Multimodal Dependent Type Theory

Daniel Gratzer
Aarhus University
gratzer@cs.au.dk

G. A. Kavvos
Aarhus University
alex.kavvos@cs.au.dk

Andreas Nuyts
imec-DistriNet, KU Leuven
andreas.nuyts@cs.kuleuven.be

Lars Birkedal
Aarhus University
birkedal@cs.au.dk

Abstract

We introduce MTT, a dependent type theory which supports multiple modalities. MTT is parametrized 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.

ACM Reference Format:
Daniel Gratzer, G. A. Kavvos, Andreas Nuyts, and Lars Birkedal. 2020. Multimodal Dependent Type Theory. In *Proceedings of ACM Conference (Conference'17)*. ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

In order to increase the expressivity of Martin-Löf Type Theory (MLTT) we often wish to extend it with new connectives, and in particular with unary type operators that we call *modalities* or *modal operators*. Some of these modal operators arise as shorthands, while others are introduced as a device for expressing structure that appears in particular models. Whereas the former class of modalities are internally definable [58], the latter often require extensive modifications to the basic structure of type-theoretic judgments. In some cases we are even able to prove that these changes are necessary, by showing that the modality in question cannot be expressed internally: see e.g. the 'no-go' theorems by Shulman [63, §4.1] and Licata et al. [38]. This paper is concerned with the development a systematic approach to the formulation of type theories with multiple modalities.

The addition of a modality to a dependent type theory is a non-trivial exercise. Modal operators often interact with the context of a type or term in a complicated way, and naïve approaches lead to undesirable interplay with other type formers and substitution. However, the consequent gain in expressivity is substantial, and so it is well worth the effort. For example, modalities have been used to express guarded recursive definitions [9, 14, 15, 30], parametric quantification [50, 51], proof irrelevance [3, 50, 53], and to define operations on which only exist globally and may be false in an arbitrary context [38]. There has also been concerted effort towards the development of a dependent type theory corresponding to Lawvere's *axiomatic cohesion* [37], which has many interesting applications [29, 36, 60, 61, 63].

Despite this recent flurry of developments, a unifying account of modal dependent type theory has yet to emerge. Faced with a new modal situation, a type theorist must handcraft 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, so that one cannot share the burden of proof even between 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 MLTT$_{🔒}$ [27] to a normalization proof for the modal dependent type theory of Birkedal et al. [13], 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.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

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. [13] and MLTT🔒 [27] respectively, so that our proof of canonicity applies to both.

In fact, we take things one step further: MTT is not just multimodal, but also *multimode*. That is, each judgment of MTT can be construed as existing in a particular *mode*. All modes have some things in common—e.g. there will be dependent sums in each—but some might possess distinguishing features. From a semantic point of view, different modes correspond to different context categories. In this light, modalities intuitively correspond to *functors* between those categories: in fact, they will be structures slightly weaker than *dependent right adjoints* (DRAs) [13].

Mode theories At a high level, MTT can be thought of as a machine that converts a concrete description of modes and modalities into a type theory. This description, which is often called a *mode theory*, is given in the form of a *small strict 2-category* [39, 40, 57]. A mode theory gives rise to the following correspondence:

$$\begin{aligned}\text{object} &\sim \text{mode}\\ \text{morphism} &\sim \text{modality}\\ \text{2-cell} &\sim \text{natural map between modalities}\end{aligned}$$

The equations between morphisms and between 2-cells in a mode theory can be used to precisely specify the interactions we want between different modalities. We will illustrate this point with an example.

Instantiating MTT Suppose we have a mode theory $\mathcal{M}$ with a single object m, a single generating morphism $\mu : m \to m$, and no non-trivial 2-cells. Equipping MTT with $\mathcal{M}$ produces a type theory with a single modal type constructor, $\langle \mu \mid - \rangle$. This is the simplest non-trivial setting, and we can prove very little about it without additional 2-cells.

If we add a 2-cell $\epsilon : \mu \Rightarrow 1$ to $\mathcal{M}$, we can define a function

$$\mathsf{extract}_A : \langle \mu \mid A \rangle \to A$$

inside the type theory. If we also add a 2-cell $\delta : \mu \Rightarrow \mu \circ \mu$ then we can also define

$$\mathsf{duplicate}_A : \langle \mu \mid A \rangle \to \langle \mu \mid \langle \mu \mid A \rangle \rangle$$

Furthermore, we can control the precise interaction between $\mathsf{duplicate}_A$ and $\mathsf{extract}_A$ by adding more equations that relate ϵ and δ. For example, we may ask that $\mathcal{M}$ be the *walking comonad* [59] which leads to a type theory with a dependent S4-like modality [24, 53, 54, 63]. We can be even more specific, e.g. by asking that (μ, ϵ, δ) be *idempotent*.

Thus, a morphism $\mu : n \to m$ introduces a modality $\langle \mu \mid - \rangle$, and a 2-cell $\alpha : \mu \Rightarrow \nu$ of $\mathcal{M}$ allows the definition of a function of type $\langle \mu \mid A \rangle \to \langle \nu \mid A \rangle$ @ m.

Relation to other modal type theories Most work on modal type theories still defies classification. However, we can informatively position MTT with respect to two qualitative criteria, viz. usability and generality.

Much of the prior work on modal type theory has focused on bolting a specific modality onto a type theory. The benefit of this approach is that the syntax can be designed to be as convenient as possible for the application at hand. For example, spatial/cohesive type theory [63] features two modalities, $\flat$ and $\sharp$, and is presented in a dual-context style. This judgmental structure, however, is applicable only because of the particular properties of $\flat$ and $\sharp$. Nevertheless, the numerous pen-and-paper proofs in *op. cit.* demonstrate that the resulting system is easy to use.

At the other end of the spectrum, the framework of Licata-Shulman-Riley (LSR) [40] comprises an extremely general toolkit for simply-typed, substructural modal type theory. Its dependent generalization, which is currently under development, is able to handle a very large class of modalities. However, this generality comes at a price: its syntax is complex and unwieldy, even in the simply-typed case.

MTT attempts to strike a delicate balance between those two extremes. By avoiding substructural settings and some kinds of modalities we obtain a noticeably simpler apparatus. These restrictions imply that, unlike LSR, we do not need to annotate our term formers with delayed substitutions, and that our system straightforwardly extends to dependent types. We also show that MTT can be used for many important examples, and that it is simple enough to be used in pen-and-paper calculations.

Contributions In summary, we make the following contributions:

- We introduce MTT, a general type theory for multiple modes and multiple interacting modalities.
- We define its semantics, which constitute a category of models.
- We prove that MTT satisfies *canonicity*, an important metatheoretic property, through a modern *gluing* argument [5, 23, 33, 62].

- We instantiate MTT with various mode theories, and show its value in reasoning about guarded recursion [15], degrees of relatedness [50], and other modal situations.

For want of space we omit many details and proofs, which can be found in the accompanying technical report.

2 The Syntax of MTT

We now present the syntax of MTT. For the rest of this paper we fix a mode theory $\mathcal{M}$, and use m, n, o to stand for modes, μ, ν, τ for modalities, and α, β, γ for 2-cells.

In broad terms, MTT consists of a collection of type theories, one for each mode $m \in \mathcal{M}$. These type theories will eventually appear in one another, but only as spectres under a modality. We thus begin by describing the individual type theories at each mode, and only then discuss how modalities can be used to relate them.

2.1 The Type Theory at Each Mode

Each mode in MTT is inhabited by a standard Martin-Löf Type Theory (MLTT), and accordingly includes the usual judgments. For example, we have the judgment $\Gamma\ \mathsf{ctx} @ m$ which states that Γ is a well-formed context *in that particular mode m*. There are likewise judgments for types, terms, and substitutions at each mode.

In lieu of an exhaustive list of rules, we show only the important ones in Fig. 1. Briefly, each mode contains ordinary intensional type theory with dependent sums, dependent products, intensional identity types, booleans, and one universe. Both sums and products satisfy an η rule.

Universes à la Coquand There are several ways to present universes in type theory [31, §2.1.6] [41, 52]. We use the approach of Coquand [22], which is close to Tarski-style universes. However, instead of inductively defining *codes* that represent particular types, Coquand-style universes come with an *explicit isomorphism* between types and terms of the universe U.

If this isomorphism were to cover all types then *Girard's paradox* [21] would apply, so we must restrict it to *small types*. This, in turn, forces us to stratify our types into *small* and *large*. The judgment $\Gamma \vdash A\ \mathsf{type}_0 @ m$ states that A is a small type, and $\Gamma \vdash A\ \mathsf{type}_1 @ m$ that it is large. The universe itself must be a large type, but otherwise both levels are closed under all other connectives. Finally, we introduce an operator that *lifts* a small type to a large one:

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

The lifting operation commutes definitionally with all the connectives, e.g. $\Uparrow(A \to B) = \Uparrow A \to \Uparrow B$. We will use large types for the most part: only they will be allowed in contexts, and the judgment $\Gamma \vdash M : A @ m$ will presuppose that A is large. As we will not have terms at small types, we will not need the term lifting operations used by Coquand [22] and Sterling [64].

Following this stratification, we may introduce operations that exhibit the isomorphism:

$$\frac{\Gamma \vdash M : \mathsf{U} @ m}{\Gamma \vdash \mathsf{El}(M)\ \mathsf{type}_0 @ m} \qquad \frac{\Gamma \vdash A\ \mathsf{type}_0 @ m}{\Gamma \vdash \mathsf{Code}(A) : \mathsf{U} @ m}$$

along with the equations $\mathsf{Code}(\mathsf{El}(M)) = M$ and $\mathsf{El}(\mathsf{Code}(A)) = A$. The advantage of universes à la Coquand is now evident: rather than having to introduce Tarski-style codes, we now find that they are *definable*. For example, assuming $M : \mathsf{U}$ and $x : \mathsf{El}(M) \vdash N : \mathsf{U}$, we let

$$(x : M) \mathrel{\widehat{\to}} N \triangleq \mathsf{Code}((x : \mathsf{El}(M)) \to \mathsf{El}(N)) : \mathsf{U}$$

We can then calculate that

$$\begin{aligned} \mathsf{El}((x : M) \mathrel{\widehat{\to}} N) &= \mathsf{El}(\mathsf{Code}((x : \mathsf{El}(M)) \to \mathsf{El}(N))) \\ &= (x : \mathsf{El}(M)) \to \mathsf{El}(N) \end{aligned}$$

We will often suppress $\mathsf{El}(-)$ and $\Uparrow -$, and simply use $M : \mathsf{U}$ as a type.

2.2 Introducing a Modality

Having sketched the basic type theory inhabiting each mode, we now show how these type theories interact.

Suppose $\mathcal{M}$ contains a modality $\mu : n \to m$. We would like to think of μ as a 'map' from mode n to mode m. Then, for each $\vdash A\ \mathsf{type} @ n$ we would like a type $\vdash \langle \mu \mid A \rangle\ \mathsf{type} @ m$. On the level of terms we would similarly like for each $\vdash M : A @ n$ an induced term $\vdash \mathsf{mod}_\mu(M) : \langle \mu \mid A \rangle @ m$.

These constructs would be entirely satisfactory, were it not for the presence of *open terms*. To illustrate the problem, suppose we have a type $\Gamma \vdash A\ \mathsf{type} @ n$. We would hope that the corresponding modal type would live in the same context, i.e. that $\Gamma \vdash \langle \mu \mid A \rangle\ \mathsf{type} @ m$. However, this is not possible, as Γ is only a context at mode n, and cannot be carried over verbatim to mode m. Hence, the only pragmatic option is to introduce an operation that allows a context to cross over to another mode.

Forming a modal type There are several different proposed solutions to this problem in the literature [e.g. 19, 54]. In the case of MTT we will use a *Fitch-style* discipline [9, 13, 27]: we will require that μ induce an operation on contexts in the *reverse* direction, which we will denote by a *lock*:

$$\textsc{cx/lock}\ \frac{\Gamma\ \mathsf{ctx} @ m}{\Gamma, \text{🔒}_\mu\ \mathsf{ctx} @ n}$$

Intuitively, 🔒_μ behaves like a left adjoint to $\langle \mu \mid - \rangle$. However, $\langle \mu \mid - \rangle$ acts on types while $-, \text{🔒}_\mu$ acts on contexts, so this cannot be an adjunction. Birkedal et al. [13] call this situation a *dependent right adjoint* (DRA). A DRA essentially consists of a type former $\mathbf{R}$ and a context operation $\mathbf{L}$ such that

$$\{N \mid \mathbf{L}(\Gamma) \vdash N : A\} \cong \{M \mid \Gamma \vdash M : \mathbf{R}(A)\} \qquad (\dagger)$$

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

$$\frac{}{\Gamma \vdash \mathsf{U}\ \mathsf{type}_1 \mathbin{@} m} \qquad \frac{\Gamma\ \mathsf{ctx} \mathbin{@} m}{\Gamma \vdash \mathbb{B}\ \mathsf{type}_\ell \mathbin{@} m} \qquad \frac{\ell \leq \ell' \qquad \Gamma\ \mathsf{ctx} \mathbin{@} m \qquad \Gamma \vdash A\ \mathsf{type}_\ell \mathbin{@} m}{\Gamma \vdash \Uparrow A\ \mathsf{type}_{\ell'} \mathbin{@} m}$$

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

Figure 1. Selected mode-local rules.

See Birkedal et al. [13] for a formal definition.

Just as with DRAs, the MTT formation and introduction rules for modal types effectively *transpose* types and terms across this adjunction:

$$\textsc{tp/modal}\ \frac{\Gamma, \text{🔒}_\mu \vdash A\ \mathsf{type}_\ell \mathbin{@} n}{\Gamma \vdash \langle \mu \mid A \rangle\ \mathsf{type}_\ell \mathbin{@} m} \qquad \textsc{tm/modal-intro}\ \frac{\Gamma, \text{🔒}_\mu \vdash M : A \mathbin{@} n}{\Gamma \vdash \mathsf{mod}_\mu(M) : \langle \mu \mid A \rangle \mathbin{@} m}$$

It remains to show how to eliminate modal types. Previous work on Fitch-style calculi [13, 27] has employed elimination rules which essentially invert the introduction rule TM/MODAL-INTRO. Such rules *remove* one or more locks from the context during type-checking, and sometimes even trim a part of it. For example, a rule of this sort would be

$$\frac{\text{🔒}_\mu \notin \Gamma' \qquad \Gamma \vdash M : \langle \mu \mid A \rangle \mathbin{@} m}{\Gamma, \text{🔒}_\mu, \Gamma' \vdash \mathsf{open}(M) : A \mathbin{@} n}$$

However, this kind of rule tends to be unruly, and requires delicate work to prove even basic results, such as the admissibility of substitution: see the technical report by Gratzer et al. [28] for a particularly laborious case. The results in *op. cit.* could not possibly reuse any of the work of Birkedal et al. [13], as a small change in the syntax leads to many subtle changes in the metatheory. Consequently, it seems unlikely that one could adapt this approach to a modality-agnostic setting like ours.

We will use a different technique, which is reminiscent of dual-context calculi [35]. First, we will let the variable rule control the use of modal variables. Then, we will take a 'modal cut' rule, which will allow the substitution of modal terms for modal variables, to be our modal elimination rule.

Accessing a modal variable The behavior of modal types can often be clarified by asking a simple question: when can we use $x : \langle \mu \mid A \rangle$ to construct a term of type A? In previous Fitch-style calculi we would use the modal elimination rule to reduce the goal to $\langle \mu \mid A \rangle$, and then—*had the modal elimination rule not eliminated x from the context*—we would simply use the variable. We may thus write down a term of type A using a variable $x : \langle \mu \mid A \rangle$ only when our context has the appropriate structure, and the final arbiter of that is the modal elimination rule.

MTT turns this idea on its head: rather than handing control over to the modal elimination rule, we delegate this decision to the variable rule itself. In order to ascertain whether we can use a variable in our calculus, the variable rule examines *the locks to the right of the variable.* The rule of thumb is this: we should always be able to access $\langle \mu \mid A \rangle$ behind 🔒_μ. Carrying the $-, \text{🔒}_\mu \dashv \langle \mu \mid - \rangle$ analogy further, we see that the simplest judgment that fits this, namely $\Gamma, x : \langle \mu \mid A \rangle, \text{🔒}_\mu \vdash x : A \mathbin{@} n$, corresponds to the *counit.*

To correctly formulate the variable rule, we will require one more idea: following modal type theories based on *left division* [1, 2, 50, 51, 53], every variable in the context will be annotated with a modality, $x : (\mu \mid A)$. Intuitively a variable $x : (\mu \mid A)$ is the same as a variable $x : \langle \mu \mid A \rangle$, but the annotations are part of the structure of a context while $\langle \mu \mid A \rangle$ is a type. This small circumlocution will ensure that the variable rule respects substitution.

The most general form of the variable rule will be able to handle the interaction of modalities, so we present it in stages. A first 'counit-like' approximation is then

$$\textsc{tm/var/counit}\ \frac{\text{🔒} \notin \Gamma_1 \qquad \Gamma_0, \text{🔒}_\mu \vdash A\ \mathsf{type}_1 \mathbin{@} n}{\Gamma_0, x : (\mu \mid A), \text{🔒}_\mu, \Gamma_1 \vdash x : A \mathbin{@} m}$$

The first premise requires that no further locks occur in Γ_1.

Context extension The switch to modality-annotated declarations $x : (\mu \mid A)$ also requires us to revise the context extension rule. The revised version, CX/EXTEND, closely follows the formation rule for $\langle \mu \mid - \rangle$: if $\Gamma, \text{🔒}_\mu \vdash A\ \mathsf{type}_1 \mathbin{@} n$ is a type in the locked context Γ, then we may extend the context Γ to include a declaration $x : (\mu \mid A)$, so that x stands for a term of type A *under the modality μ*.

The elimination rule The difference between a modal type $\langle \mu \mid A \rangle$ and an annotated declaration $x : (\mu \mid A)$ in the context is navigated by the modal elimination rule. In brief, its role is to enable the substitution of a term of the former type for a variable with the latter declaration. The full rule is complex, so in this section we will only discuss the case of a single modality, $\mu : n \longrightarrow m$. The rule for this μ is

$$\textsc{tm/modal-elim/single-modality}\quad \frac{\Gamma \vdash M_0 : \langle \mu \mid A \rangle \mathbin{@} m \qquad \Gamma, x : (1 \mid \langle \mu \mid A \rangle) \vdash B \ \mathsf{type}_1 \mathbin{@} m \qquad \Gamma, y : (\mu \mid A) \vdash M_1 : B[\mathsf{mod}_\mu(y)/x] \mathbin{@} m}{\Gamma \vdash \mathsf{let}\ \mathsf{mod}_\mu(y) \leftarrow M_0\ \mathsf{in}\ M_1 : B[M_0/x] \mathbin{@} m}$$

Forgetting dependence for a moment, this rule is close to the dual context style [35, 54]: if we think of annotations as separating the context into multiple zones, then $y : (\mu \mid A)$ clearly belongs to the 'modal' part.

In the dependent case we also need a motive $\Gamma, x : (1 \mid \langle \mu \mid A \rangle) \vdash B\ \mathsf{type}_1 \mathbin{@} m$, which depends on a variable of modal type, but under the identity modality 1. This premise is then fulfilled by M_0 in the conclusion. In a sense, this rule permits a form of *modal induction*: every variable $x : (1 \mid \langle \mu \mid A \rangle)$ can be assumed to be of the form $\mathsf{mod}_\mu(y)$ for some $y : (\mu \mid A)$. This kind of rule has appeared before in dependent modal type theory, mainly in the work of Shulman [63].

In the type theory of Birkedal et al. [13] modalities are taken to be dependent right adjoints, with terms witnessing Eq. (†). This isomorphism can encode tm/modal-elim/single-modality, but tm/modal-elim/single-modality cannot encode Eq. (†). As a result, modalities in MTT are weaker than DRAs.

2.3 Multiple Modalities

Thus far we have only considered a single modality. In this section we discuss the small changes that are needed to enable MTT to support multiple interacting modalities. The final version of the modal rules is given in Fig. 2.

Multimodal locks We have so far only used the operation $-, \text{🔒}_\mu$ on contexts for the single modality $\mu : n \longrightarrow m$. This operation should also work for any modality with the same rule cx/lock, hence inducing an action of locks on contexts that is contravariant with respect to the mode. The only question, then, is how these locks should interact. This is where the mode theory comes in: locks should be *functorial*, so that $\nu : o \longrightarrow n$, $\mu : n \longrightarrow m$, and $\Gamma\ \mathsf{ctx} \mathbin{@} m$ imply $\Gamma, \text{🔒}_\mu, \text{🔒}_\nu = \Gamma, \text{🔒}_{\mu \circ \nu}\ \mathsf{ctx} \mathbin{@} o$. We additionally ask that the identity modality $1 : m \longrightarrow m$ at each mode has a trivial, invisible action on contexts, i.e. $\Gamma, \text{🔒}_1 = \Gamma$.

These two actions, which are encoded by cx/compose and cx/id, ensure that 🔒 *is a contravariant functor on* $\mathcal{M}$, mapping each mode m to the category of contexts $\Gamma\ \mathsf{ctx} \mathbin{@} m$. The contravariance originates from the fact that $\mathcal{M}$ is a specification of the behavior of the modalities $\langle \mu \mid - \rangle$, so that their left-adjoint-like counterparts $-, \text{🔒}_\mu$ act with the opposite variance.

The full variable rule We have seen that 🔒 induces a functor from $\mathcal{M}$ to categories of contexts, but we have not yet used the 2-cells of $\mathcal{M}$. In short, a 2-cell $\alpha : \mu \Rightarrow \nu$ contravariantly induces a substitution from $\Gamma, \text{🔒}_\nu$ to $\Gamma, \text{🔒}_\mu$. We will discuss this further in Section 4, but for now we only mention that this gives rise to an *admissible operation on types*: for each 2-cell we obtain an operation $(-)^\alpha$ such that $\Gamma, \text{🔒}_\mu \vdash A\ \mathsf{type} \mathbin{@} m$ implies $\Gamma, \text{🔒}_\nu \vdash A^\alpha\ \mathsf{type} \mathbin{@} m$.

In order to prove the admissibility of this operation we need a more expressive variable rule that builds in the action of 2-cells. The first iteration (tm/var/counit) required that the lock and the variable annotation were an exact match. We relax this requirement by allowing for a mediating 2-cell:

$$\textsc{tm/var/combined}\quad \frac{\mu, \nu : n \longrightarrow m \qquad \alpha : \mu \Rightarrow \nu}{\Gamma, x : (\mu \mid A), \text{🔒}_\nu \vdash x^\alpha : A^\alpha \mathbin{@} n}$$

The superscript in x^α is now part of the syntax: each variable must be annotated with the 2-cell, though we will still write x to mean x^{1_μ}. The final form of the variable rule, which appears as tm/var in Fig. 2, is only a slight generalization which allows the variable to occur at positions other than the very front of the context. In fact, tm/var can be reduced to tm/var/combined by using weakening to remove variables to the right of x, and then invoking functoriality to fuse all the locks to the right of x into a single one with modality $\mathsf{locks}(\Gamma_1)$.

The full elimination rule Recall that the elimination rule for a single modality (tm/modal-elim/single-modality) allowed us to plug a term of type $\langle \mu \mid A \rangle$ for an assumption $x : (\mu \mid A)$. Some additional generality is needed to cover the case where the motive $x : (\nu \mid \langle \mu \mid A \rangle) \vdash B\ \mathsf{type} \mathbin{@} m$ depends on x under a modality $\nu \neq 1$. This is where the composition of modalities in $\mathcal{M}$ comes in handy: our new rule will use it to absorb ν by replacing the assumption $x : (\nu \mid \langle \mu \mid A \rangle)$ with $x : (\nu \circ \mu \mid A)$. The new rule, tm/modal-elim, is given in Fig. 2. The simpler rule may be recovered by setting $\nu \triangleq 1$.

Modal dependent products In the technical report we have supplemented MTT with a primitive *modal dependent product* type, $(x : (\mu \mid A)) \to B$, which bundles together $\langle \mu \mid - \rangle$ and the ordinary product. If we ignore η-equality, $(x : (\mu \mid A)) \to B$ can be defined as $(x_0 : \langle \mu \mid A \rangle) \to (\mathsf{let}\ \mathsf{mod}_\mu(x) \leftarrow x_0\ \mathsf{in}\ B)$. This modal $\prod$-type is convenient for programming but it is not essential, so we defer further discussion to the technical report.

3 Programming with Modalities

In this section we show how MTT can be used to program and reason with modalities. We develop a toolkit of modal combinators, which we then use in Section 3.2 to show how MTT can be effortlessly used to present an idempotent comonad.

3.1 Modal Combinators

We first show how each 2-cell $\alpha : \mu \Rightarrow \nu$ with $\mu, \nu : n \longrightarrow m$ induces a natural transformation $\langle \mu \mid - \rangle \to \langle \nu \mid - \rangle$. Given $\Gamma, \text{🔒}_\mu \vdash A\ \mathsf{type}_1 \mathbin{@} m$, we define

$$\begin{aligned}
&\mathbf{coe}[\alpha : \mu \Rightarrow \nu](-) \ :\ \langle \mu \mid A \rangle \to \langle \nu \mid A^\alpha \rangle \\
&\mathbf{coe}[\alpha : \mu \Rightarrow \nu](x) \triangleq \mathsf{let}\ \mathsf{mod}_\mu(x_0) \leftarrow x\ \mathsf{in}\ \mathsf{mod}_\nu(x_0^\alpha)
\end{aligned}$$

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

$$\textsc{cx/lock}\quad \frac{\mu : n \longrightarrow m \qquad \Gamma\ \mathsf{ctx}\ @\ m}{\Gamma, \text{🔒}_{\mu}\ \mathsf{ctx}\ @\ n} \qquad\qquad \textsc{cx/extend}\quad \frac{\mu : n \longrightarrow m \qquad \Gamma\ \mathsf{ctx}\ @\ m \qquad \Gamma, \text{🔒}_{\mu} \vdash A\ \mathsf{type}_1\ @\ n}{\Gamma, x : (\mu \mid A)\ \mathsf{ctx}\ @\ m}$$

$$\textsc{cx/id}\quad \frac{\Gamma\ \mathsf{ctx}\ @\ m}{\Gamma = \Gamma, \text{🔒}_{1}\ \mathsf{ctx}\ @\ m} \qquad\qquad \textsc{cx/compose}\quad \frac{\nu : o \longrightarrow n \qquad \mu : n \longrightarrow m \qquad \Gamma\ \mathsf{ctx}\ @\ m}{\Gamma, \text{🔒}_{\mu}, \text{🔒}_{\nu} = \Gamma, \text{🔒}_{\mu \circ \nu}\ \mathsf{ctx}\ @\ o}$$

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

$$\textsc{tp/modal}\quad \frac{\Gamma, \text{🔒}_{\mu} \vdash A\ \mathsf{type}_{\ell}\ @\ n}{\Gamma \vdash \langle \mu \mid A \rangle\ \mathsf{type}_{\ell}\ @\ m} \qquad \textsc{tm/var}\quad \frac{\nu : m \longrightarrow n \qquad \alpha : \nu \Rightarrow \mathsf{locks}(\Gamma_1)}{\Gamma_0, x : (\nu \mid A), \Gamma_1 \vdash x^{\alpha} : A^{\alpha}\ @\ m} \qquad \textsc{tm/modal-intro}\quad \frac{\Gamma, \text{🔒}_{\mu} \vdash M : A\ @\ n}{\Gamma \vdash \mathsf{mod}_{\mu}(M) : \langle \mu \mid A \rangle\ @\ m}$$

$$\textsc{tm/modal-elim}\quad \frac{\begin{array}{c}\nu : m \longrightarrow o \qquad \mu : n \longrightarrow m \qquad \Gamma, x : (\nu \mid \langle \mu \mid A \rangle) \vdash B\ \mathsf{type}_1\ @\ o \\ \Gamma, \text{🔒}_{\nu} \vdash M_0 : \langle \mu \mid A \rangle\ @\ m \qquad \Gamma, x : (\nu \circ \mu \mid A) \vdash M_1 : B[\mathsf{mod}_{\mu}(x)/x]\ @\ o\end{array}}{\Gamma \vdash \mathsf{let}_{\nu}\ \mathsf{mod}_{\mu}(x) \leftarrow M_0\ \mathsf{in}\ M_1 : B[M_0/x]\ @\ o}$$

$$\textsc{tm/modal-beta}\quad \frac{\begin{array}{c}\nu : m \longrightarrow o \qquad \mu : n \longrightarrow m \qquad \Gamma, x : (\nu \mid \langle \mu \mid A \rangle) \vdash B\ \mathsf{type}_1\ @\ o \\ \Gamma, \text{🔒}_{\nu \circ \mu} \vdash M_0 : A\ @\ n \qquad \Gamma, x : (\nu \circ \mu \mid A) \vdash M_1 : B[\mathsf{mod}_{\mu}(x)/x]\ @\ o\end{array}}{\Gamma \vdash \mathsf{let}_{\nu}\ \mathsf{mod}_{\mu}(x) \leftarrow \mathsf{mod}_{\mu}(M_0)\ \mathsf{in}\ M_1 = M_1[M_0/x] : B[\mathsf{mod}_{\mu}(M_0)/x]\ @\ o}$$

$$\boxed{\mathsf{locks}(\Gamma)}$$

$$\mathsf{locks}(\cdot) = 1 \qquad \mathsf{locks}(\Gamma, x : (\mu \mid A)) = \mathsf{locks}(\Gamma) \qquad \mathsf{locks}(\Gamma, \text{🔒}_{\mu}) = \mathsf{locks}(\Gamma) \circ \mu$$

Figure 2. Selected modal rules.

With this operation, we have completed the correspondence from Section 1: objects of $\mathcal{M}$ correspond to modes, morphisms to modalities, and 2-cells to coercions.

We can also show that the assignment $\mu \mapsto \langle \mu \mid - \rangle$ is, in some sense, *functorial*. Unlike the action of locks, this functoriality is not definitional, but only a type-theoretic *equivalence* [66, §4]. Fixing $\Gamma, \text{🔒}_{\mu \circ \nu} \vdash A\ \mathsf{type}_1\ @\ m$, let

$$\begin{aligned} &\mathbf{comp}_{\mu,\nu} : \langle \mu \mid \langle \nu \mid A \rangle \rangle \to \langle \mu \circ \nu \mid A \rangle \\ &\mathbf{comp}_{\mu,\nu}(x) \triangleq \mathsf{let}\ \ \mathsf{mod}_{\mu}(x_0) \leftarrow x\ \mathsf{in} \\ &\qquad\qquad\qquad \mathsf{let}_{\mu}\ \mathsf{mod}_{\nu}(x_1) \leftarrow x_0\ \mathsf{in} \\ &\qquad\qquad\qquad \mathsf{mod}_{\mu \circ \nu}(x_1) \end{aligned}$$

$$\begin{aligned} &\mathbf{comp}^{-1}_{\mu,\nu} : \langle \mu \circ \nu \mid A \rangle \to \langle \mu \mid \langle \nu \mid A \rangle \rangle \\ &\mathbf{comp}^{-1}_{\mu,\nu}(x) \triangleq \mathsf{let}\ \mathsf{mod}_{\mu \circ \nu}(x_0) \leftarrow x\ \mathsf{in}\ \mathsf{mod}_{\mu}(\mathsf{mod}_{\nu}(x_0)) \end{aligned}$$

We elide the 2-cell annotations on variables, as they are all identities (i.e. we only need tm/var/counit). Even in this small example the context equations that involve locks are essential: for $\langle \mu \mid \langle \nu \mid A \rangle \rangle$ to be a valid type we need that $\Gamma, \text{🔒}_{\mu}, \text{🔒}_{\nu} = \Gamma, \text{🔒}_{\mu \circ \nu}$, which is ensured by cx/compose. Additionally, observe that $\mathbf{comp}_{\mu,\nu}$ relies crucially on the multimodal elimination rule tm/modal-elim: we must pattern-match on x_0, which is under μ in the context.

These combinators are only propositionally inverse. In one direction, the proof is

$$\begin{aligned} \_ &: (x : \langle \mu \mid \langle \nu \mid A \rangle \rangle) \to \mathsf{Id}_{\langle \mu \mid \langle \nu \mid A \rangle \rangle}(x, \mathbf{comp}^{-1}_{\mu,\nu}(\mathbf{comp}_{\mu,\nu}(x))) \\ \_ &\triangleq \lambda x.\ \mathsf{let}\ \mathsf{mod}_{\mu}(x_0) \leftarrow x\ \mathsf{in}\ \mathsf{let}_{\mu}\ \mathsf{mod}_{\nu}(x_1) \leftarrow x_0\ \mathsf{in} \\ &\quad\ \mathsf{refl}(\mathsf{mod}_{\mu}(\mathsf{mod}_{\nu}(x))) \end{aligned}$$

This is a typical example of reasoning about modalities: we use the modal elimination rule to induct on a modally-typed term. This reduces it to a term of the form $\mathsf{mod}(-)$, and the result follows definitionally. It is equally easy to construct an equivalence $\langle 1 \mid A \rangle \simeq A$.

As a final example, we will show that each modal type satisfies *axiom K*, a central axiom of Kripke-style modal logics. This combinator will be immediately recognizable to functional programmers as the term that shows that $\langle \mu \mid - \rangle$

is an *applicative functor* [44].

$$\begin{aligned} - \circledast_\mu - \; &: \; \langle \mu \mid A \to B \rangle \to \langle \mu \mid A \rangle \to \langle \mu \mid B \rangle \\ f \circledast_\mu a \; &\triangleq \; \mathsf{let}\ \mathsf{mod}_\mu(f_0) \leftarrow f \ \mathsf{in} \\ &\quad\;\; \mathsf{let}\ \mathsf{mod}_\mu(a_0) \leftarrow a \ \mathsf{in} \\ &\quad\;\; \mathsf{mod}_\mu(f_0(a_0)) \end{aligned}$$

We can also define a stronger combinator, which corresponds to a dependent form of the Kripke axiom [13], and which generalizes $\circledast_\mu$ to dependent products $(x : A) \to B(x)$.

3.2 Idempotent Comonads in MTT

A great deal of prior work in modal type theory has focused on *comonads* [24, 27, 54, 63], and in particular *idempotent* comonads. Shulman [63, Theorem 4.1] has shown that such modalities necessitate changes to the judgmental structure, as the only idempotent comonads that are internally definable are of the form $- \times U$ for some proposition U. In this section we present a mode theory for idempotent comonads, and prove that the resulting type theory internally satisfies the expected equations using just the combinators of the previous section.

We define the mode theory $\mathcal{M}_{\mathrm{ic}}$ to consist of a single mode m, and a single non-trivial morphism $\mu : m \to m$. We will enforce idempotence by setting $\mu \circ \mu = \mu$. Finally, in order to induce a morphism $\langle \mu \mid A \rangle \to A$ we include a unique non-trivial 2-cell $\epsilon : \mu \to 1$. We force uniqueness of this 2-cell by imposing equations such as $\mu \star \epsilon = \epsilon \star \mu = \epsilon$. The resulting mode theory is a 2-category, albeit a very simple one: it is in fact only a *poset-enriched* category.

We can show that $\langle \mu \mid A \rangle$ is a comonad by defining the expected operations using the combinators of Section 3.1:

$$\begin{aligned} &\mathsf{dup}_A : \langle \mu \mid A \rangle \to \langle \mu \mid \langle \mu \mid A \rangle \rangle && \mathsf{extract}_A : \langle \mu \mid A \rangle \to A^{\epsilon} \\ &\mathsf{dup}_A \triangleq \mathbf{comp}^{-1}_{\mu,\mu} && \mathsf{extract}_A \triangleq \mathbf{coe}[\epsilon : \mu \Rightarrow 1] \end{aligned}$$

We must also show that dup_A and $\mathsf{extract}_A$ satisfy the comonad laws, but that automatically follows from general facts pertaining to **coe** and **comp**.[1] This is indicative of the benefits of using MTT: every general result about it also applies to this instance, including the canonicity theorem of Section 5.

4 The Substitution Calculus of MTT

Until this point we have presented a curated, high-level view of MTT, and we have avoided any discussion of its metatheory. Yet, these syntactic aspects can be quite complex, and have historically proven to be sticking points for modal type theory. While these details are not necessary for the casual reader, it is essential to validate that MTT is syntactically well-behaved, enjoying e.g. a substitution principle.

We have opted for a modern approach in the analysis of MTT by presenting it as a *generalized algebraic theory* (GAT) [17, 34]. While this simplifies the study of its semantics (see Section 5), it can also be used to study the syntax. For example, the formulation of MTT as a GAT naturally leads us to include *explicit substitutions* [26, 43] in the syntax. Thus, substitution in MTT is not a metatheoretic operation on raw terms, but a piece of the syntax. This presentation helps us carefully state the equations that govern substitutions and their interaction with type formers. We consequently obtain an elegant *substitution calculus*, which can often be quite complex for modal type theories. We only discuss the modal aspects of substitution here; the full calculus may be found in the technical report.

Modal substitutions In addition to the usual rules, MTT features substitutions corresponding to the 1- and 2-cells of the mode theory. First, recall that for each modality $\mu : n \to m$ we have the operation 🔒_μ on contexts. In keeping with the algebraic syntax, we will write $-.\text{🔒}_\mu$ instead of $-, \text{🔒}_\mu$ in this section. We extend its action to substitutions:

$$\textsc{sb/lock}\quad \frac{\mu : n \to m \qquad \Gamma \vdash \delta : \Delta \mathrel{@} m}{\Gamma.\text{🔒}_\mu \vdash \delta.\text{🔒}_\mu : \Delta.\text{🔒}_\mu \mathrel{@} n}$$

Second, each 2-cell $\alpha : \mu \Rightarrow \nu$ induces a *natural transformation* between 🔒_ν and 🔒_μ, whose component at Γ is

$$\textsc{sb/key}\quad \frac{\alpha : \mu \Rightarrow \nu}{\Gamma.\text{🔒}_\nu \vdash \text{🗝}^{\alpha}_{\Gamma} : \Gamma.\text{🔒}_\mu \mathrel{@} n}$$

These substitutions come with equations that ensure that $-.\text{🔒}_\mu$ is a functor, $\text{🗝}^{\alpha}_{\Gamma}$ is a natural transformation, and that together they form a 2-functor $\mathcal{M}^{\mathrm{coop}} \to \mathbf{Cat}$: see Fig. 3.

While it is no longer necessary to prove that substitution is *admissible*, we would like to show that explicit substitutions can be pushed inside terms, and ultimately eliminated on closed terms. The proof of canonicity (Theorem 5.5) implicitly contains such an algorithm, but it is overkill: a simple argument directly proves that all explicit substitutions can be propagated down to variables.

Moreover, we may define the admissible operation mentioned in Section 2.3 by letting $A^{\alpha} \triangleq A[\text{🗝}^{\alpha}_{\Gamma}]$, and using the algorithm mentioned above to derive steps that eliminate the 'key' substitution.

Pushing substitutions under modalities In order for the aforementioned algorithm to work, we must specify how substitutions commute with the modal connectives of MTT. Unlike previous work [28], the necessary equations are straightforward:

$$\begin{aligned} \langle \mu \mid A \rangle[\delta] &= \langle \mu \mid A[\delta.\text{🔒}_\mu] \rangle \\ \mathsf{mod}_\mu(M)[\delta] &= \mathsf{mod}_\mu(M[\delta.\text{🔒}_\mu]) \end{aligned}$$

This simplicity is not coincidental. Previous modal type theories included rules that, in one way or another, *trimmed* the context during type checking: some removed variables [54, 56, 63], while others erased context formers, e.g. locks [13,

[1] In particular, our modal combinators satisfy a variant of the *interchange law* of a 2-category.

$$\textsc{sb/lock-id}\quad \frac{\mu : n \longrightarrow m}{\Gamma \vdash \mathsf{id}.\text{🔒}_\mu = \mathsf{id} : \Gamma \,@\, n} \qquad \textsc{sb/id-lock}\quad \frac{\Gamma \vdash \delta : \Delta \,@\, m}{\Gamma \vdash \delta.\text{🔒}_1 = \delta : \Delta \,@\, m}$$

$$\textsc{sb/lock-compose}\quad \frac{\mu : n \longrightarrow m \qquad \Gamma_0 \vdash \gamma_1 : \Gamma_1 \,@\, m \qquad \Gamma_1 \vdash \gamma_2 : \Gamma_2 \,@\, m}{\Gamma_0.\text{🔒}_\mu \vdash (\gamma_2 \circ \gamma_1).\text{🔒}_\mu = (\gamma_2.\text{🔒}_\mu) \circ (\gamma_1.\text{🔒}_\mu) : \Gamma_2.\text{🔒}_\mu \,@\, m}$$

$$\textsc{sb/compose-lock}\quad \frac{\mu : n \longrightarrow m \qquad \nu : o \longrightarrow n \qquad \Gamma \vdash \delta : \Delta \,@\, m}{\Gamma.\text{🔒}_{\mu\circ\nu} \vdash \delta.\text{🔒}_{\mu\circ\nu} = \delta.\text{🔒}_\mu.\text{🔒}_\nu : \Delta.\text{🔒}_{\mu\circ\nu} \,@\, m}$$

$$\textsc{sb/natural}\quad \frac{\mu, \nu : n \longrightarrow m \qquad \alpha : \nu \Rightarrow \mu \qquad \Gamma \vdash \delta : \Delta \,@\, m}{\Gamma.\text{🔒}_\mu \vdash \text{🔑}^\alpha_\Delta \circ (\delta.\text{🔒}_\mu) = (\delta.\text{🔒}_\nu) \circ \text{🔑}^\alpha_\Gamma : \Delta.\text{🔒}_\nu \,@\, n}$$

Figure 3. Selection of rules from the equational theory of modal substitutions.

27]. In either case, it was necessary to show that the trimming operation, which we may write as $\|\Gamma\|$, is functorial: $\Gamma \vdash \delta : \Delta$ should imply $\|\Gamma\| \vdash \|\delta\| : \|\Delta\|$. Unfortunately, the proof of this fact is almost always very complicated. Some type theories avoid it by 'forcing' substitution to be admissible using delayed substitutions [11, 40], but this causes serious complications to the equational theory.

MTT circumvents this by avoiding any context trimming. As a result, we need neither delayed substitutions nor a complex proof of admissibility.

5 The Semantics of MTT

As mentioned in Section 4, we have structured MTT as a GAT. As a result, MTT *automatically* induces a category of models and (strict) homomorphisms between them [17, 34]. However, this notion of model follows the syntax quite closely. In order to work with it more effectively we factor it into pieces, using the more familiar definition of *categories with families* (CwFs) [25].[2] We will then use this notion of model to present a *semantic* proof of canonicity via gluing [5, 23, 33, 62].

Like MTT itself, the definition of model is parametrized by a mode theory, so we fix a mode theory $\mathcal{M}$.

Mode-local structure Recall that MTT is divided into several modes, each of which is closed under the standard connectives of MLTT. Accordingly, a model of MTT requires a CwF $(C[m], \mathcal{T}_m, \widetilde{\mathcal{T}}_m)$ for each mode $m \in \mathcal{M}$. Each CwF is required to be a model of MLTT with $\sum$, $\prod$ and Id types, and a Coquand-style universe. This part of the definition is entirely standard, and can be found in the literature [8, 25, 31]. The novel portion of a MTT model describes the relations between CwFs induced by the 1- and 2-cells of $\mathcal{M}$.

Locks and keys Recall that for Γ ctx $@\, m$ and $\mu : n \longrightarrow m$ we have a context $\Gamma, \text{🔒}_\mu$ ctx $@\, n$, and that this construction extends functorially to substitutions. Hence, we will require for each modality $\mu : n \longrightarrow m$ a functor $[\![\text{🔒}_\mu]\!] : C[m] \longrightarrow C[n]$. Similarly, each $\alpha : \mu \Rightarrow \nu$ induces a natural transformation from $-, \text{🔒}_\nu$ to $-, \text{🔒}_\mu$. Accordingly, a model should come with a natural transformation $[\![\text{🔑}^\alpha]\!] : [\![\text{🔒}_\nu]\!] \Rightarrow [\![\text{🔒}_\mu]\!]$. Moreover, the equalities of Fig. 3 require that the assignments $\mu \mapsto \text{🔒}_\mu$ and $\alpha \mapsto \text{🔑}^\alpha$ be strictly 2-functorial. Thus, this part of the model can be succinctly summarized by requiring a 2-functor $C[-] : \mathcal{M}^{\mathrm{coop}} \longrightarrow \mathbf{Cat}$. The contravariance accounts for the fact μ corresponds to $\langle \mu \mid - \rangle$, but that the functor $[\![\text{🔒}_\mu]\!]$ models $-, \text{🔒}_\mu$, which acts with the opposite variance.

Modal comprehension structure Context declarations in MTT are annotated with a modality, and the context extension rule cx/extend involves locks. Thus, our CwFs should be equipped with more structure than mere context extension to support it.

Recall that, in an ordinary CwF C, given a context $\Gamma \in C$ and a type $A \in \mathcal{T}(\Gamma)$ we have a context $\Gamma.A$ along with a substitution $\mathbf{p} : \Gamma.A \longrightarrow \Gamma$, and a term $\mathbf{q} \in \widetilde{\mathcal{T}}(\Gamma.A, A[\mathbf{p}])$.

To model MTT we need a modal comprehension operation, which for each context $\Gamma \in C[m]$, modality $\mu : n \longrightarrow m$, and type $A \in \mathcal{T}_n([\![\text{🔒}_\mu]\!](\Gamma))$ yields

- a context $\Gamma.(\mu \mid A) \in C[m]$,
- a substitution $\mathbf{p} : \Gamma.(\mu \mid A) \longrightarrow \Gamma$, and
- a term $\mathbf{q} \in \widetilde{\mathcal{T}}_n([\![\text{🔒}_\mu]\!](\Gamma.(\mu \mid A)), A[[\![\text{🔒}_\mu]\!](\mathbf{p})])$

where $\Gamma.(\mu \mid A)$ is universal in an appropriate sense.

Intuitively, $\mathbf{q}$ corresponds to tm/var/counit. As mentioned before, this suffices to model the full variable rule tm/var, as $\mathbf{p}$, 🔑^α_-, and $\mathbf{q}$ can be used to define it from tm/var/counit.

Modal types The interpretation of the modal type $\langle \mu \mid - \rangle$ for a modality $\mu : n \longrightarrow m$ requires operations for the formation, introduction, and elimination rules. Just as with the other connectives, these are a direct translation of the rules tp/modal, tm/modal-intro, and tm/modal-elim to the language of CwFs. For example, for every $\Gamma \in C[m]$, $A \in \mathcal{T}_n([\![\text{🔒}_\mu]\!](\Gamma))$, and $M \in \widetilde{\mathcal{T}}_n([\![\text{🔒}_\mu]\!](\Gamma), A)$, we require $\mathbf{mod}_\mu(M) \in \widetilde{\mathcal{T}}_m(\Gamma, \mathbf{Mod}_\mu(A))$.

This discussion leads to the following definition.

Definition 5.1. A model of MTT is a 2-functor $C[-] : \mathcal{M}^{\mathrm{coop}} \longrightarrow \mathbf{Cat}$, equipped with the following structure:

- for each $m \in \mathcal{M}$, a CwF $(C[m], \mathcal{T}_m, \widetilde{\mathcal{T}}_m)$ that is closed under $\prod$, $\sum$, Id, and U,
- a modal comprehension structure for $\mathcal{M}$ on these CwFs, and

[2] In the technical report we have used a more categorical presentation of CwFs, known as *natural models* [8]. However, in the interest of clarity we state our results in terms of CwFs here.

- for each modality $\mu : n \to m$, a modal type structure $(\mathbf{Mod}_\mu, \mathbf{mod}_\mu, \mathbf{open}_\mu)$.

Definition 5.2. A morphism between models $F : \mathcal{C}[-]_1 \to \mathcal{C}[-]_2$ is a strict 2-natural transformation such that each $F_m : \mathcal{C}[m]_1 \to \mathcal{C}[m]_2$ is part of a strict CwF morphism [18] which strictly preserves modal comprehension and types.

We observed in Section 2.3 that modalities in MTT are weaker than DRAs [13].[3] Since DRAs are often easier to construct, we make this relation formal.

Theorem 5.3. *A 2-functor $\mathcal{C}[-] : \mathcal{M}^{\text{coop}} \to \mathbf{Cat}$ satisfying the following two conditions induces a model of MTT:*

1. *for each $m \in \mathcal{M}$, there is a CwF $(\mathcal{C}[m], \mathcal{T}_m, \widetilde{\mathcal{T}}_m)$ that is closed under $\prod$, $\sum$, Id, and U.*
2. *for each $\mu : n \to m$, $[\![\text{🔒}_\mu]\!] : \mathcal{C}[m] \to \mathcal{C}[n]$ has a DRA.*

In practice virtually all the models of MTT that we consider will be constructed by applying Theorem 5.3. We can also use it to immediately prove consistency:

Corollary 5.4. *There is no closed term of type $\mathsf{Id}_{\mathbb{B}}(\mathsf{tt}, \mathsf{ff})$.*

Proof. By Theorem 5.3, any model $\mathcal{C}$ of MLTT is a valid model of MTT: send each mode to $\mathcal{C}$, and each modality to the identity. Therefore, a closed term of type $\mathsf{Id}_{\mathbb{B}}(\mathsf{tt}, \mathsf{ff})$ in MTT would also be a term of the same type in MLTT. We may therefore reduce the consistency of MTT to that of a model of MLTT, and in particular the set-theoretic one. □

5.1 Canonicity

We can now use MTT models to prove *canonicity* via gluing. Canonicity is an important metatheoretic result: it establishes the computational adequacy of MTT by ensuring that every *closed* term already is in or is equal to a *canonical form*—a value. Canonicity is traditionally established through a *logical relation* [42, 65]. However, this method becomes very complicated when we have universes, as their presence makes the definition by induction on types impossible. It is instead necessary to construct a (large) relation on types, which associates a pair of types with a PER; the logical relation on terms is then subordinated to this relation on types [4, 6]. This technique requires significant effort, and involves many proofs by simultaneous induction.

This approach can be simplified by replacing proof-*irrelevant* logical relations by a proof-relevant gluing construction [45]. This leads to the construction of a model in which (a) types are paired with proof-relevant predicates and (b) terms are equivalence classes of syntactic terms, along with a (type-determined) proof of their canonicity. The proof-relevance is crucial in the case of the universe, which contains not just the canonicity data for $A : \mathsf{U}$ but also the predicate for $\mathsf{El}(A)$.

[3] While Birkedal et al. [13] only consider endofunctors, there is no obstacle to extending the definition of a DRA to different categories.

The full details of the glued model can be found in the technical report. Once we construct it, the initiality of syntax [17, 34] provides a witness of canonicity for every term.

Theorem 5.5 (Canonicity). *If $\cdot, \text{🔒}_\nu \vdash M : A @ m$ is a closed term, then the following conditions hold:*

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

6 Applying MTT

We will now show concretely how MTT can be used in specific modal situations by varying the mode theory. We will focus on two different examples: *guarded recursion* [15, 20, 47], which captures productive recursive definitions through a combination of modalities, and *adjoint modalities* [39, 40, 57, 63, 67], where two modalities form an adjunction internal to the type theory. In both cases we will show how to reconstruct examples from *op. cit.* in MTT. The case of guarded recursion is particularly noteworthy, as the specialization of MTT to the appropriate mode theory leads to a new syntax which is considerably simpler than previous work.

6.1 Guarded Recursion

The key idea of guarded recursion [47] is to use the *later modality* ($\blacktriangleright$) to mark data which may only be used after some progress has been made, thereby enforcing productivity at the level of types. Concretely, the later modality is equipped with three basic operations:

$$\mathsf{next} : A \to \blacktriangleright A \qquad (\circledast) : \blacktriangleright(A \to B) \to \blacktriangleright A \to \blacktriangleright B$$

$$\mathsf{l\ddot{o}b} : (\blacktriangleright A \to A) \to A$$

The first two operators make $\blacktriangleright$ into an applicative functor [44] while the third, which is known as Löb induction, encodes guarded recursion: it enables us to define a term recursively, provided the recursion is provably productive.

The perennial example is, of course, the guarded stream type $\mathsf{Str}_A \cong A \times \blacktriangleright \mathsf{Str}_A$. This recursive type requires that the head of the stream is immediately available, but the tail may only be accessed after some productive work has taken place. This allows us to e.g. construct an infinite stream of ones:

$$\mathsf{inf\_stream\_of\_ones} \triangleq \mathsf{l\ddot{o}b}(s.\ \mathsf{cons}(1, s))$$

However, Str_A does not behave like a coinductive type: we may only define *causal* operations on streams, which excludes e.g. tail. In order to regain coinductive behavior, Clouston et al. [20] introduced a second modality, $\square$ ('always'), an idempotent comonad for which

$$\square \blacktriangleright A \simeq \square A. \qquad (*)$$

Combining this modality with $\blacktriangleright$ has proved rather tricky: previous work has used *delayed substitutions* [15], or has

$$
\begin{tikzcd}
t \arrow[loop left, "\ell"] \arrow[r, bend left, "\gamma"] & s \arrow[l, bend left, "\delta"]
\end{tikzcd}
\qquad
\begin{array}{cc}
\delta \circ \gamma \leq 1 & 1 = \gamma \circ \delta \\
1 \leq \ell & \gamma = \gamma \circ \ell
\end{array}
$$

Figure 4. $\mathcal{M}_g$: a mode theory for guarded recursion.

replaced $\Box$ with *clock quantification* [7, 9, 16, 46]. The former poses serious implementation issues, and—while more flexible—the latter does not enjoy the conceptual simplicity of a single modality. In contrast, MTT enables us to effortlessly combine the two modalities and satisfy Eq. $(*)$.

To encode guarded recursion inside MTT, we must

1. choose a mode theory which induces an applicative functor $\blacktriangleright$ and a comonad $\Box$ satisfying Eq. $(*)$,
2. construct the *intended model* of MTT with this mode theory, i.e. a model where these modalities are interpreted in the standard way [14], and
3. include Löb induction as an axiom.

To begin, we define $\mathcal{M}_g$ to be the mode theory generated by Fig. 4. We require that $\mathcal{M}_g$ is poset-enriched, i.e. that there is at most one 2-cell between a pair of modalities, μ, ν, which we denote $\mu \leq \nu$ when it exists. As $\mathcal{M}_g$ is not a full 2-category, we do not need to state any coherence equations between 2-cells.

Unlike prior guarded type theories, Fig. 4 has *two modes*. We will think of elements of s as being *constant types and terms*, while types in t may *vary over time*. The reason for enforcing this division will become apparent in Theorem 6.3, but for now observe that we can construct an idempotent comonad $b \triangleq \delta \circ \gamma$.

Lemma 6.1. *$\langle b \mid - \rangle$ is an idempotent comonad and $\langle \ell \mid - \rangle$ is an applicative functor.*

Proof. Follows from the combinators in Section 3. $\square$

Next, Eq. $(*)$, which was hard to force in previous type theories, is provable: as $\gamma \circ \ell = \gamma$, the combinator $\mathbf{comp}_{b,\ell}$ from Section 3.1 has the appropriate type:

$$\mathbf{comp}_{b,\ell} : \langle b \mid \langle \ell \mid A \rangle \rangle \simeq \langle b \circ \ell \mid A \rangle = \langle b \mid A \rangle$$

In order to construct the intended model, recall that the standard interpretation of guarded type theory uses the *topos of trees*, $\mathbf{PSh}(\omega)$: see Birkedal et al. [14] for a thorough discussion. Crucially, it is easy to see that $\Box = \Delta \circ \Gamma$, where

$$
\begin{array}{ll}
\Gamma : \mathbf{PSh}(\omega) \longrightarrow \mathbf{Set} & \Delta : \mathbf{Set} \longrightarrow \mathbf{PSh}(\omega) \\
\Gamma \triangleq X \mapsto \mathrm{Hom}(1, X) & \Delta \triangleq S \mapsto \lambda\_.S
\end{array}
$$

As both $\mathbf{Set}$ and $\mathbf{PSh}(\omega)$ are models of MLTT [14, 31], we may use Theorem 5.3 to construct the intended model.

Theorem 6.2. *There exists a model of MTT with this mode theory where $\langle b \mid - \rangle$ is interpreted as $\Box$ and $\langle \ell \mid - \rangle$ as $\blacktriangleright$.*

$$
\textsc{tm/lob}\quad
\frac{\Gamma, x : (\ell \mid A^{1 \leq \ell}) \vdash M : A \mathbin{@} t}{\Gamma \vdash \mathrm{löb}(x.\ M) : A \mathbin{@} t}
$$

$$
\textsc{tm/lob-beta}\quad
\frac{\Gamma, x : (\ell \mid A^{1 \leq \ell}) \vdash M : A \mathbin{@} t}{\Gamma \vdash \mathrm{löb}(x.\ M) = M[\mathrm{next}(\mathrm{löb}(x.\ M))/x] : A \mathbin{@} t}
$$

Figure 5. Axiomatization of Löb induction in MTT

Proof. We choose the 2-functor which sends $s \mapsto \mathbf{Set}$ and $t \mapsto \mathbf{PSh}(\omega)$. Moreover, we define $[\![\text{🔒}_\ell]\!]$, $[\![\text{🔒}_\delta]\!]$, and $[\![\text{🔒}_\gamma]\!]$ to be the left adjoints of $\blacktriangleright$, Δ, and Γ respectively [13, 49]. $\square$

From this point onwards we will write $\blacktriangleright \triangleq \langle \ell \mid - \rangle$, $\Delta \triangleq \langle \gamma \mid - \rangle$, and $\Box \triangleq \langle \delta \mid - \rangle$.

The only thing that remains is to add Löb induction. This is a *modality-specific* operation that cannot be expressed in the mode theory, so we must add it as an axiom: see Fig. 5 for the precise formulation. Unfortunately, any axiom disrupts the metatheory of MTT so canonicity no longer applies. However, adding it to the type theory is sound, as the model supports it. At this point we may as well assume *equality reflection* [32], as is commonplace in previous guarded type theories [15]. This is stronger than necessary (function extensionality would suffice), but it simplifies proofs and makes comparison to previous work more direct.

Programming with Guarded MTT We can now use MTT to program with and reason about guarded recursion. For instance, we can define coinductive streams:

$$
\begin{array}{l}
\mathrm{Str} : \mathrm{U} \longrightarrow \mathrm{U} \mathbin{@} s \\
\mathrm{Str}(A) \triangleq \Gamma(\mathrm{löb}(S.\ \Delta(A) \times \blacktriangleright S))
\end{array}
$$

Unlike prior guarded type theories, we have defined this stream operator not in mode t, which represents $\mathbf{PSh}(\omega)$, but in mode s, which represents $\mathbf{Set}$. Accordingly, this definition does not use $\Box$. It first uses Δ to convert A to a t-type, and then Γ to move the recursive definition back to s. This alleviates some bookkeeping: in previous work [15] the stream type was actually coinductive only if A was a constant type (i.e. $A \simeq \Box A$). Accordingly, theorems about streams had to pass around proofs that the elements of the stream are constant. In our case, defining Str at mode s ensures that the elements of the stream are automatically constant. Hence, $\mathrm{Str}(A)$ is equivalent to the familiar definition, but it is no longer necessary to carry through proofs of constancy. Therefore, for *any* $A : \mathrm{U} \mathbin{@} s$ we have

Theorem 6.3. *$\mathrm{Str}(A)$ is the final coalgebra for $S \mapsto A \times S$ in mode s.*

We can also program with $\mathrm{Str}(A)$ by more directly appealing to the underlying guarded structure. For instance, we can define a 'zip with' function. Let $\mathsf{Str}'_A = \text{löb}(S.\ \Delta(A) \times \blacktriangleright S)$ and write z_h and z_t for $\mathsf{pr}_0(z)$ and $\mathsf{pr}_1(z)$ respectively:

$$\begin{aligned}
&\mathsf{zipWith}' : \Delta(A \to B \to C) \to \mathsf{Str}'_A \to \mathsf{Str}'_B \to \mathsf{Str}'_C \\
&\mathsf{zipWith}'(f) \triangleq \text{löb}(r.\ \lambda x, y.\ (f \circledast_\delta x_h \circledast_\delta y_h, r \circledast_\ell x_t \circledast_\ell y_t)
\end{aligned}$$

$$\begin{aligned}
&\mathsf{zipWith} : (A \to B \to C) \to \mathsf{Str}(A) \to \mathsf{Str}(B) \to \mathsf{Str}(C) \\
&\mathsf{zipWith}(f) \triangleq \lambda x, y.\ \mathsf{mod}_\gamma(\mathsf{zipWith}'(\mathsf{mod}_\delta(f))) \circledast_\gamma x \circledast_\gamma y
\end{aligned}$$

where $\circledast_\mu$ is defined in Section 3.1.

We can also use dependent types to reason about guarded recursive programs. For example,

Theorem 6.4. *If f is commutative then $\mathsf{zipWith}(f)$ is commutative. That is, given $A, B : \mathsf{U}$ and $f : A \to A \to B$ there is a term of the following type:*

$$\begin{aligned}
&((a_0, a_1 : A) \to \mathsf{Id}(f(a_0, a_1), f(a_1, a_0))) \to \\
&\quad (s_0, s_1 : \mathsf{Str}(A)) \to \mathsf{Id}(\mathsf{zipWith}(f, s_0, s_1), \mathsf{zipWith}(f, s_1, s_0))
\end{aligned}$$

All things considered, instantiating MTT with $\mathcal{M}_g$ yields a highly expressive guarded dependent type theory with coinductive types. Unlike prior systems, e.g. Bahr et al. [9], we do not need clock variables or syntactic checks of constancy. Moreover, the syntax is more robust than previous work that combines $\square$ and $\blacktriangleright$ [15, 20], as there is no need for delayed substitutions. Unfortunately, the addition of the Löb axiom means Theorem 5.5 cannot be directly applied, but the syntax remains sound and tractable.

6.2 Internal Adjunctions

Up to this point we have only considered mode theories which are poset-enriched: there is at most one 2-cell between any pair of modalities. This has worked well for describing strict structures (Section 3.2), as well as some specific settings (Section 6.1). However, we would like to use MTT to reason about less strict categorical models. In this section we will show that we can readily use MTT to reason about a pair $\nu \dashv \mu$ of adjoint modalities.

Adjoint modalities are common in modal type theory, much in the same way that adjunctions are ubiquitous in mathematics [38–40, 57, 63]. For example, the adjunction $\delta \dashv \gamma$ played an important role in the previous section. However, that particular case is unusually well-behaved, as it arises from a Galois connection. In contrast, the behavior of general adjoint modalities is much more subtle. We will show that by instantiating MTT with a particular mode theory we can internally prove many properties of adjoint modalities that have previously been established only in special cases.

To begin, we pick the *walking adjunction* [59] for our mode theory, i.e. the 2-category generated by Fig. 6. This mode theory is the *classifying 2-category* for internal adjunctions: every 2-functor $\mathcal{M}_{\mathsf{adj}}{}^{\mathsf{coop}} \simeq \mathcal{M}_{\mathsf{adj}} \to \mathbf{Cat}$ determines a pair of adjoint functors, and vice versa. Consequently, substitutions $\Delta \to \Gamma.\text{🔒}_\mu$ are in bijection with substitutions $\Delta.\text{🔒}_\nu \to \Gamma$. However, this is not enough on its own: we must also show that $\langle \nu \mid - \rangle$ and $\langle \mu \mid - \rangle$ form an adjunction *inside* MTT.

$$\begin{tikzcd} n \arrow[r, bend left, "\mu"] & m \arrow[l, bend left, "\nu"] \end{tikzcd}
\qquad
\begin{aligned}
&\eta : 1 \Rightarrow \mu \circ \nu \\
&\epsilon : \nu \circ \mu \Rightarrow 1 \\
&1_\mu = (1_\mu \star \epsilon) \circ (\eta \star 1_\mu) \\
&1_\nu = (\epsilon \star 1_\nu) \circ (1_\nu \star \eta)
\end{aligned}$$

Figure 6. $\mathcal{M}_{\mathsf{adj}}$: a mode theory for adjunctions

Recovering the adjunction in MTT We can construct the unit and counit internally:

$$\mathsf{unit} : A \to \langle \mu \mid \langle \nu \mid A^\eta \rangle\rangle \qquad \mathsf{counit} : \langle \nu \mid \langle \mu \mid A \rangle\rangle \to A^\epsilon$$

$$\begin{aligned}
&\mathsf{unit}(x) \quad \triangleq \mathsf{mod}_\mu(\mathsf{mod}_\nu(x^\eta)) \\
&\mathsf{counit}(x) \triangleq \mathsf{let}\ \mathsf{mod}_\nu(y_0) \leftarrow x\ \mathsf{in}\ \mathsf{let}_\nu\ \mathsf{mod}_\mu(y_1) \leftarrow y_0\ \mathsf{in}\ y_1^\epsilon
\end{aligned}$$

In order to account for dependence we must adjust the type A by a 2-cell. For example, in the definition of unit we assume $\Gamma \vdash A\ \mathsf{type}_1 \ @\ m$, so $\langle \mu \mid \langle \nu \mid A \rangle\rangle$ is ill-typed. We can, however, obtain a version of A that is typable in the context $\Gamma, \text{🔒}_{\mu\circ\nu}$ by applying $(-)^\eta$ to it, as in TM/VAR.

We can prove that these two operations form an adjunction by showing they satisfy the triangle identities, e.g.

$$\begin{aligned}
&_ \ : (x : \langle \nu \mid A \rangle) \to \mathsf{Id}_{\langle \nu \mid A \rangle}(x, \mathsf{counit}(\mathsf{mod}_\nu(\mathsf{unit}) \circledast_\nu x)) \\
&_ \triangleq \lambda x.\ \mathsf{let}\ \mathsf{mod}_\nu(y) \leftarrow x\ \mathsf{in}\ \mathsf{refl}(\mathsf{mod}_\nu(y))
\end{aligned}$$

This proof relies on the fact that the modalities ν and μ satisfy the triangle identities themselves in $\mathcal{M}_{\mathsf{adj}}$.

The existence of the unit and counit is enough to internally determine an adjunction. We might want to use an alternative description, e.g. to manipulate a natural bijection of hom-sets, $\mathrm{Hom}(L(A), B) \cong \mathrm{Hom}(A, R(B))$.

Unfortunately, this isomorphism cannot be recovered internally. First, notice that $\langle \nu \mid A \rangle \to B$ and $A \to \langle \mu \mid B \rangle$ are types in different modes—n and m respectively—so $(\langle \nu \mid A \rangle \to B) \simeq (A \to \langle \mu \mid B \rangle)$ is ill-typed. Second, even if $n = m$ so that ν and μ are endomodalities and this equivalence is well-typed, an internal equivalence is a stronger condition than a bijection of hom-sets: it is equivalent to an isomorphism of exponential objects $B^{L(A)} \cong R(B)^A$.

Prior work [38] addressed this by introducing a third modality $\square$, such that terms of $\square A$ represent *global* elements of A, and then requiring transposition only for functions under $\square$. Global elements of B^A are in bijection with $\mathrm{Hom}(A, B)$, so the postulated equivalence corresponds to the expected bijection. We can rephrase this argument in MTT. Suppose that $n = m$, and that $\mathrm{Hom}(m, m)$ is equipped with an initial object, i.e. a modality $\tau : m \to m$ and a unique 2-cell $! : \tau \to \xi$ for all ξ. Then,

Theorem 6.5. *The following equivalence is definable in MTT:* $\langle \tau \mid \langle \nu \mid A^! \rangle \to B \rangle \simeq \langle \tau \mid A \to \langle \mu \mid B^! \rangle\rangle$.

Crisp induction for the left adjoint Having internalized $\nu \dashv \mu$, many of the classical results about adjunctions can be replayed inside MTT. For instance, by carrying out a proof that left adjoints preserve colimits internally to MTT, we recover *modal* or *crisp induction principles* for ν [39, 63]. We can then show e.g. that $\langle \nu \mid \mathbb{B} \rangle \simeq \mathbb{B}$. However, in order to construct this equivalence it will be convenient to formulate a general induction principle for $\langle \nu \mid \mathbb{B} \rangle$.

Supposing that $\Gamma, \text{🔒}_{\nu \circ \mu} \vdash C : \langle \nu \mid \mathbb{B} \rangle \to \mathsf{U} \mathbin{@} m$, we can define a term

$$\begin{aligned} &\mathsf{if}^{\nu}_{C} : \langle \nu \circ \mu \mid C(\mathsf{mod}_{\nu}(\mathsf{tt})) \rangle \to \langle \nu \circ \mu \mid C(\mathsf{mod}_{\nu}(\mathsf{ff})) \rangle \\ &\quad \to (b : \langle \nu \mid \mathbb{B} \rangle) \to C^{\epsilon}(b) \end{aligned}$$

This is a version of the conditional that operates on $\langle \nu \mid \mathbb{B} \rangle$ rather than $\mathbb{B}$. In fact, more is possible: in the technical report we prove that if^{ν} can be constructed for any C, not just small types. Using this stronger induction principle, we can show

Theorem 6.6. $\langle \nu \mid \mathbb{B} \rangle \simeq \mathbb{B}$

Similarly, we can prove that ν preserves identity types:

Theorem 6.7. $\langle \nu \mid \mathsf{Id}_A(M, N) \rangle \simeq \mathsf{Id}_{\langle \nu \mid A \rangle}(\mathsf{mod}_{\nu}(M), \mathsf{mod}_{\nu}(N))$

This instantiation of MTT with $\mathcal{M}_{\mathrm{adj}}$ yields a systematic treatment of an internal transposition axiom [38], and is sufficiently expressive to derive crisp induction principles [63]. In both cases we can use MTT instead of a handcrafted modal type theory. Moreover, as we have not added any new axioms to deal with internal adjunctions, our canonicity result applies.

6.3 Further Examples

In addition to the examples described above, we have applied MTT to a wide variety of other situations, including

- parametricity, via degrees of relatedness [50],
- synchronous and guarded programming with warps [30],
- finer grained notions of realizability and local maps of categories of assemblies [12].

While interesting, we cannot discuss the details of these applications here for want of space. We invite the interested reader to consult the accompanying technical report.

7 Related Work

MTT is related to many prior modal type theories. In particular, its formulation draws on three important techniques: split contexts, left division, and the Fitch style.

Split-context type theories [24, 35, 48, 54, 55, 63, 67] divide the context into different *zones*, one for each modality, which are then manipulated by modal connectives. This has proven to be an effective approach for a number of modalities, and sometimes even scales to full dependent type theories [24, 63, 67]. However, the structure of contexts becomes very complex as the number of modalities increases.

In order to manage this complexity, some modal type theories employ *left-division*: each variable declaration in the context is annotated with a modality, and a *left-division operation*, which is a left adjoint to post-composition of modalities, is used to state the introduction rules [1–3, 50, 51, 53]. Left-division calculi handle multiple modalities and support full dependent types, but many important modal situations cannot be equipped with a left-division structure.

Another technique stipulates that modalities are essentially right adjoints, with the corresponding left adjoints being constructors on contexts. These *Fitch-style* type theories [9, 10, 13, 19, 27] are relatively simple, which has made them convenient for programming applications [10, 27]. Nevertheless, scaling this approach to a multimodal setting has proven difficult. In particular, extending the elimination rule to a multimodal setting remains an open problem.

MTT synthesizes these approaches by including both Fitch-style locks and left-division-style annotations in its judgmental structure. The combination of these devices circumvents the difficulties that plagued previous calculi. For example, this combination obviates the need for a left division operation, instead MTT uses a Fitch-style introduction rule. On the other hand, MTT includes a left-division-style elimination rule which smoothly accommodates multiple interacting modalities.

Most prior modal type theories have focused on incorporating a specific collection of modalities. The sole exception is the work of Licata et al. (LSR) [40]. The LSR framework supports an arbitrary collection of substructural modalities over simple types, and there is ongoing work on a dependently-typed system. The price to pay for this expressivity is practicality: for example, some LSR connectives require *delayed substitutions* [15], which complicate the equational theory, and make pen-and-paper calculations cumbersome.

8 Conclusion

We introduced and studied MTT, a dependent type theory parametrized by a mode theory that describes interacting modalities. We have demonstrated that MTT may be used to reason about several important modal settings, and proven basic metatheorems about its syntax, including canonicity.

In the future we plan to further develop the metatheory of MTT. We specifically hope to prove that MTT enjoys normalization, and hence that type-checking is decidable—provided the mode theory is. This result would pave the way to a practical implementation of a multimodal proof assistant.

We also hope to extend our analysis to some class of *modality-specific* operations, e.g. Löb induction. These operations cannot be captured by a mode theory, and so can only be added *axiomatically* to MTT (as was done in Section 6.1), thus invalidating some of our metatheorems. However, such operations play an important role in many applications, and should be accounted for in a systematic way.

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.

Alex Kavvos was supported in part by a research grant (12386, Guarded Homotopy Type Theory) from the VILLUM Foundation. Andreas Nuyts holds a PhD Fellowship from the Research Foundation - Flanders (FWO). This work was supported in part by a Villum Investigator grant (no. 25804), Center for Basic Research in Program Verification (CPV), from the VILLUM Foundation.

References

[1] Andreas Abel. 2006. *A Polymorphic Lambda-Calculus with Sized Higher-Order Types.* Ph.D. Dissertation. Ludwig-Maximilians-Universität München.

[2] Andreas Abel. 2008. Polarised subtyping for sized types. *Mathematical Structures in Computer Science* 18, 5 (2008), 797–822. https://doi.org/10.1017/S0960129508006853

[3] Andreas Abel and Gabriel Scherer. 2012. On Irrelevance and Algorithmic Equality in Predicative Type Theory. *Logical Methods in Computer Science* Volume 8, Issue 1 (03 2012). https://doi.org/10.2168/LMCS-8(1:29)2012

[4] Stuart Frazier Allen. 1987. *A non-type-theoretic semantics for type-theoretic language.* Ph.D. Dissertation.

[5] Thorsten Altenkirch and Ambrus Kaposi. 2016. Normalisation by Evaluation for Dependent Types. In *1st International Conference on Formal Structures for Computation and Deduction (FSCD 2016).* https://doi.org/10.4230/LIPIcs.FSCD.2016.6

[6] Carlo Angiuli. 2019. *Computational Semantics of Cartesian Cubical Type Theory.* Ph.D. Dissertation.

[7] Robert Atkey and Conor McBride. 2013. Productive coprogramming with guarded recursion. In *Proceedings of the 2013 ACM SIGPLAN International Conference on Functional Programming*. ACM, New York, NY.

[8] Steve Awodey. 2018. Natural models of homotopy type theory. 28, 2 (2018), 241–286. https://doi.org/10.1017/S0960129516000268

[9] P. Bahr, H. B. Grathwohl, and R. E. Møgelberg. 2017. The clocks are ticking: No more delays!. In *2017 32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*. 1–12.

[10] Patrick Bahr, Christian Uldal Graulund, and Rasmus Ejlers Møgelberg. 2019. Simply RaTT: A Fitch-style Modal Calculus for Reactive Programming Without Space Leaks. *Proc. ACM Program. Lang.* 3, ICFP, Article 109 (July 2019), 109:1–109:27 pages. https://doi.org/10.1145/3341713

[11] G. M. Bierman and V. C. V. de Paiva. 2000. On an Intuitionistic Modal Logic. *Studia Logica* 65, 3 (2000). https://doi.org/10.1023/A:1005291931660

[12] Lars Birkedal. 2000. Developing Theories of Types and Computability via Realizability. *Electronic Notes in Theoretical Computer Science* 34 (2000).

[13] Lars Birkedal, Ranald Clouston, Bassel Mannaa, Rasmus Ejlers Møgelberg, Andrew M. Pitts, and Bas Spitters. 2018. Modal Dependent Type Theory and Dependent Right Adjoints. (2018). arXiv:1804.05236 http://arxiv.org/abs/1804.05236 To appear in Mathematical Structures in Computer Science.

[14] Lars Birkedal, Rasmus Ejlers Møgelberg, Jan Schwinghammer, and Kristian Støvring. 2012. First steps in synthetic guarded domain theory: step-indexing in the topos of trees. *Logical Methods in Computer Science* (2012).

[15] Aleš Bizjak, Hans Bugge Grathwohl, Ranald Clouston, Rasmus E. Møgelberg, and Lars Birkedal. 2016. Guarded Dependent Type Theory with Coinductive Types. In *Foundations of Software Science and Computation Structures: 19th International Conference, FOSSACS 2016, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2016, Eindhoven, The Netherlands, April 2–8, 2016, Proceedings*, Bart Jacobs and Christof Löding (Eds.). Springer Berlin Heidelberg, 20–35.

[16] Ales Bizjak and Rasmus Ejlers Møgelberg. 2015. A Model of Guarded Recursion With Clock Synchronisation. *Electr. Notes Theor. Comput. Sci.* (2015). https://doi.org/10.1016/j.entcs.2015.12.007

[17] John Cartmell. 1978. *Generalised Algebraic Theories and Contextual Categories.* Ph.D. Dissertation.

[18] Simon Castellan, Pierre Clairambault, and Peter Dybjer. 2015. Undecidability of Equality in the Free Locally Cartesian Closed Category. *Logical Methods in Computer Science* 13 (04 2015). https://doi.org/10.23638/LMCS-13(4:22)2017

[19] Ranald Clouston. 2018. Fitch-Style Modal Lambda Calculi. In *Foundations of Software Science and Computation Structures*, Christel Baier and Ugo Dal Lago (Eds.). Springer International Publishing, 258–275.

[20] Ranald Clouston, Aleš Bizjak, Hans Bugge Grathwohl, and Lars Birkedal. 2015. Programming and Reasoning with Guarded Recursion for Coinductive Types. In *Foundations of Software Science and Computation Structures*, Andrew Pitts (Ed.). Springer Berlin Heidelberg, 407–421.

[21] Thierry Coquand. 1986. An Analysis of Girard's Paradox. In *Proceedings of the First Annual IEEE Symposium on Logic in Computer Science (LICS 1986).* IEEE Computer Society Press, 227–236.

[22] Thierry Coquand. 2013. Presheaf model of type theory. http://www.cse.chalmers.se/~coquand/presheaf.pdf

[23] Thierry Coquand. 2018. Canonicity and normalization for Dependent Type Theory. arXiv:1810.09367 https://arxiv.org/abs/1810.09367

[24] Valeria de Paiva and Eike Ritter. 2015. Fibrational Modal Type Theory, In Proceedings of the Tenth Workshop on Logical and Semantic Frameworks, with Applications (LSFA 2015). *Electronic Notes in Theoretical Computer Science.* https://doi.org/10.1016/j.entcs.2016.06.010

[25] Peter Dybjer. 1996. Internal type theory. In *Types for Proofs and Programs: International Workshop, TYPES '95 Torino, Italy, June 5–8, 1995 Selected Papers*, Stefano Berardi and Mario Coppo (Eds.). Springer Berlin Heidelberg, 120–134.

[26] Johan G. Granström. 2013. *Treatise on Intuitionistic Type Theory.* Springer Publishing Company, Incorporated.

[27] Daniel Gratzer, Jonathan Sterling, and Lars Birkedal. 2019. Implementing a Modal Dependent Type Theory. *Proc. ACM Program. Lang.*, Article 107 (2019), 29 pages. https://doi.org/10.1145/3341711

[28] Daniel Gratzer, Jonathan Sterling, and Lars Birkedal. 2019. Normalization-by-Evaluation for Modal Dependent Type Theory. https://jozefg.github.io/papers/2019-implementing-modal-dependent-type-theory-tech-report.pdf Technical Report for the ICFP paper by the same name.

[29] Jacob A Gross, Daniel R Licata, Max S New, Jennifer Paykin, Mitchell Riley, Michael Shulman, and Felix Wellen. 2017. Differential Cohesive Type Theory (Extended Abstract). In *Extended abstracts for the Workshop "Homotopy Type Theory and Univalent Foundations"*. https://hott-uf.github.io/2017/abstracts/cohesivett.pdf

[30] Adrien Guatto. 2018. A Generalized Modality for Recursion. In *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS '18).* ACM. https://doi.org/10.1145/3209108.3209148

[31] Martin Hofmann. 1997. *Syntax and semantics of dependent types.* Springer London, London, 13–54. https://doi.org/10.1007/978-1-4471-0963-1_2

[32] B. Jacobs. 1999. *Categorical Logic and Type Theory.* Number 141 in Studies in Logic and the Foundations of Mathematics. North Holland.

[33] Ambrus Kaposi, Simon Huber, and Christian Sattler. 2019. Gluing for type theory. In *Proceedings of the 4th International Conference on Formal Structures for Computation and Deduction (FSCD 2019)*, Herman Geuvers (Ed.), Vol. 131.

[34] Ambrus Kaposi, András Kovács, and Thorsten Altenkirch. 2019. Constructing Quotient Inductive-inductive Types. *Proc. ACM Program. Lang.* 3, POPL, Article 2 (Jan. 2019), 24 pages. https://doi.org/10.1145/3290315

[35] G. A. Kavvos. 2017. Dual-context calculi for modal logic. In *2017 32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*. 1–12. https://doi.org/10.1109/LICS.2017.8005089 arXiv:1602.04860

[36] G. A. Kavvos. 2019. Modalities, Cohesion, and Information Flow. *Proceedings of the ACM on Programming Languages* 3, POPL (Jan. 2019), 20:1–20:29. https://doi.org/10.1145/3290333

[37] William F. Lawvere. 2007. Axiomatic Cohesion. *Theory and Applications of Categories* 19 (06 2007), 41–49.

[38] Daniel R. Licata, Ian Orton, Andrew M. Pitts, and Bas Spitters. 2018. Internal Universes in Models of Homotopy Type Theory. In *3rd International Conference on Formal Structures for Computation and Deduction, FSCD 2018, July 9-12, 2018, Oxford, UK*. 22:1–22:17. https://doi.org/10.4230/LIPIcs.FSCD.2018.22

[39] Daniel R. Licata and Michael Shulman. 2016. *Adjoint Logic with a 2-Category of Modes*. Springer International Publishing, 219–235. https://doi.org/10.1007/978-3-319-27683-0_16

[40] Daniel R. Licata, Michael Shulman, and Mitchell Riley. 2017. A Fibrational Framework for Substructural and Modal Logics. In *2nd International Conference on Formal Structures for Computation and Deduction (FSCD 2017) (Leibniz International Proceedings in Informatics (LIPIcs))*, Dale Miller (Ed.), Vol. 84.

[41] Zhaohui Luo. 2012. Notes on Universes in Type Theory. http://www.dcs.rhul.ac.uk/home/zhaohui/universes.pdf Notes for a talk at Institute for Advanced Study in Princeton in Oct 2012.

[42] Per Martin-Löf. 1975. An intuitionistic theory of types: predicative part. In *Logic Colloquium '73, Proceedings of the Logic Colloquium*, H.E. Rose and J.C. Shepherdson (Eds.). Studies in Logic and the Foundations of Mathematics, Vol. 80. North-Holland, 73–118.

[43] Per Martin-Löf. 1992. Substitution calculus. Notes from a lecture given in Göteborg.

[44] Conor McBride and Ross Paterson. 2008. Applicative Programming with Effects. *J. Funct. Program.* 18, 1 (Jan. 2008), 1–13. https://doi.org/10.1017/S0956796807006326

[45] John C. Mitchell and Andre Scedrov. 1993. Notes on sconing and relators. In *Computer Science Logic*, E. Börger, G. Jäger, H. Kleine Büning, S. Martini, and M. M. Richter (Eds.). Springer Berlin Heidelberg, 352–378.

[46] Rasmus Ejlers Møgelberg. 2014. A Type Theory for Productive Coprogramming via Guarded Recursion. In *Proceedings of the Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS) (CSL-LICS '14)*. https://doi.org/10.1145/2603088.2603132

[47] H. Nakano. 2000. A modality for recursion. In *Proceedings Fifteenth Annual IEEE Symposium on Logic in Computer Science (Cat. No.99CB36332)*. IEEE Computer Society, 255–266.

[48] Aleksandar Nanevski, Frank Pfenning, and Brigitte Pientka. 2008. Contextual modal type theory. *ACM Transactions Computational Logic* 9 (June 2008). https://doi.org/10.1145/1352582.1352591

[49] Andreas Nuyts. 2018. Presheaf Models of Relational Modalities in Dependent Type Theory. arXiv:cs.LO/1805.08684

[50] Andreas Nuyts and Dominique Devriese. 2018. 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. https://doi.org/10.1145/3209108.3209119

[51] Andreas Nuyts, Andrea Vezzosi, and Dominique Devriese. 2017. Parametric Quantifiers for Dependent Type Theory. *Proc. ACM Program. Lang.* 1, ICFP (2017). https://doi.org/10.1145/3110276

[52] Erik Palmgren. 1998. On universes in type theory. *Twenty Five Years of Constructive Type Theory* (1998), 191–204. http://www2.math.uu.se/~palmgren/universe.pdf

[53] Frank Pfenning. 2001. Intensionality, Extensionality, and Proof Irrelevance in Modal Type Theory. In *Symposium on Logic in Computer Science*. 01.

[54] Frank Pfenning and Rowan Davies. 2001. A Judgmental Reconstruction of Modal Logic. *Mathematical Structures in Computer Science* 11 (2001), 511–540.

[55] Brigitte Pientka, Andreas Abel, Francisco Ferreira, David Thibodeau, and Rébecca Zucchini. 2019. A Type Theory for Defining Logics and Proofs. In *Proceedings of the 34th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*.

[56] Dag Prawitz. 1967. Natural Deduction. A Proof-Theoretical Study. *Journal of Symbolic Logic* 32, 2 (1967), 255–256.

[57] Jason Reed. 2009. A Judgmental Deconstruction of Modal Logic. (2009). http://www.cs.cmu.edu/~jcreed/papers/jdml.pdf Manuscript.

[58] Egbert Rijke, Michael Shulman, and Bas Spitters. 2020. Modalities in homotopy type theory. *Logical Methods in Computer Science* Volume 16, Issue 1 (2020). https://lmcs.episciences.org/6015

[59] Stephen Schanuel and Ross Street. 1986. The free adjunction. *Cahiers de topologie et géométrie différentielle catégoriques* 27, 1 (1986), 81–83.

[60] Urs Schreiber. 2013. Differential cohomology in a cohesive infinity-topos. *arXiv e-prints*, Article arXiv:1310.7930 (10 2013), arXiv:1310.7930 pages. arXiv:math-ph/1310.7930

[61] Urs Schreiber and Michael Shulman. 2014. Quantum Gauge Field Theory in Cohesive Homotopy Type Theory. In *Proceedings 9th Workshop on Quantum Physics and Logic Brussels, Belgium, 10-12 October 2012*. 109–126. https://doi.org/10.4204/EPTCS.158.8

[62] Michael Shulman. 2014. Univalence for inverse diagrams and homotopy canonicity. *Mathematical Structures in Computer Science* (2014). https://doi.org/10.1017/s0960129514000565

[63] Michael Shulman. 2018. Brouwer's fixed-point theorem in real-cohesive homotopy type theory. *Mathematical Structures in Computer Science* 28, 6 (2018), 856–941. https://doi.org/10.1017/S0960129517000147

[64] Jonathan Sterling. 2019. Algebraic Type Theory and Universe Hierarchies. (2019). arXiv:1902.08848 http://arxiv.org/abs/1902.08848

[65] W. W. Tait. 1967. Intensional Interpretations of Functionals of Finite Type I. 32, 2 (1967), 198–212. http://www.jstor.org/stable/2271658

[66] The Univalent Foundations Program. 2013. *Homotopy Type Theory: Univalent Foundations of Mathematics*. https://homotopytypetheory.org/book, Institute for Advanced Study.

[67] Colin Zwanziger. 2019. Natural Model Semantics for Comonadic and Adjoint Type Theory: Extended Abstract. In *Preproceedings of Applied Category Theory Conference 2019*.