be applied to types. This is a fundamental example arising in many different settings [SS12; ND18; Shu18].

We define a mode theory freely generated by the following diagram

$$\begin{tikzcd} m \arrow[r, bend left, "\mu"] & n \arrow[l, bend left, "\nu"] \end{tikzcd}$$

and the following two-cells, subject to the given equalities:

$$\eta : 1 \Rightarrow \mu \circ \nu \qquad\qquad \epsilon : \nu \circ \mu \Rightarrow 1$$

$$1_\mu = (1_\mu \star \epsilon) \circ (\eta \star 1_\mu) \qquad\qquad 1_\nu = (\epsilon \star 1_\nu) \circ (1_\nu \star \eta)$$

This 2-category could be called the "walking adjunction"; a 2-functor out of it classifies an adjunction in the codomain. It is routine to calculate that this category is also "self-dual" as a two-category: $M^{\mathsf{coop}} \simeq M$ (of course, the equivalence swaps $m$ and $n$, $\mu$ and $\nu$, and $\eta$ and $\epsilon$). Therefore, a model of this instantiation of $\mathsf{MTT}$ must start from a pair of categories, representing the two sorts of contexts, and an adjunction between them. We wish to show that this relationship extends to the modal types themselves, so that $\langle \nu \mid - \rangle \dashv \langle \mu \mid - \rangle$. We will prove this by exhibiting the unit and counit of such an adjunction, and show that they satisfy the required properties.

$$\begin{aligned} u \;&:\; (x : A) \to \langle \mu \mid \langle \nu \mid A[\text{🔑}^{\eta}_{\cdot, x:(1|A)}] \rangle \rangle \\ u &\triangleq \lambda x.\, \mathsf{mod}_\mu(\mathsf{mod}_\nu(x^\eta)) \end{aligned}$$

$$\begin{aligned} e \;&:\; (x : \langle \nu \mid \langle \mu \mid A \rangle \rangle) \to A[\text{🔑}^{\epsilon}_{\cdot, x:(1|\langle \nu | \langle \mu | A \rangle \rangle)}] \\ e &\triangleq \lambda x.\, \mathsf{let}\ \mathsf{mod}_\nu(y_0) \leftarrow x\ \mathsf{in}\ \mathsf{let}_\nu\ \mathsf{mod}_\mu(y_1) \leftarrow y_0\ \mathsf{in}\ y_1^\epsilon \end{aligned}$$

**Theorem 2.8.1.** *The terms $u$ and $e$ satisfy the triangle equalities up to internal equality, e.g.:*

$$\begin{tikzcd} \nu \arrow[r, "\nu \star \eta"] \arrow[rr, bend right, "1_\nu"'] & \nu \circ \mu \circ \nu \arrow[r, "\epsilon \star \nu"] & \nu \end{tikzcd}$$

$$\begin{tikzcd} \mu \arrow[r, "\eta \star \mu"] \arrow[rr, bend right, "1_\mu"'] & \mu \circ \nu \circ \mu \arrow[r, "\mu \star \eta"] & \mu \end{tikzcd}$$

*Proof.* We construct the terms witnessing these internal equalities as follows:

$$\begin{aligned} \_ \;&:\; (x : \langle \nu \mid A \rangle) \to \mathsf{Id}_{\langle \nu | A \rangle}(x, e(\mathsf{mod}_\nu(u) \circledast_\nu x)) \\ \_ &\triangleq \lambda x.\, \mathsf{let}\ \mathsf{mod}_\nu(y) \leftarrow x\ \mathsf{in}\ \mathsf{refl}(\mathsf{mod}_\nu(y)) \end{aligned}$$

$$\begin{aligned} \_ \;&:\; (x : \langle \mu \mid A \rangle) \to \mathsf{Id}_{\langle \mu | A \rangle}(x, \mathsf{mod}_\mu(e) \circledast_\mu u(x)) \\ \_ &\triangleq \lambda x.\, \mathsf{let}\ \mathsf{mod}_\mu(y) \leftarrow x\ \mathsf{in}\ \mathsf{refl}(\mathsf{mod}_\mu(y)) \end{aligned}$$

Recall that $\circledast$ is the term for axiom K constructed in Section 1.3. The proof that this term typechecks is involved and relies on interchange law from Section 1.2. For the sake of explicitness, we present part of this proof for the first equality. When type-checking this proof, we must show that $\mathsf{refl}(\mathsf{mod}_\nu(y))$ has the type $\mathsf{Id}_{\langle \nu | A \rangle}(x, e(\mathsf{mod}_\nu(u) \circledast_\nu \mathsf{mod}_\mu(y)))$. Let us consider the rightmost term of this equality type:

$$
\begin{aligned}
e(\mathsf{mod}_\nu(u) \circledast_\nu \mathsf{mod}_\mu(y)) &= e(\mathsf{mod}_\nu(u(y))) \\
&= e(\mathsf{mod}_\nu(\mathsf{mod}_\mu(\mathsf{mod}_\nu(y^\eta)))) \\
&= \mathsf{mod}_\nu(y^\eta)^\epsilon \\
&= \mathsf{mod}_\nu(y[\text{🗝}^{\eta}_{\Gamma.\text{🔒}_\nu}])[\text{🗝}^{\epsilon}_{\Gamma}] \\
&= \mathsf{mod}_\nu(y[\text{🗝}^{\eta}_{\Gamma.\text{🔒}_\nu \circ \text{🗝}^{\epsilon}_{\Gamma}.\text{🔒}_\nu}]) \\
&= \mathsf{mod}_\nu(y[\text{🗝}^{\nu \star \eta}_{\Gamma} \circ \text{🗝}^{\epsilon \star \nu}_{\Gamma}]) && (*) \\
&= \mathsf{mod}_\nu(y[\text{🗝}^{\epsilon \star \nu \circ \nu \star \eta}_{\Gamma}]) \\
&= \mathsf{mod}_\nu(y[\text{🗝}^{1_\mu}_{\Gamma}]) \\
&= \mathsf{mod}_\nu(y[\mathsf{id}]) \\
&= \mathsf{mod}_\nu(y)
\end{aligned}
$$

The crucial move here is to observe that as part of a two-functor, $-.\text{🔒}_\nu$ and $\text{🗝}^{-}_{-}$ preserve whiskering, as was used in $(*)$. This preservation is ensured, in particular, by the interchange law demanded in Section 1.2. (The line above $(*)$ just swaps both substitutions visually due to 2-contravariance.) □

**Theorem 2.8.2.** *If $\mathcal{C}$ and $\mathcal{D}$ are two categories which can be equipped with models of type theory, and there is a pair of dependent right adjoints between them, $[\![\text{🔒}_\mu]\!]$, $[\![\text{🔒}_\nu]\!]$, where the left adjoints (the maps between categories of contexts) are adjoint, $[\![\text{🔒}_\nu]\!] \dashv [\![\text{🔒}_\mu]\!]$, then $\mathcal{C}$ and $\mathcal{D}$ model MTT with $\mathcal{M}$.*

*Proof.* A straightforward application of Theorem 1.4.11. □

**Theorem 2.8.3.** *Any model of $\mathcal{M}$ must interpret $[\![\text{🔒}_\mu]\!]$ and $[\![\text{🔒}_\nu]\!]$ as adjoint functors. Moreover, if $\mathbf{Mod}_\mu$ and $\mathbf{Mod}_\nu$ are induced by the adjunctions $[\![\text{🔒}_\mu]\!] \dashv R_\mu$ and $[\![\text{🔒}_\nu]\!] \dashv R_\nu$ lifted to a dependent right adjoints (Lemma 2.1.3), then $R_\nu \dashv R_\mu$.*

*Proof.* The first claim is a result of a the fact that adjoint functors are precisely adjoint morphisms in the 2-category, $\mathbf{Cat}$. Since adjoint morphisms are preserved by 2-functors, and $\nu$ and $\mu$ in $\mathcal{M}^{\mathsf{coop}}$ are internally adjoint, $[\![\text{🔒}_\nu]\!]$ and $[\![\text{🔒}_\mu]\!]$ are necessarily adjoint: $[\![\text{🔒}_\nu]\!] \dashv [\![\text{🔒}_\mu]\!]$.

Moreover, if $[\![\text{🔒}_\nu]\!] \dashv R_\nu$, where $R_\nu$ lifts to $\mathbf{Mod}_\nu$, by the uniqueness of adjoints we must have that $R_\nu \cong [\![\text{🔒}_\mu]\!]$. If we also have that $\mathbf{Mod}_\mu$ is induced by a functor $R_\mu \vdash [\![\text{🔒}_\mu]\!]$, we then have $R_\nu \dashv R_\mu$. □

### *2.8.1 When is Transposition Internally Definable?*

In Licata et al. [Lic+18], a crucial move in the construction of the univalent universe is a right adjoint to the path type. The addition of this adjoint is difficult, however, because the adjoint does not internalize in a pleasant way. In particular, Licata et al. [Lic+18] showed that *if* the transposition action of the adjunction is definable inside the type theory, then the interval is trivial. Thus far, we have avoided such issues in our treatment of adjoints and worked exclusively with the unit and counit. However, since the adjoint to $-^{\mathbb{I}}$ is definable as an adjoint in the above sense, somewhere this issue must emerge.

Indeed, the crucial issue is that not all modalities give rise to an *internal functor*. Categorically, an internal functor is one whose action on morphisms can be defined as an arrow $A^B \to F(B)^{F(A)}$.[16] Of course, an immediate problem is that these two arrows do not live in the same category in our case: one lives in mode $m$ and the other in mode $n$. Even supposing we are considering an endoadjunction, such

[16]This is a special case of an enriched functor, making use of the observation that a cartesian closed category is self-enriched.

a term could still not be constructed. The functorial action of a modality does not extend to arbitrary terms. That is, for the walking adjunction we do not have a term of the following type:[17]

$$(A \to B) \to (\langle \mu \mid A \rangle \to \langle \mu \mid B \rangle)$$

Instead, we have something akin to axiom K, which gives us $\langle \mu \mid A \to B \rangle \to (\langle \mu \mid A \rangle \to \langle \mu \mid B \rangle)$. Taking advantage of the fact that any term constructible in a closed context is constructible under a modality, this yields a functor on *closed terms*. In general, therefore, we cannot define the transposition operator one might hope for, some isomorphism

$$(\langle \nu \mid A \rangle \to B) \cong (A \to \langle \mu \mid B \rangle)$$

Instead, we have a pair of terms where $\Gamma.\text{🔒}_{\nu \circ \mu} \vdash A\ \mathsf{type}_1 \mathbin{@} m$ and $\Gamma.\text{🔒}_{\mu} \vdash B\ \mathsf{type}_1 \mathbin{@} m$ for the first and $\Gamma.\text{🔒}_{\nu} \vdash A\ \mathsf{type}_1 \mathbin{@} m$ and $\Gamma.\text{🔒}_{\nu \circ \mu} \vdash B\ \mathsf{type}_1 \mathbin{@} m$ for the second:

$$\begin{aligned}
&\mathbf{transp}^{\to}_{\nu \dashv \mu} \ : \ \langle \mu \mid \langle \nu \mid A \rangle \to B \rangle \to A[\text{🔑}^{1}_{\mu \circ \nu}[\eta]] \to \langle \mu \mid B \rangle \\
&\mathbf{transp}^{\to}_{\nu \dashv \mu} \triangleq \lambda f.\ \lambda x.\ f \circledast_{\mu} u(x)
\end{aligned}$$

$$\begin{aligned}
&\mathbf{transp}^{\leftarrow}_{\nu \dashv \mu} \ : \ \langle \nu \mid A \to \langle \mu \mid B \rangle \rangle \to \langle \nu \mid A \rangle \to B[\text{🔑}^{\nu \circ \mu}_{1}[\epsilon]] \\
&\mathbf{transp}^{\leftarrow}_{\nu \dashv \mu} \triangleq \lambda f.\ \lambda x.\ e(f \circledast_{\nu} x)
\end{aligned}$$

In certain cases we can simplify these operations, albeit with loss of power. For instance, if these are endoadjunctions and we have an initial modality $\bot$, then we could construct transpositions of the following type:

$$\begin{gathered}
\langle \bot \mid \langle \nu \mid A \rangle \to B \rangle \to \langle \bot \mid A[\text{🔑}^{\bot \circ \nu}_{\bot}] \to \langle \mu \mid B[\text{🔑}^{\bot \circ \nu}_{\bot}] \rangle \rangle \\
\langle \bot \mid A \to \langle \mu \mid B \rangle \rangle \to \langle \bot \mid \langle \nu \mid A[\text{🔑}^{\bot}_{\bot \circ \nu}] \rangle \to B[\text{🔑}^{\bot \circ \mu}_{\bot}] \rangle
\end{gathered}$$

In the case of Licata et al. [Lic+18] this $\bot$ modality is precisely the global sections modality and these operators are the appropriate transpositions required by their constructions.[18]

The status of transposition in MTT is therefore complex, the naïve transposition operation is not even well-typed, and there are variety of possible replacements. It is worth emphasizing, however, that these replacements do not require extensions to the mode theory: they are all constructible from the unit and counit, for which there is no problem of internal versus external.

### 2.8.2 Crisp or Modal Induction Principles

Recall the typing rule for $\mathsf{let}_{\nu}\ \mathsf{mod}_{\mu}(\_) \leftarrow M_0\ \mathsf{in}\ M_1$ from Section 1.2:

$$\frac{\begin{array}{c} \nu : \mathrm{Hom}_{\mathcal{M}}(o, n) \\ \mu : \mathrm{Hom}_{\mathcal{M}}(n, m) \qquad \Gamma\ \mathsf{ctx} \mathbin{@} m \qquad \Gamma.\text{🔒}_{\mu}.\text{🔒}_{\nu} \vdash A\ \mathsf{type}_1 \mathbin{@} o \qquad \Gamma.\text{🔒}_{\mu} \vdash M_0 : \langle \nu \mid A \rangle \mathbin{@} n \\ \Gamma.(\mu \mid \langle \nu \mid A \rangle) \vdash B\ \mathsf{type}_1 \mathbin{@} m \qquad \Gamma.(\mu \circ \nu \mid A) \vdash M_1 : B[\uparrow.\mathsf{mod}_{\nu}(\mathbf{v}_0)] \mathbin{@} m \end{array}}{\Gamma \vdash \mathsf{let}_{\mu}\ \mathsf{mod}_{\nu}(\_) \leftarrow M_0\ \mathsf{in}\ M_1 : B[\mathsf{id}.M_0] \mathbin{@} m}$$

Notice that there is an "extra" modality parameterizing this rule, $\nu$, which modifies $M_0$ as well as the data supplied to $M_1$. This extra generality is not frivolous; we can only define $\mathbf{comp}_{\nu,\mu}$ in Section 1.3.1 because we can eliminate a modality "under" another.

[17] Indeed, if we had a term of this type, we could easily show that all modalities contain a point (a map $A \to \langle \mu \mid A \rangle$) which would trivialize any comonads.

[18] Note that, while Licata et al. [Lic+18] have the internal types $\langle \nu \mid - \rangle = \wp$ and $\langle \mu \mid - \rangle = \surd$, the type system they use has only special judgemental support for the global sections modality $\flat$ and not for $\nu$ and $\mu$.

One might hope that a similar level of flexibility for all pattern matching-type elimination rules. However, the current rule for booleans does not include this extra modality:

$$\frac{\Gamma\ \mathsf{ctx} \mathbin{@} m \qquad \Gamma.(1 \mid \mathbb{B}) \vdash A\ \mathsf{type}_1 \mathbin{@} m \\ \Gamma \vdash M_t : A[\mathsf{id}.\mathsf{tt}] \mathbin{@} m \qquad \Gamma \vdash M_f : A[\mathsf{id}.\mathsf{ff}] \mathbin{@} m \qquad \Gamma.🔒_1 \vdash N : \mathbb{B} \mathbin{@} m}{\Gamma \vdash \mathsf{if}(A; M_t; M_f; N) : A[\mathsf{id}.N] \mathbin{@} m}$$

Indeed, if we changed $1$ to an arbitrary $\nu$, then this rule would state something considerably stronger: not only do we have the expected elimination principle for $\mathbb{B}$, but all of our modalities would have to *preserve* $\mathbb{B}$. Semantically, this is nonsense: modalities correspond to right adjoints and right adjoints do not necessarily preserve colimits. For a concrete example, consider axiomitizing the irrelevance modality. This modality preserves all limits, and we can arrange it into a dependent right adjoint. If we could somehow prove the "stronger" boolean elimination rule, we would be able to case on $\mathbb{B}$ when it appears in an irrelevant term. This would mean that we could construct two computations which behave differently when supplied with different "irrelevant" arguments, which was precisely what irrelevance was meant to prevent. The issue here is that while irrelevance preserves limits, it does *not* preserve colimits, and in particular it does not preserve $\mathbb{B}$.

In what circumstances can we safely recover the stronger elimination rules? If the stronger boolean elimination principle corresponded to the preservation of booleans, it seems reasonable to expect that we can recover it when $\nu$ is a left adjoint: left adjoints preserve colimits. This idea underlies the *crisp induction principles* in Shulman [Shu18]. There, the adjunction $\flat \dashv \sharp$ was sufficient to recover modalized elimination principles for the identity type, coproducts, and others. We will demonstrate that the same principle can be applied to MTT when the mode theory specifies an adjunction of modalities.

**Theorem 2.8.4.** $\langle \nu \mid \mathbb{B} \rangle \simeq \mathbb{B}$

*Proof.* Rather than directly constructing the equivalence, it will prove slightly easier to factor this process into two steps:

1. First, we define a general purpose *crisp induction principle* for $\langle \nu \mid \mathbb{B} \rangle$. This construction mirrors the one in Shulman [Shu18], though generalized slightly to not rely on the idempotence of any modalities.

2. We use this crisp induction principle to construct both the maps *and* the proofs that these maps are suitably inverse to each other.

For the definition of crisp if, parameterize what follows by the motive, $\Gamma.🔒_{\nu \circ \mu}.(\nu \mid \mathbb{B}) \vdash C\ \mathsf{type}_1 \mathbin{@} m$.

$$\begin{array}{ll}
\Gamma.🔒_\nu \vdash h & : (b : \mathbb{B}) \to \langle \mu \mid C(\mathsf{tt}) \rangle \to \langle \mu \mid C(\mathsf{ff}) \rangle \to \langle \mu \mid C[\uparrow.🔒_\mu.b^\eta] \rangle \mathbin{@} n \\
h(b, t, f) & \triangleq \mathsf{if}(b.\ \langle \mu \mid C(b^\eta) \rangle; t; f; b)
\end{array}$$

$$\begin{array}{ll}
\Gamma \vdash \mathsf{crisp_if}_C & : (b : (\nu \mid \mathbb{B})) \to \langle \nu \circ \mu \mid C(\mathsf{tt}) \rangle \to \langle \nu \circ \mu \mid C(\mathsf{ff}) \rangle \to C[(🔑^\epsilon_\Gamma \circ \uparrow).b] \mathbin{@} m \\
\mathsf{crisp_if}_C(b, t, f) & \triangleq e(\mathsf{mod}_\nu(h(b)) \circledast_\nu t \circledast_\nu f)
\end{array}$$

It is slightly subtle to see that this definition is well-typed. In particular, in order to see that the substitution applied to $C$ is correct, we make use of the following calculation:

$$
\begin{aligned}
&\left(\uparrow.\text{🔒}_{\mu}.\mathbf{v}_0[\text{🔑}^{\eta}_{\Gamma.\text{🔒}_{\nu}.(1|\mathbb{B})}]\right) \circ (\uparrow.\text{🔒}_{\nu}.\mathbf{v}_0.\text{🔒}_{\mu}) \circ \text{🔑}^{\epsilon}_{\Gamma.(\nu|\mathbb{B})} \\
&= \left(\uparrow.\text{🔒}_{\nu\circ\mu}.\mathbf{v}_0[\text{🔑}^{\eta}_{\Gamma.\text{🔒}_{\nu}.(1|\mathbb{B})} \circ \uparrow.\text{🔒}_{\nu}.\mathbf{v}_0.\text{🔒}_{\mu\circ\nu}]\right) \circ \text{🔑}^{\epsilon}_{\Gamma.(\nu|\mathbb{B})} \\
&= \left(\uparrow.\text{🔒}_{\nu\circ\mu}.\mathbf{v}_0[\uparrow.\text{🔒}_{\nu}.\mathbf{v}_0 \circ \text{🔑}^{\eta}_{\Gamma.(\nu|\mathbb{B}).\text{🔒}_{\nu}}]\right) \circ \left(\text{🔑}^{\epsilon}_{\Gamma.(\nu|\mathbb{B})}.\text{🔒}_{\mu}\right) \\
&= \left(\uparrow.\text{🔒}_{\nu\circ\mu}.\mathbf{v}_0[\text{🔑}^{\eta}_{\Gamma.(\nu|\mathbb{B}).\text{🔒}_{\nu}}]\right) \circ \text{🔑}^{\epsilon}_{\Gamma.(\nu|\mathbb{B})} \\
&= \left(\uparrow.\text{🔒}_{\nu\circ\mu} \circ \text{🔑}^{\epsilon}_{\Gamma.(\nu|\mathbb{B})}.\mathbf{v}_0[\text{🔑}^{\nu\star\eta}_{\Gamma.(\nu|\mathbb{B})} \circ \text{🔑}^{\epsilon\star\nu}_{\Gamma.(\nu|\mathbb{B})}]\right) \\
&= (\text{🔑}^{\epsilon}_{\Gamma} \circ \uparrow).\mathbf{v}_0
\end{aligned}
$$

This uses the triangle identities again. The calculation is very similar to the reasoning used in order to show that $e$ and $u$ satisfies the triangle identities.

With crisp_if in hand, we can take actually construct the required equivalence.

$$
\begin{aligned}
&\mathsf{b} &&: \langle \nu \mid \mathbb{B} \rangle \to \mathbb{B} \mathrel{@} m \\
&\mathsf{b}(x) &&\triangleq \mathsf{let}\ \mathsf{mod}_{\nu}(x') \leftarrow x\ \mathsf{in}\ h(x') \\
& && \quad \text{where } h(x) \triangleq \mathsf{crisp\_if}_{\mathbb{B}}(\mathsf{mod}_{\nu\circ\mu}(\mathsf{tt}), \mathsf{mod}_{\nu\circ\mu}(\mathsf{ff}), x)
\end{aligned}
$$

$$
\begin{aligned}
&\mathsf{b}^{-1} &&: \mathbb{B} \to \langle \nu \mid \mathbb{B} \rangle \mathrel{@} m \\
&\mathsf{b}^{-1} &&\triangleq \lambda x.\ \mathsf{if}(\_.\ \langle \nu \mid \mathbb{B} \rangle; \mathsf{mod}_{\nu}(\mathsf{tt}); \mathsf{mod}_{\nu}(\mathsf{ff}); x) \\
&\_ &&: (b : \langle \nu \mid \mathbb{B} \rangle) \to \mathsf{Id}_{\langle \nu|\mathbb{B} \rangle}(b, \mathsf{b}^{-1}(\mathsf{b}(b))) \mathrel{@} m \\
&\_(x) &&\triangleq \mathsf{let}\ \mathsf{mod}_{\nu}(x') \leftarrow x\ \mathsf{in}\ h(x') \\
& && \quad \text{where } h(x) \triangleq \mathsf{crisp\_if}_{b.\ \mathsf{Id}_{\langle \nu|\mathbb{B} \rangle}(\mathsf{mod}_{\nu}(b), \mathsf{b}^{-1}(h(b)))}(\mathsf{mod}_{\nu\circ\mu}(\mathsf{refl}(\mathsf{mod}_{\nu}(\mathsf{tt}))), \mathsf{mod}_{\nu\circ\mu}(\mathsf{refl}(\mathsf{mod}_{\nu}(\mathsf{ff}))), x))
\end{aligned}
$$

$$
\begin{aligned}
&\_ &&: (b : \mathbb{B}) \to \mathsf{Id}_{\mathbb{B}}(b, \mathsf{b}(\mathsf{b}^{-1}(b))) \mathrel{@} m \\
&\_(x) &&\triangleq \mathsf{if}(b.\ \mathsf{Id}_{\mathbb{B}}(b, \mathsf{b}(\mathsf{b}^{-1}(b))); \mathsf{refl}(\mathsf{tt}); \mathsf{refl}(\mathsf{ff}); x)
\end{aligned}
$$

□

**Theorem 2.8.5.** $\langle \nu \mid \mathsf{Id}_A(M_0, M_1) \rangle \simeq \mathsf{Id}_{\langle \nu|A \rangle}(\mathsf{mod}_{\nu}(M_0), \mathsf{mod}_{\nu}(M_1))$

*Proof.* As in Theorem 2.8.4, we will start by constructing a general modal induction principle and then use this induction principle to prove this equivalence.

For the definition of crisp J, let us fix $\Gamma.\text{🔒}_{\nu} \vdash A\ \mathsf{type}_1 \mathrel{@} mn$ and the motive:

$$\Gamma.\text{🔒}_{\nu\circ\mu}, x_0 : (\nu \mid A^{\nu\star\eta}), x_1 : (\nu \mid A^{\nu\star\eta}), p : (\nu \mid \mathsf{Id}_{A^{\nu\star\eta}}(x_0, x_1)) \vdash C\ \mathsf{type}_1 \mathrel{@} m$$

We now define crisp J as follows:

$$
\begin{aligned}
&\Gamma.\text{🔒}_{\nu} \vdash h &&: (x_0, x_1 : A)(p : \mathsf{Id}_A(x_0, x_1)) \to \\
& && \quad \langle \mu \mid (a : (\nu \mid A^{\nu\star\eta})) \to C(a, a, \mathsf{refl}(a)) \rangle \to \\
& && \quad \langle \mu \mid C[\uparrow^3.\text{🔒}_{\mu}.x_0^{\eta}.x_1^{\eta}.p^{\eta}] \rangle \mathrel{@} n \\
&h(x_0, x_1, p, b) &&\triangleq \mathsf{J}(a_0, a_1, p.\ \langle \mu \mid C(x_0^{\eta}, x_1^{\eta}, p^{\eta}) \rangle, b, p) \\[1ex]
&\Gamma \vdash \mathsf{crisp\_J}_C &&: (x_0, x_1 : (\nu \mid A))(p : (\nu \mid \mathsf{Id}_A(x_0, x_1))) \to \\
& && \quad \langle \nu \circ \mu \mid (a : (\nu \mid A^{\nu\star\eta})) \to C(a, a, \mathsf{refl}(a)) \rangle \to \\
& && \quad C[(\text{🔑}^{\epsilon}_{\Gamma} \circ \uparrow^3).x_0.x_1.p] \mathrel{@} m \\
&\mathsf{crisp\_J}_C(x_0, x_1, p, b) &&\triangleq e(\mathsf{mod}_{\nu}(h(x_0, x_1, p)) \circledast_{\nu} b)
\end{aligned}
$$

We can now construct the desired equivalence directly. Let us suppose we have $\Gamma.\text{🔒}_{\nu} \vdash A\ \mathsf{type}_1 \mathbin{@} \nu$ and $\Gamma.\text{🔒}_{\nu} \vdash M_0, M_1 : A \mathbin{@} \nu$:

$$
\begin{array}{ll}
\mathsf{id} & : \langle \nu \mid \mathsf{Id}_A(M_0, M_1)\rangle \to \mathsf{Id}_{\langle \nu \mid A\rangle}(\mathsf{mod}_{\nu}(M_0), \mathsf{mod}_{\nu}(M_1)) \mathbin{@} m \\
\mathsf{id}(p) & \triangleq \mathsf{let}\ \mathsf{mod}_{\nu}(p') \leftarrow p\ \mathsf{in}\ h(p') \\
& \quad \text{where } h(p) \triangleq \mathsf{crisp\_J}_{x_0, x_1, p.\ \mathsf{Id}_{\langle \nu \mid A^{\nu \star \eta}\rangle}(x_0, x_1)}(\mathsf{mod}_{\nu \circ \mu}(\lambda a.\ \mathsf{refl}(\mathsf{mod}_{\nu}(a))), M_0, M_1, p) \\
\\
\mathsf{id}^{-1} & : \mathsf{Id}_{\langle \nu \mid A\rangle}(\mathsf{mod}_{\nu}(M_0), \mathsf{mod}_{\nu}(M_1)) \to \langle \nu \mid \mathsf{Id}_A(M_0, M_1)\rangle \mathbin{@} m \\
\mathsf{id}^{-1}(p) & \triangleq \mathsf{J}(M, x.\ \mathsf{let}\ \mathsf{mod}_{\nu}(x') \leftarrow x\ \mathsf{in}\ \mathsf{mod}_{\nu}(\mathsf{refl}(x')), p) \\
& \quad \text{where } M(\mathsf{mod}_{\nu}(x_0), \mathsf{mod}_{\nu}(x_1), p) \triangleq \langle \nu \mid \mathsf{Id}_A(x_0', x_1')\rangle \\
\_ & : (p : \langle \nu \mid \mathsf{Id}_A(M_0, M_1)\rangle) \to \mathsf{Id}_{\langle \nu \mid \mathsf{Id}_A(M_0, M_1)\rangle}(p, \mathsf{id}^{-1}(\mathsf{id}(p))) \mathbin{@} m \\
\_(p) & \triangleq \mathsf{let}\ \mathsf{mod}_{\nu}(p') \leftarrow p\ \mathsf{in}\ h(p') \\
& \text{where } h(p) \triangleq \mathsf{crisp\_J}_{x_0, x_1, p.\ \mathsf{Id}_{\langle \nu \mid \mathsf{Id}_{A^{\nu \star \eta}}(x_0, x_1)\rangle}(\mathsf{mod}_{\nu}(p), \mathsf{id}^{-1}(h(p)))}(\mathsf{mod}_{\nu \circ \mu}(\lambda x.\ \mathsf{refl}(\mathsf{mod}_{\nu}(\mathsf{refl}(x)))), M_0, M_1, p)) \\
\\
\_ & : (p : \mathsf{Id}_{\langle \nu \mid A\rangle}(\mathsf{mod}_{\nu}(M_0), \mathsf{mod}_{\nu}(M_1))) \to \mathsf{Id}_{\mathsf{Id}_{\langle \nu \mid A\rangle}(\mathsf{mod}_{\nu}(M_0), \mathsf{mod}_{\nu}(M_1))}(p, \mathsf{id}(\mathsf{id}^{-1}(p))) \mathbin{@} m \\
\_(p) & \triangleq \mathsf{J}(M, x.\ \mathsf{let}\ \mathsf{mod}_{\nu}(x') \leftarrow x\ \mathsf{in}\ \mathsf{mod}_{\nu}(\mathsf{refl}(x')), p) \\
& \quad \text{where } M(\mathsf{mod}_{\nu}(x_0), \mathsf{mod}_{\nu}(x_1), p) = \mathsf{Id}_{\mathsf{Id}_{\langle \nu \mid A\rangle}(\mathsf{mod}_{\nu}(x_0), \mathsf{mod}_{\nu}(x_1))}(p, \mathsf{id}(\mathsf{id}^{-1}(p)))
\end{array}
$$

For the last equality proof we have adopted the informal pattern matching syntax one might expect in an implementation of MTT; without this syntactic nicety the motive is too unreadable for a paper proof. $\square$

## 2.9 Relative Realizability

Thus far we have limited our examples to certain *presheaf toposes*. This simplifies matters because the semantics of dependent type theory and dependent right adjoints are both well understood in this context. There is, however, no fundamental restriction in MTT that requires us to work with presheaves.

In this section we turn our attention to examples arising in categorical realizability. These include the category of assemblies, triposes, and realizability toposes. The use of assemblies as a model of dependent type theory is far from novel, but we recall some definitions and details here.

### 2.9.1 Preliminary Aspects of Categorical Realizability

To begin with, our notion of realizability isolates a particular abstract model of computation: a PCA. This is a combinatorial definition which is concise to state and easy to work with, if inconvenient to actually program in. We refer the reader to Van Oosten [Oos08] and Longley and Normann [LN15] for a comprehensive summary.

**Definition 2.9.1** (Partial Combinatory Algebra)**.** A partial combinatory algebra is a set $A$ equipped with a partial operator $\cdot : A \times A \rightharpoonup A$. This operator represents application and associates to the left, we will often suppress it entirely, writing $ab$. Moreover, there must be distinguished elements $S, K \in A$ satisfying the following:

- $S \cdot a$, $S \cdot a \cdot b$ are both defined for all $a, b \in A$.
- $S \cdot a \cdot b \cdot c \simeq (a \cdot c) \cdot (b \cdot c)$.
- $K \cdot a$ is defined for all $a \in A$.
- $K \cdot a \cdot b = a$

With this abstract notion of computation, we can construct a category which "glues" the category of sets to $A$, such that arrows between these sets are computable.

**Definition 2.9.2** (Assembly)**.** An assembly is a pair $(X, \Phi)$ of a set $X$ and a map $X \to \mathcal{P}(A) \setminus \{\emptyset\}$. A morphism between assemblies $X \xrightarrow{f} Y$ is a set-theoretic function $f : X \to Y$ such that there exists an element $a \in A$ satisfying $\forall x \in X, b \in \Phi_X(x).\ ab \in \Phi_Y(f(x))$. We will say that $a$ *tracks* $f$ and write $a \Vdash f$.

**Theorem 2.9.3.** *The category of assemblies over a PCA,* $\mathbf{Asm}(A)$ *is a locally cartesian closed regular category.*

*Proof.* This is a standard result. A detailed textbook proof is given by Van Oosten [Oos08] for regularity and cartesian closure. $\square$

**Definition 2.9.4** (Uniform Family)**.** A uniform family $(I, X_i)$ is a pair of an assembly $I$ and a family of assemblies $(X_i)_I$ indexed over the underlying set of $I$. A morphism of uniform families is a pair of two functions, $(I, X_i) \xrightarrow{(f,g)} (J, Y_j)$ where $f : I \to J$ is a map of assemblies, and $g : X_i \to Y_j$ is an indexed family of maps $g_i : X_i \to Y_{f(i)}$. Moreover, we require that $g$ be *uniformly tracked*, that is, there a code $a \in A$ such that for all $i \in I$ and $n \in \Phi_I(i)$, $e \cdot i \Vdash g_i$.

**Theorem 2.9.5.** $\mathbf{UFam}(A)$ *is a split fibration of* $\mathbf{Asm}(A)$ *and equivalent to* $\mathsf{cod} : \mathbf{Asm}(A)^{\to} \to \mathbf{Asm}(A)$.

*Proof.* Another standard result, proven in Jacobs [Jac99] for example. We observe that the splitting induces a functor $\mathbf{Asm}(A)^{\mathsf{op}} \to \mathbf{Cat}$ which acts by precomposition on families. That is, $f^*((I, (X_i)_{i\in I})) = (J, (X_{f(j)})_{j\in J})$. $\square$

**Corollary 2.9.6.** $\mathbf{UFam}(A)$ *is a model of (extensional) dependent type theory, with contexts being drawn from* $\mathbf{Asm}(A)$ *and types in context* $X$ *being uniform families over* $X$*.*

*Proof.* This is proven in Jacobs [Jac99] using categories with attributes rather than natural models. We will discuss this difference in greater detail in a moment. □

**Theorem 2.9.7.** *Each Grothendieck universe* $\mathcal{V}$ *induces a universe in* $\mathbf{UFam}(A)$*, generic for all (fiberwise)* $\mathcal{V}$*-small types.*

*Proof.* The universe is given by the following uniform family over $1$ (which is just an assembly) $\mathsf{U} = ((X \in \mathcal{V}) \times X \to \mathcal{P}(A), \lambda\_.\ A)$. The generic fibration is given by $\mathsf{U}$ together with the following assembly over it:

$$\widetilde{U} = (\textstyle\sum_{(A,\Phi_A)\in \mathsf{U}} A, \lambda((A,\Phi_A),x).\ \Phi_A(x))$$

A textbook account of this proof can be found in Luo [Luo94]. □

Note that we are not asking of for the *impredicative* universe of modest sets [Jac99], merely the standard predicative universes induced by our ambient set theory. While an impredicative universe could be incorporated into our framework, there is no need for impredicativity in what follows.

**Corollary 2.9.8.** $\mathbf{UFam}(A)$ *is a model of Martin-Löf Type Theory with a hierarchy of universes à la Coquand.*

*Proof.* This is an immediate corollary of Theorems 2.9.5 and 2.9.7. Since, however, we have used natural models throughout the rest of this work we will take a moment to show how this structure explicitly lifts to a natural model. The distinction is purely formal: full, split comprehension categories, cwfs, and natural models are all equivalent.

First, we take the category of contexts to be $\mathbf{Asm}(A)$. The presheaf of types over it is defined by sending $\Gamma$ to the uniform families over $\Gamma$. The presheaf of terms consists is defined as follows:

$$\Gamma \mapsto \textstyle\sum_{(\Gamma,A):\mathbf{UFam}(A)} \mathrm{Hom}_{\mathbf{UFam}(A)}((\Gamma,1),A)$$

In both cases, the action of substitution is given by precomposition and changing the indices appropriately in the families.

Context extension comes from the comprehension structure, but explicitly given an assembly $\Gamma$ and a uniform family $A$ over it:

$$\Gamma.A \triangleq (\Gamma \times A, \lambda(\gamma,a).\ \Phi_\Gamma(\gamma) \wedge \Phi_{A(\gamma)}(a))$$

Where $\wedge$ is the standard "cartesian product" of sets of a realizers:

$$U \wedge V = \{a \in A \mid \pi_1 \cdot a \in U \wedge \pi_2 \cdot a \in V\}$$

It is a routine calculation to show that this induces the required natural model structure. The remaining definitions of dependent sums, products, equality, etc. are likewise transported between the equivalence of cwfs and categories with attributes. We do not actually need to inspect how these are constructed in what follows, however, we will not write out the definitions here. The full proof may be found in Hofmann [Hof97]. □

Already in this framework there are evident modalities. For instance, the discrete functor $\nabla : \mathbf{Set} \to \mathbf{Asm}(A)$ induces a size-preserving dependent right adjoint.[19] This allows us to embed non-computable data into a universe of computable functions. Further adjoints, however, naturally arise in the context of *relative realizability*.

[19]It is perhaps confusing that objects $\nabla(A)$ are called *discrete objects* in realizability theory, because $\nabla$ behaves precisely like the *codiscrete* functor in axiomatic cohesion.

### 2.9.2 *Preliminary Definitions for Relative Realizability*

**Definition 2.9.9** (Relative PCA)**.** A relative PCA is a PCA with a chosen subset $A_\sharp \subseteq A$ which is closed under application and contains $S$ and $K$

One should intuitively see $A_\sharp$ as the collection of computable elements of $A$ while $A$ itself may contain other (continuous) elements. See Birkedal [Bir00] for examples of relative PCAs.

**Definition 2.9.10** (Relative Assemblies)**.** The category of relative assemblies, $\mathbf{Asm}(A, A_\sharp)$ has as objects the objects of $\mathbf{Asm}(A)$, but morphisms are required to be tracked by an element of $A_\sharp$.

We can now define the functors we wish to interpret in our modal type theory:

$$
\begin{tikzcd}
\mathbf{Asm}(A, A_\sharp) \arrow[d, "\Gamma" description] \\
\mathbf{Asm}(A_\sharp) \arrow[u, bend left=70, "\Delta"] \arrow[u, bend right=70, "\nabla"']
\end{tikzcd}
\qquad \Delta \dashv \Gamma \dashv \nabla
$$

First, the definition of $\Delta$ is the straightforward inclusion. We know that $A_\sharp \subseteq A$, and so an assembly over $A_\sharp$ is a specific case of an assembly over $A$. Moreover, the definition of morphism is the same in these two categories so this functor is full and faithful. The definition of $\Gamma$ is only moderately more complex:

$$\Gamma(X, \Phi_X) = (X, \lambda x.\ A_\sharp \cap \Phi_X(x))$$

In other words, $\Gamma$ removes the non-computable realizers from each assembly. The definition on morphisms is trivial since we have already required that morphisms be computable and $A_\sharp$ is closed under application. Finally, the definition of $\nabla$. This is intuitively meant to "pad" each set of realizers with arbitrary computable data, but its definition is slightly more complex than this:

$$\nabla(X, \Phi_X) = (X, \lambda x. \bigcup_{\phi \subseteq A} \phi \wedge (\phi \cap A_\sharp \supset \Phi_X(x)))$$

In this definition we have used some notation common to tripos theory and categorical realizability more generally. In particular,

$$
\begin{aligned}
U \wedge V &= \{a \in A \mid \pi_1 \cdot a \in U \wedge \pi_2 \cdot a \in V\} \\
U \supset V &= \{a \in A \mid \forall b \in U.\ a \cdot b \in V\}
\end{aligned}
$$

We now wish to show that $\Gamma$ and $\nabla$ can be extended to functors with an action on $\mathbf{UFam}(A_\sharp)$ and $\mathbf{UFam}(A, A_\sharp)$. The action on objects is obvious for both, sending $(I, X_{i \in I})$ to $(F(I), (F(X_i))_{i \in F(I)})$. This definition is taking advantage of the fact that both of these maps behave as the *identity* on all the set structure. So, for instance, $\Gamma$ has no impact on the underling set of the indexing assembly $I$. The more difficult question is to see that this has an action on terms (which is sufficient to see that that this has an action on terms).

**Lemma 2.9.11.** $\Gamma$ *lifts to a dependent right adjoint.*

*Proof.* We have already defined how $\Gamma$ acts on types, it remains to show how it acts on terms (morphisms in the $\mathbf{UFam}(A_\sharp)$ fibers) and to show that it respects context extension weakly and substitution strictly.

First, for the action on arrows in $\mathbf{UFam}(A_\sharp)$, given an arrow $(j, y) : (I, X_{i\in I}) \to (J, Y_{j\in J})$, we must show that $(j, y) : (\Gamma(I), (\Gamma(X_i))_{i\in I}) \to (\Gamma(J), (\Gamma(Y_j))_{j\in J})$ is tracked. We know that there is some $e_j \Vdash j$ in $\mathbf{Asm}(A, A_\sharp)$. Therefore, this same $e_j$ tracks $\Gamma(I) \xrightarrow{j} \Gamma(J)$ in $\mathbf{Asm}(A_\sharp)$. Next, we must also have that there is a realizer $e_g \in A_\sharp$, such that for all $i$ and $a_i \in \Phi_I(i)$, $e_g \cdot i \Vdash y_i$. By precisely the same argument as for $e_j$, we can see that $e_g$ is also a valid realizer for $g$. This definition ensures that $\Gamma$ has an extension to types and terms.

Next, it is immediate that this functor strictly preserve substitution, as substitution is given essentially by "precomposition" in the index of the family. What remains to be shown is that context extension is preserved up to isomorphism.

Recall that context extension in $\mathbf{Asm}(A)$ is defined as follows:

$$\Delta.A = (\textstyle\sum_{\gamma\in\Delta} A_\gamma, \lambda(\gamma, a).\ \Phi_\Delta(\gamma) \wedge \Phi_{A_\gamma}(a))$$

However, taking the intersection of $U \wedge V$ with $A_\sharp$ is computably equivalent to $(U \cap A_\sharp) \wedge (V \cap A_\sharp)$. Therefore $\Gamma(\Delta.A) \cong \Gamma(\Delta).\Gamma(A)$, as required. $\square$

**Lemma 2.9.12.** $\nabla$ *extends to a dependent right adjoint.*

*Proof.* We have exactly the same proof obligations as Lemma 2.9.12. To start with, suppose again that we have $(j, y) : (I, X_{i\in I}) \to (J, Y_{j\in J})$ in $\mathbf{UFam}(A_\sharp)$, we wish to show that $F(j, y)$ induces an arrow in $\mathbf{UFam}(A, A_\sharp)$. As before, we keep the same set-theoretic arrows. What remains to be shown is that they are still tracked. We had a realizer $e_j \Vdash j$ before, the new realizer is defined as $\lambda^*\langle p, a\rangle.\ \langle p, \lambda^* x.\ e_j(a(x))\rangle$. In order to show that this is type-correct, let us first suppose that we have some realizer $\langle p, a\rangle \in \Phi_{\nabla I}(i)$. We then have that $p \in \phi \subseteq A$ and $a \in (\phi \cap A_\sharp \supset \Phi_X(x))$. Therefore, we can pick the same $\phi$ to instantiate the existential quantifier in $\Phi_{\nabla J}(f(i))$, and it is easily observed that $\langle p, \lambda^* x.\ e_j(a(x))\rangle$ indeed has the correct type. Now, the same basic procedure applies to the uniform realizer in the fibers. Given $e_g$ which tracks $g$ uniformly in each fiber, we define the following:

$$\lambda^* i.\ \lambda^*\langle p, a\rangle.\ \langle p, \lambda^* x.\ e \cdot i \cdot (a \cdot x)\rangle$$

It is another tedious but routine inspection to see that this has the correct type. It is again easily seen that this strictly commutes with substitution. It remains to show that this preserves context extension. Suppose again we have $\Delta$ and $\Delta \vdash A$. We know that the following defines context extension:

$$\Delta.A = (\textstyle\sum_{\gamma\in\Delta} A_\gamma, \lambda(\gamma, a).\ \Phi_\Delta(\gamma) \wedge \Phi_{A_\gamma}(a))$$

Now, $\nabla(\Delta.A)$ is therefore equipped with the following set of realizers for $\langle\gamma, a\rangle$:

$$\textstyle\bigcup_{\phi\subseteq A} \phi \wedge (\phi \cap A_\sharp \supset \Phi_\Delta(\gamma) \wedge \Phi_{A_\gamma}(a))$$

Now, on the other hand, at $\langle\gamma, a\rangle$, we have that $\nabla(\Gamma).\nabla(A)$ has the following set of realizers:

$$\left(\textstyle\bigcup_{\phi\subseteq A} \phi \wedge (\phi \cap A_\sharp \supset \Phi_\Delta(\gamma))\right) \wedge \left(\textstyle\bigcup_{\phi\subseteq A} \phi \wedge (\phi \cap A_\sharp \supset \Phi_{A(\gamma)}(a))\right)$$

In order to complete the isomorphism, it suffices to give any pair of realizers (not necessarily inverses!) which go between these two sets. It is obvious how to from first to the second. Going from the second to the first requires a small degree of creativity:

$$\lambda^*\langle\langle p_1, a_1\rangle, \langle p_2, a_2\rangle\rangle.\ \langle\langle p_1, p_2\rangle, \lambda^*\langle x_1, x_2\rangle.\ \langle a_1 \cdot x_1, a_2 \cdot x_2\rangle\rangle$$

In particular, we vary the choice of $\phi$ between the two. If we are given realizers at $\phi_1$ and $\phi_2$, we produce a realizer at $\phi_1 \wedge \phi_2$. $\square$

**Lemma 2.9.13.** *Both* $\Gamma$ *and* $\nabla$ *preserve* $\mathsf{U}$*-smallness.*

*Proof.* This follows from the fact that both $\Gamma$ and $\nabla$ have a trivial action on the sets underlying the assemblies. □

These results together tell us that the adjoint situation induced by relative realizability is entirely within the grasp of MTT.

### 2.9.3 MTT for Relative Realizability

Now that we have proven that these categories of assemblies are models of type theory and that the functors between them are dependent right adjoints, it is straightforward to use MTT to reason about this situation.

For our mode theory, we pick the following:

$$\begin{tikzcd} m \arrow[r, bend left, "\mu"] & n \arrow[l, bend left, "\nu"] \end{tikzcd}$$

We demand that these form an internal adjoint, following Section 2.8:

$$\eta : 1 \Rightarrow \mu \circ \nu \qquad\qquad \epsilon : \nu \circ \mu \Rightarrow 1$$

$$1_\mu = (\mu \star \epsilon) \circ (\eta \star \mu) \qquad\qquad 1_\nu = (\epsilon \star \nu) \circ (\nu \star \eta)$$

We will interpret $\mu$ as $\nabla$ and $\nu$ as $\Gamma$ in our intended model. Under this interpretation, we would also like to ensure that $\nu$ is full and faithful. This can be enforced in the mode theory by requiring $\eta$ to be an isomorphism.

It follows from the calculations in Section 2.8 that they are adjoint to each other. We therefore will content ourselves with showing that $\Gamma$ is full and faithful in this subsection.

First, let us recall the standard categorical argument that if the unit is an isomorphism then the left adjoint is full and faithful.

**Lemma 2.9.14.** *Given an adjunction* $L \dashv R$*, if* $\eta$ *is an isomorphism then* $L$ *is full and faithful.*

*Proof.* We wish to show that $\mathrm{Hom}(A, B) \cong \mathrm{Hom}(L(A), L(B))$. We observe from the adjoint $L \dashv R$ that $\mathrm{Hom}(L(A), L(B))$ is isomorphic to $\mathrm{Hom}(A, R(L(B)))$. However, we can post-compose with the isomorphism $\eta^{-1}$ and conclude that $\mathrm{Hom}(A, R(L(B))) \cong \mathrm{Hom}(A, B)$, completing the proof. □

A complication emerges when we attempt to replay this argument in MTT. As discussed in length in Section 2.8.1, it is not easy to internalize transposition inside MTT. We therefore cannot hope for an equivalence $(A \to B) \simeq (\langle \nu \mid A \rangle \to \langle \nu \mid B \rangle)$. Indeed, such a thing is not even possible to state as written, since $A, B$ must live in mode $n$, and yet $\langle \nu \mid A \rangle, \langle \nu \mid B \rangle$ must live in mode $m$. Instead, we can obtain a similar statement.

**Theorem 2.9.15.** *Assuming function extensionality,* $(A \to B) \simeq \langle \mu \mid \langle \nu \mid A \rangle \to \langle \nu \mid B \rangle \rangle$

*Proof.* We have already seen most of the left-to-right direction:

$$\begin{aligned} &h_0 &&: (A \to B) \to \langle \mu \mid \langle \nu \mid A \rangle \to \langle \nu \mid B \rangle \rangle \\ &h_0(f) &&\triangleq \mathsf{let}\ \mathsf{mod}_\mu(f') \leftarrow \mathbf{comp}_{\mu,\nu}(u(f))\ \mathsf{in}\ \mathsf{mod}_\mu(\lambda a.\ f' \circledast_\nu a) \end{aligned}$$

For the reverse, we use the following map:

$$\begin{aligned} h_1 &: \langle \mu \mid \langle \nu \mid A \rangle \to \langle \nu \mid B \rangle \rangle \to A \to B \\ h_1(f, a) &\triangleq \mathbf{coe}[\eta^{-1} : \mu \circ \nu \Rightarrow 1](f \circledast_\mu u(a)) \end{aligned}$$

The proof that these are appropriately mutually inverse requires function extensionality. With function extensionality, however, it follows the expected patter of "induct and reduce". □

# 3 Conclusions

We have contributed MTT, a type theory for working with multiple interacting modalities. In §1, we developed a precise account of MTT's metatheory and semantics. In §2, we explored the applicability of MTT and demonstrated its utility in working with realistic modal situations.

*Towards an Implementation of MTT* A major point of future work is the development of an implementation of MTT. Substantial preliminary implementation efforts are already underway with `Menkar` [Nuy19]. In addition to the engineering effort, a systematic account for an algorithmic syntax of MTT as well as proof of normalization is needed. We believe that the general ideas of Gratzer, Sterling, and Birkedal [GSB19] are applicable to this situation and a similar series of proofs could be carried out for MTT, perhaps applying more modern *gluing* techniques [Coq18]. The hope would be to prove that the judgments $\Gamma \vdash M = N : A \mathbin{@} m$ and $\Gamma \vdash A = B\ \mathsf{type}_\ell \mathbin{@} m$ are decidable relative to a decision procedure for equality in the underlying mode theory.

We have largely ignore the issue of the decidability of our mode theories in §2, but this issue is central to any implementation of MTT. In particular, some of the mode theories considered in, e.g. Section 2.7, are clearly undecidable. In the case of Section 2.7 for instance, it would be necessary to construct a fixed set of warps for which equality is decidable, but which are still expressive enough to recover most of the original mode theory.

*Left Adjoints* As discussed in Section 1.8.4, MTT trades a measure of generality for a degree of simplicity compared to LSR. One might hope, however, that it would be possible to include a connective for *left adjoints*, as well as the current connective which models right adjoints without losing all of this simplicity. It is not obvious that this can be done without significantly changing MTT; the introduction rule for modalities is exceptionally specific to a right adjoint. This additionally flexibility would allow us to model several modalities which are currently out of reach. For instance, when modeling an adjoint chain we cannot model the final adjoint. If we could include left adjoints, this would no longer be an issue.

*Directed Type Theory and op* In directed type theory [Nor19], there are a wide variety of modalities. One, however, stands out as a distinctly unique phenomenon: $A^{\mathsf{op}}$. However, this does *not* seem to fit into our framework, because $-^{\mathsf{op}}$ does not seem to be part of a dependent right adjoint. Essentially, the left adjoint must be $-^{\mathsf{op}}$ when constructing the term part of the DRA, and the left adjoint must be $1$ (or $-^{\mathsf{co}}$ when working in 2-categories) when constructing the type part of the DRA. It does not seem possible, therefore, to make directed type theory with an opposite modality a straightforward instance of MTT.

In Section 2.4, we built extensions of MTT (Theorems 2.4.7 and 2.4.14) in which terms and their types' codes use the available variables with a different modality. It is not unimaginable that the technique used there may provide a solution to directed type theory.

*Modalities which Model Effects* Unfortunately, it is highly unlikely that any modality which captures an effect would form a modality suitable for MTT. Recall that modalities in MTT preserve products and will also preserve a unit type, if one were added to MTT. This would give us the following equivalence:

$$\langle \mu \mid 1 \rangle \simeq 1$$

If $\langle \mu \mid - \rangle$ is intended to represent an effectful computation, this equivalence tells us that there are no interesting effects possible. If there were, we would have at least 2 distinct inhabitants in $\langle \mu \mid 1 \rangle$: the computation which has an effect and the computation which does not. This immediately rules out modeling Moggi [Mog91] within MTT. Additionally, it means that while Call-by-Push-Value [Lev12] can be partially incorporated in MTT, we can only model one of the two modalities. That is, while we can model the inclusion of values into computations, the reverse is not a (weak) dependent right adjoint.

A framework sufficiently general to include arbitrary monads as modalities must have a notion of modality which does not assume the existence of a left adjoint, and therefore must look quite different than MTT. Since, however, any monad can be decomposed into the composition of a left and right adjoint, a framework which can model left adjoints could likely handle this application.

## *Acknowledgments*

We are grateful for productive conversations with Carlo Angiuli, Dominique Devriese, Adrien Guatto, Magnus Baunsgaard Kristensen, Daniel Licata, Rasmus Ejlers Møgelberg, Matthieu Sozeau, Jonathan Sterling, and Andrea Vezzosi.

We additionally wish to thank Mario Alvarez-Picallo for the title *Type Theory à la Mode*.

# Bibliography


[Abe06] Andreas Abel. "A Polymorphic Lambda-Calculus with Sized Higher-Order Types". PhD thesis. Ludwig-Maximilians-Universität München, 2006 (cit. on p. 63).

[Abe08] Andreas Abel. "Polarised subtyping for sized types". In: *Mathematical Structures in Computer Science* 18.5 (2008), pp. 797–822. DOI: 10.1017/S0960129508006853. URL: https://doi.org/10.1017/S0960129508006853 (cit. on p. 63).

[AGJ14] Robert Atkey, Neil Ghani, and Patricia Johann. "A relationally parametric model of dependent type theory". In: *The 41st Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL '14, San Diego, CA, USA, January 20-21, 2014*. 2014, pp. 503–516. DOI: 10.1145/2535838.2535852. URL: https://doi.org/10.1145/2535838.2535852 (cit. on pp. 83, 85–87).

[AM13] Robert Atkey and Conor McBride. "Productive Coprogramming with Guarded Recursion". In: *Proceedings of the 18th ACM SIGPLAN International Conference on Functional Programming*. ICFP '13. Association for Computing Machinery, 2013, pp. 197–208. DOI: 10.1145/2500365.2500597. URL: https://doi.org/10.1145/2500365.2500597 (cit. on p. 77).

[And92] Jean-Marc Andreoli. "Logic Programming with Focusing Proofs in Linear Logic". In: *Journal of Logic and Computation* 2.3 (June 1992), pp. 297–347. ISSN: 0955-792X. DOI: 10.1093/logcom/2.3.297 (cit. on p. 61).

[AS12] Andreas Abel and Gabriel Scherer. "On Irrelevance and Algorithmic Equality in Predicative Type Theory". In: *Logical Methods in Computer Science* 8.1 (2012). DOI: 10.2168/LMCS-8(1:29)2012 (cit. on pp. 5, 62).

[Atk12] Robert Atkey. "Relational Parametricity for Higher Kinds". In: *Computer Science Logic (CSL'12) - 26th International Workshop/21st Annual Conference of the EACSL, CSL 2012, September 3-6, 2012, Fontainebleau, France*. 2012, pp. 46–61. DOI: 10.4230/LIPIcs.CSL.2012.46. URL: https://doi.org/10.4230/LIPIcs.CSL.2012.46 (cit. on pp. 83–85).

[Awo18] Steve Awodey. "Natural models of homotopy type theory". In: *Mathematical Structures in Computer Science* 28.2 (2018), pp. 241–286. ISSN: 09601295. DOI: 10.1017/S0960129516000268. eprint: 1406.3219 (cit. on pp. 21, 22, 24, 25, 27, 28, 31, 33, 59, 61).

[BCM15] Jean-Philippe Bernardy, Thierry Coquand, and Guilhem Moulin. "A Presheaf Model of Parametric Type Theory". In: *Electr. Notes Theor. Comput. Sci.* 319 (2015), pp. 67–82. DOI: 10.1016/j.entcs.2015.12.006. URL: https://doi.org/10.1016/j.entcs.2015.12.006 (cit. on pp. 83, 86).

[BGM17] Patrick Bahr, Hans Bugge Grathwohl, and Rasmus Ejlers Møgelberg. "The clocks are ticking: No more delays!" In: *2017 32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*. IEEE, 2017. DOI: 10.1109/LICS.2017.8005097. URL: http://www.itu.dk/people/mogel/papers/lics2017.pdf (cit. on pp. 5, 63, 77, 78).

[BGM19] Patrick Bahr, Christian Uldal Graulund, and Rasmus Ejlers Møgelberg. "Simply RaTT: A Fitch-style Modal Calculus for Reactive Programming Without Space Leaks". In: *Proc. ACM Program. Lang.* 3 (ICFP 2019), 109:1–109:27. ISSN: 2475-1421. DOI: 10.1145/3341713 (cit. on p. 64).

[Bir+12] Lars Birkedal, Rasmus Møgelberg, Jan Schwinghammer, and Kristian Støvring. "First steps in synthetic guarded domain theory: step-indexing in the topos of trees". In: *Logical Methods in Computer Science* 8.4 (2012). Ed. by Patrick Baillot. DOI: 10.2168/LMCS-8(4:1)2012 (cit. on pp. 5, 74, 101).

[Bir+20] Lars Birkedal, Ranald Clouston, Bassel Mannaa, Rasmus Ejlers Møgelberg, Andrew M. Pitts, and Bas Spitters. "Modal dependent type theory and dependent right adjoints". In: *Mathematical Structures in Computer Science* 30.2 (2020), pp. 118–138. DOI: 10.1017/S0960129519000197. eprint: 1804.05236 (cit. on pp. 5, 6, 19, 26, 33, 34, 37, 61, 63, 64, 67, 78, 98).

[Bir00] Lars Birkedal. "Developing Theories of Types and Computability via Realizability". In: *Electronic Notes in Theoretical Computer Science* 34 (2000) (cit. on p. 111).

[Biz+16] Aleš Bizjak, Hans Bugge Grathwohl, Ranald Clouston, Rasmus E. Møgelberg, and Lars Birkedal. "Guarded Dependent Type Theory with Coinductive Types". In: *Foundations of Software Science and Computation Structures*. Ed. by Bart Jacobs and Christof Löding. Springer Berlin Heidelberg, 2016, pp. 20–35 (cit. on pp. 5, 59, 75, 77, 78, 82, 100).

[BJP12] Jean-Philippe Bernardy, Patrik Jansson, and Ross Paterson. "Proofs for free - Parametricity for dependent types". In: *J. Funct. Program.* 22.2 (2012), pp. 107–152. DOI: 10.1017/S0956796812000056. URL: https://doi.org/10.1017/S0956796812000056 (cit. on p. 86).

[BP11] Mathieu Boespflug and Brigitte Pientka. "Multi-level Contextual Type Theory". In: *Electronic Proceedings in Theoretical Computer Science* 71 (Oct. 2011). DOI: 10.4204/EPTCS.71.3 (cit. on p. 61).

[BS15] Peter Brottveit Bock and Carsten Schürmann. "A Contextual Logical Framework". In: vol. 9450. Nov. 2015, pp. 402–417. ISBN: 978-3-662-48898-0. DOI: 10.1007/978-3-662-48899-7_28 (cit. on p. 61).

[Car78] John Cartmell. "Generalised Algebraic Theories and Contextual Categories". PhD thesis. University of Oxford, 1978 (cit. on pp. 11, 21, 39).

[CD14] Pierre Clairambault and Peter Dybjer. "The biequivalence of locally cartesian closed categories and Martin-Löf type theories". In: *Mathematical Structures in Computer Science* 24.6 (2014). DOI: 10.1017/S0960129513000881 (cit. on pp. 37, 71).

[Clo+15] Ranald Clouston, Aleš Bizjak, Hans Bugge Grathwohl, and Lars Birkedal. "Programming and Reasoning with Guarded Recursion for Coinductive Types". In: *Foundations of Software Science and Computation Structures*. Ed. by Andrew Pitts. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 407–421. ISBN: 978-3-662-46678-0 (cit. on p. 78).

[Clo18] Ranald Clouston. "Fitch-Style Modal Lambda Calculi". In: *Foundations of Software Science and Computation Structures*. Ed. by Christel Baier and Ugo Dal Lago. Springer International Publishing, 2018, pp. 258–275 (cit. on p. 63).

[Coq13] Thierry Coquand. *Presheaf model of type theory*. 2013. URL: http://www.cse.chalmers.se/~coquand/presheaf.pdf (cit. on pp. 11, 72).

[Coq18] Thierry Coquand. *Canonicity and normalization for Dependent Type Theory*. 2018. eprint: 1810.09367 (cit. on pp. 37, 47, 115).

[Coq96] Thierry Coquand. "An algorithm for type-checking dependent types". In: *Science of Computer Programming* 26.1 (1996), pp. 167–177. ISSN: 0167-6423. DOI: https://doi.org/10.1016/0167-6423(95)00021-6 (cit. on p. 11).

[dR15] Valeria de Paiva and Eike Ritter. "Fibrational Modal Type Theory". In: *Proceedings of the Tenth Workshop on Logical and Semantic Frameworks, with Applications (LSFA 2015)*. 2015. DOI: 10.1016/j.entcs.2016.06.010 (cit. on p. 61).

[Dyb96] Peter Dybjer. "Internal type theory". In: *Types for Proofs and Programs*. Ed. by Stefano Berardi and Mario Coppo. Berlin, Heidelberg: Springer Berlin Heidelberg, 1996, pp. 120–134. DOI: 10.1007/3-540-61780-9_66 (cit. on pp. 21, 25, 37).

[Fio12] Marcelo Fiore. *Discrete generalised polynomial functors*. Slides from talk given at ICALP 2012. 2012. URL: https://www.cl.cam.ac.uk/~mpf23/talks/ICALP2012.pdf (cit. on p. 22).

[Gab+16] Marco Gaboardi, Shin-ya Katsumata, Dominic Orchard, Flavien Breuvart, and Tarmo Uustalu. "Combining Effects and Coeffects via Grading". In: *Proceedings of the 21st ACM SIGPLAN International Conference on Functional Programming*. ICFP 2016. 2016. DOI: 10.1145/2951913.2951939 (cit. on p. 63).

[Gir93] Jean-Yves Girard. "On the unity of logic". In: *Annals of Pure and Applied Logic* 59.3 (1993), pp. 201–217. ISSN: 0168-0072. DOI: 10.1016/0168-0072(93)90093-S (cit. on p. 61).

[Gro+17] Jacob A Gross, Daniel R Licata, Max S New, Jennifer Paykin, Mitchell Riley, Michael Shulman, and Felix Wellen. "Differential Cohesive Type Theory (Extended Abstract)". In: *Extended abstracts for the Workshop "Homotopy Type Theory and Univalent Foundations"*. 2017. URL: https://hott-uf.github.io/2017/abstracts/cohesivett.pdf (cit. on p. 5).

[GSB19] Daniel Gratzer, Jonathan Sterling, and Lars Birkedal. "Implementing a Modal Dependent Type Theory". In: *Proc. ACM Program. Lang.* 3 (ICFP 2019). DOI: 10.1145/3341711 (cit. on pp. 5, 6, 11, 63, 64, 78, 97, 99, 115).

[Gua18] Adrien Guatto. "A Generalized Modality for Recursion". In: *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*. LICS '18. ACM, 2018. DOI: 10.1145/3209108.3209148 (cit. on p. 100).

[Has94a] Ryu Hasegawa. "Categorical Data Types in Parametric Polymorphism". In: *Mathematical Structures in Computer Science* 4.1 (1994), pp. 71–109. DOI: 10.1017/S0960129500000372. URL: https://doi.org/10.1017/S0960129500000372 (cit. on p. 84).

[Has94b] Ryu Hasegawa. "Relational limits in general polymorphism". In: *Publications of Research Institute for Mathematical Sciences* 30 (1994), pp. 535–576 (cit. on p. 84).

[Hof94] Martin Hofmann. "On the Interpretation of Type Theory in Locally Cartesian Closed Categories". In: *Computer Science Logic*. 1994 (cit. on p. 72).

[Hof97] Martin Hofmann. "Syntax and Semantics of Dependent Types". In: *Semantics and Logics of Computation*. Ed. by Andrew M. Pitts and P. Dybjer. Cambridge University Press, 1997, pp. 79–130. DOI: 10.1017/CBO9780511526619.004. URL: https://www.tcs.ifi.lmu.de/mitarbeiter/martin-hofmann/pdfs/syntaxandsemanticsof-dependenttypes.pdf (cit. on pp. 11, 25, 72, 85, 110).

[HS97] Martin Hofmann and Thomas Streicher. "Lifting Grothendieck Universes". Unpublished note. 1997. URL: https://www2.mathematik.tu-darmstadt.de/~streicher/NOTES/lift.pdf (cit. on pp. 72, 86).

[Jac99] B. Jacobs. *Categorical Logic and Type Theory*. Studies in Logic and the Foundations of Mathematics 141. North Holland, 1999 (cit. on pp. 109, 110).

[JY20] Niles Johnson and Donald Yau. *2-Dimensional Categories*. 2020. arXiv: 2002.06055 (cit. on pp. 49, 50).

[Kav17] G. A. Kavvos. "Dual-context calculi for modal logic". In: *2017 32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*. 2017, pp. 1–12. DOI: 10.1109/LICS.2017.8005089. arXiv: 1602.04860 (cit. on pp. 61, 98).

[Kav19] G. A. Kavvos. "Modalities, Cohesion, and Information Flow". In: *Proceedings of the ACM on Programming Languages* 3 (POPL 2019), 20:1–20:29. DOI: 10.1145/3290333 (cit. on p. 5).

[KHS19] Ambrus Kaposi, Simon Huber, and Christian Sattler. "Gluing for type theory". In: *Proceedings of the 4th International Conference on Formal Structures for Computation and Deduction (FSCD 2019)*. Ed. by Herman Geuvers. Vol. 131. 2019 (cit. on pp. 40, 46).

[KKA19] Ambrus Kaposi, András Kovács, and Thorsten Altenkirch. "Constructing Quotient Inductive-inductive Types". In: *Proc. ACM Program. Lang.* 3.POPL (Jan. 2019), 2:1–2:24. ISSN: 2475-1421. DOI: 10.1145/3290315 (cit. on pp. 11, 21, 39).

[KL12] Chantal Keller and Marc Lasson. "Parametricity in an Impredicative Sort". In: *Computer Science Logic (CSL'12) - 26th International Workshop/21st Annual Conference of the EACSL, CSL 2012, September 3-6, 2012, Fontainebleau, France*. 2012, pp. 381–395. DOI: 10.4230/LIPIcs.CSL.2012.381 (cit. on p. 86).

[Lac02] Stephen Lack. "Codescent objects and coherence". In: *Journal of Pure and Applied Algebra* 175.1 (2002). Special Volume celebrating the 70th birthday of Professor Max Kelly, pp. 223–241. ISSN: 0022-4049. DOI: 10.1016/S0022-4049(02)00136-6 (cit. on p. 49).

[Law07] F. William Lawvere. "Axiomatic cohesion". In: *Theory and Applications of Categories* 19.3 (2007), pp. 41–49. URL: http://www.tac.mta.ca/tac/volumes/19/3/19-03.pdf (cit. on p. 5).

[Lev12] P.B. Levy. *Call-By-Push-Value: A Functional/Imperative Synthesis*. Semantics Structures in Computation. Springer Netherlands, 2012. ISBN: 9789400709546 (cit. on p. 116).

[Lic+18] Daniel R. Licata, Ian Orton, Andrew M. Pitts, and Bas Spitters. "Internal Universes in Models of Homotopy Type Theory". In: *3rd International Conference on Formal Structures for Computation and Deduction (FSCD 2018)*. Ed. by H. Kirchner. Leibniz International Proceedings in Informatics (LIPIcs). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018, 22:1–22:17. DOI: 10.4230/LIPIcs.FSCD.2018.22. eprint: 1801.07664 (cit. on pp. 5, 104, 105).

[Lic19] Daniel R. Licata. Homotopy Type Theory Electronic Seminar Talks, 2019-03-21. 2019. URL: https://www.youtube.com/watch?v=-DP0wY2FBs4 (cit. on p. 5).

[LN15] J. Longley and D. Normann. *Higher-Order Computability*. Theory and Applications of Computability. Springer Berlin Heidelberg, 2015. ISBN: 9783662479926. URL: https://books.google.dk/books?id=fdXjCgAAQBAJ (cit. on p. 109).

[LS16] Daniel R. Licata and Michael Shulman. "Adjoint Logic with a 2-Category of Modes". In: *Logical Foundations of Computer Science*. Ed. by Sergei Artemov and Anil Nerode. Springer International Publishing, 2016, pp. 219–235. DOI: 10.1007/978-3-319-27683-0_16 (cit. on pp. 7, 64).

[LSR17] Daniel R. Licata, Michael Shulman, and Mitchell Riley. "A Fibrational Framework for Substructural and Modal Logics". In: *2nd International Conference on Formal Structures for Computation and Deduction (FSCD 2017)*. Ed. by Dale Miller. Vol. 84. Leibniz International Proceedings in Informatics (LIPIcs). Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2017, 25:1–25:22. DOI: 10.4230/LIPIcs.FSCD.2017.25 (cit. on pp. 6, 7, 64).

[Luo94] Zhaohui Luo. *Computation and Reasoning: A Type Theory for Computer Science*. New York, NY, USA: Oxford University Press, Inc., 1994. ISBN: 0-19-853835-9 (cit. on p. 110).

[LW15] 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* 16.3 (2015). DOI: 10.1145/2754931 (cit. on p. 72).

[MM18] Bassel Mannaa and Rasmus Ejlers Møgelberg. "The Clocks They Are Adjunctions Denotational Semantics for Clocked Type Theory". In: *3rd International Conference on Formal Structures for Computation and Deduction (FSCD 2018)*. Ed. by Hélène Kirchner. Vol. 108. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018, 23:1–23:17. ISBN: 978-3-95977-077-4. DOI: 10.4230/LIPIcs.FSCD.2018.23 (cit. on p. 78).

[MM92] Saunders Mac Lane and Ieke Moerdijk. *Sheaves in geometry and logic : a first introduction to topos theory*. Universitext. New York: Springer, 1992. ISBN: 0-387-97710-4 (cit. on p. 52).

[Mog91] Eugenio Moggi. "Notions of computation and monads". In: *Information and Computation* 93.1 (1991). Selections from 1989 IEEE Symposium on Logic in Computer Science, pp. 55–92. ISSN: 0890-5401. DOI: 10.1016/0890-5401(91)90052-4 (cit. on p. 116).

[Mou16] Guilhem Moulin. "Internalizing Parametricity". PhD thesis. Chalmers University of Technology, Gothenburg, Sweden, 2016. URL: http://publications.lib.chalmers.se/publication/235758-internalizing-parametricity (cit. on pp. 83, 86, 87).

[MP08] Conor McBride and Ross Paterson. "Applicative programming with effects". In: *Journal of Functional Programming* 18.1 (2008). DOI: 10.1017/S0956796807006326. URL: http://www.staff.city.ac.uk/~ross/papers/Applicative.pdf (cit. on p. 76).

[Nak00] H. Nakano. "A modality for recursion". In: *Proceedings Fifteenth Annual IEEE Symposium on Logic in Computer Science (Cat. No.99CB36332)*. IEEE Computer Society, 2000, pp. 255–266 (cit. on p. 100).

[ND18] Andreas Nuyts and Dominique Devriese. "Degrees of Relatedness: A Unified Framework for Parametricity, Irrelevance, Ad Hoc Polymorphism, Intersections, Unions and Algebra in Dependent Type Theory". In: *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*. LICS '18. ACM, 2018. DOI: 10.1145/3209108.3209119 (cit. on pp. 5, 62, 83, 90, 93, 94, 103).

[New18] Clive Newstead. "Algebraic models of dependent type theory". PhD thesis. Carnegie Mellon University, 2018. URL: https://sites.math.northwestern.edu/$%5Csim$newstead/thesis-clive-newstead.pdf (cit. on pp. 36–38, 52, 67).

[nLa15] nLab. *Pseudofunctors*. 2015. URL: https://ncatlab.org/nlab/show/pseudofunctors (cit. on p. 50).

[Nor19] Paige Randall North. "Type-theoretic weak factorization systems". In: (2019). arXiv: 1906.00259 [math.CT]. URL: https://arxiv.org/abs/1906.00259 (cit. on p. 115).

[NPP08] Aleksandar Nanevski, Frank Pfenning, and Brigitte Pientka. "Contextual Modal Type Theory". In: *ACM Transactions on Computational Logic* 9.3 (2008). DOI: 10.1145/1352582.1352591 (cit. on p. 61).

[Nuy18a] Andreas Nuyts. *Presheaf Models of Relational Modalities in Dependent Type Theory*. 2018. arXiv: 1805.08684 (cit. on pp. 34, 63, 67, 92, 96).

[Nuy18b] Andreas Nuyts. "Robust Notions of Contextual Fibrancy". In: *TYPES*. July 2018. URL: https://lirias.kuleuven.be/2809876 (cit. on p. 92).

[Nuy19] Andreas Nuyts. *Menkar*. https://github.com/anuyts/menkar. 2019 (cit. on pp. 65, 96, 115).

[NVD17] Andreas Nuyts, Andrea Vezzosi, and Dominique Devriese. "Parametric Quantifiers for Dependent Type Theory". In: *Proc. ACM Program. Lang.* 1.ICFP (2017). DOI: 10.1145/3110276 (cit. on pp. 5, 62, 83, 87–89, 91–93).

[OLE19] Dominic Orchard, Vilem-Benjamin Liepelt, and Harley Eades III. "Quantitative Program Reasoning with Graded Modal Types". In: *Proc. ACM Program. Lang.* (2019). DOI: 10.1145/3341714 (cit. on p. 63).

[Oos08] Jaap van Oosten. *Realizability: An Introduction to its Categorical Side.* ISSN. Elsevier Science, 2008. ISBN: 9780080560069 (cit. on p. 109).

[OP18] Ian Orton and Andrew M. Pitts. "Axioms for Modelling Cubical Type Theory in a Topos". In: *Logical Methods in Computer Science* 14.4 (2018). DOI: 10.23638/LMCS-14(4:23)2018. arXiv: 1712.04864 (cit. on p. 27).

[PD01] Frank Pfenning and Rowan Davies. "A Judgmental Reconstruction of Modal Logic". In: *Mathematical Structures in Computer Science* 11.4 (2001), pp. 511–540. DOI: 10.1017/S0960129501003322. URL: http://www.cs.cmu.edu/~fp/papers/mscs00.pdf (cit. on pp. 61, 97, 98).

[Pfe01] F. Pfenning. "Intensionality, extensionality, and proof irrelevance in modal type theory". In: *Proceedings 16th Annual IEEE Symposium on Logic in Computer Science.* IEEE, 2001, pp. 221–230. DOI: 10.1109/LICS.2001.932499. URL: https://www.cs.cmu.edu/~fp/papers/lics01.pdf (cit. on pp. 5, 62, 63).

[Pie+19] Brigitte Pientka, Andreas Abel, Francisco Ferreira, David Thibodeau, and Rébecca Zucchini. "A Type Theory for Defining Logics and Proofs". In: *Proceedings of the 34th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS).* 2019 (cit. on p. 61).

[Plo93] G. D. Plotkin. "Type theory and recursion". In: *Proceedings Eighth Annual IEEE Symposium on Logic in Computer Science.* 1993, pp. 374–. DOI: 10.1109/LICS.1993.287571 (cit. on p. 61).

[Pow89] A.J. Power. "A general coherence result". In: *Journal of Pure and Applied Algebra* 57.2 (1989), pp. 165–173. ISSN: 0022-4049. DOI: 10.1016/0022-4049(89)90113-8 (cit. on pp. 49, 50).

[PT00] Benjamin C. Pierce and David N. Turner. "Local type inference". In: *ACM Transactions Programming Language and Systems* 22.1 (2000), pp. 1–44 (cit. on p. 11).

[Ree09] Jason Reed. "A Judgmental Deconstruction of Modal Logic". 2009. URL: http://www.cs.cmu.edu/~jcreed/papers/jdml.pdf (cit. on p. 7).

[Rey83] John C. Reynolds. "Types, Abstraction and Parametric Polymorphism". In: *Information Processing 83, Proceedings of the IFIP 9th World Computer Congress, Paris, France, September 19-23, 1983.* 1983, pp. 513–523 (cit. on pp. 62, 83).

[Rie14] Emily Riehl. *Categorical homotopy theory.* Vol. 24. New Mathematical Monographs. Cambridge University Press, 2014 (cit. on p. 27).

[RR94] E. P. Robinson and Giuseppe Rosolini. "Reflexive Graphs and Parametric Polymorphism". In: *Proceedings of the Ninth Annual Symposium on Logic in Computer Science (LICS '94), Paris, France, July 4-7, 1994.* 1994, pp. 364–371. DOI: 10.1109/LICS.1994.316053. URL: https://doi.org/10.1109/LICS.1994.316053 (cit. on p. 84).

[Sch13] Urs Schreiber. "Differential cohomology in a cohesive infinity-topos". 2013 (cit. on p. 5).

[Shu18] Michael Shulman. "Brouwer's fixed-point theorem in real-cohesive homotopy type theory". In: *Mathematical Structures in Computer Science* 28.6 (2018), pp. 856–941. DOI: 10.1017/S0960129517000147. URL: https://doi.org/10.1017/S0960129517000147 (cit. on pp. 5, 61, 62, 78, 97, 98, 103, 106).

[SS12] Urs Schreiber and Michael Shulman. "Quantum Gauge Field Theory in Cohesive Homotopy Type Theory". In: *Proceedings 9th Workshop on Quantum Physics and Logic, QPL 2012, Brussels, Belgium, 10-12 October 2012*. Ed. by Ross Duncan and Prakash Panangaden. Vol. 158. EPTCS. 2012, pp. 109–126. DOI: 10.4204/EPTCS.158.8. URL: https://doi.org/10.4204/EPTCS.158.8 (cit. on pp. 5, 103).

[Ste19] Jonathan Sterling. "Algebraic Type Theory and Universe Hierarchies". 2019 (cit. on p. 12).

[Str18] Thomas Streicher. *Fibred Categories a la Jean Benabou*. 2018. eprint: 1801.02927 (cit. on pp. 49, 50).

[Tak01] Izumi Takeuti. *The Theory of Parametricity in Lambda Cube*. Tech. rep. 1217. Kyoto University, 2001 (cit. on p. 86).

[Tay99] Paul Taylor. *Practical Foundations of Mathematics*. Cambridge studies in advanced mathematics. Cambridge, New York (N. Y.), Melbourne: Cambridge University Press, 1999. ISBN: 0-521-63107-6 (cit. on pp. 11, 21).

[Uem19] Taichi Uemura. "A General Framework for the Semantics of Type Theory". In: (Apr. 2019). eprint: 1904.04097 (math.CT). URL: https://arxiv.org/abs/1904.04097 (cit. on pp. 37, 38).

[Vez18] Vezzosi, Andrea. *agda-flat*. 2018. URL: https://github.com/agda/agda/tree/flat (cit. on p. 62).

[Zwa19] Colin Zwanziger. "Natural Model Semantics for Comonadic and Adjoint Type Theory: Extended Abstract". In: *Preproceedings of Applied Category Theory Conference 2019*. 2019 (cit. on pp. 61, 62, 97).