Degrees of Relatedness

A Unified Framework for Parametricity, Irrelevance, Ad Hoc Polymorphism, Intersections, Unions and Algebra in Dependent Type Theory

Andreas Nuyts
imec-DistriNet, KU Leuven, Belgium

Dominique Devriese
imec-DistriNet, KU Leuven, Belgium

Abstract

Dependent type theory allows us to write programs and to prove properties about those programs in the same language. However, some properties do not require much proof, as they are evident from a program's implementation, e.g. if a polymorphic program is not ad hoc but relationally parametric, then we get parametricity theorems for free. If we want to safely shortcut proofs by relying on the evident good behaviour of a program, then we need a type-level guarantee that the program is indeed well-behaved. This can be achieved by annotating function types with a modality describing the behaviour of functions.

We consider a dependent type system with modalities for relational parametricity, irrelevance (i.e. type-checking time erasability of an argument) and ad hoc polymorphism. The interplay of three modalities and dependent types raises a number of questions. For example: If a term depends on a variable with a given modality, then how should its type depend on it? Are all modalities always applicable, e.g. should we consider parametric functions from the booleans to the naturals? Do we need any further modalities in order to properly reason about these three?

We develop a type system, based on a denotational presheaf model, that answers these questions. The core idea is to equip every type with a number of relations — just equality for small types, but more for larger types — and to describe function behaviour by saying how functions act on those relations. The system has modality-aware equality judgements (ignoring irrelevant parts) and modality-aware proving operators (for proving free theorems) which are even self-applicable. It also supports sized types, some form of intersection and union types, and parametric quantification over algebraic structures. We prove soundness in a denotational presheaf model.

CCS Concepts • **Theory of computation → Type theory**;

Keywords Parametricity, irrelevance, erasure, cubical type theory, presheaf semantics, intersections, unions, algebra in type theory

ACM Reference Format:
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 *LICS '18: 33rd Annual ACM/IEEE Symposium on Logic in Computer Science, July 9–12, 2018, Oxford, United Kingdom.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3209108.3209119

See the first author's website for a version with appendices.

1 Introduction

By now, several dependent type systems can be found in the literature in which function types are annotated with a *modality* which restricts the behaviour of the functions they contain. A modality for compile-time erasability is found in (adaptations of) Miquel's implicit calculus of constructions (ICC) [7, 22, 23]. Modalities for type-checking time erasability (which we will call **irrelevance**) are found in [3, 7, 23, 27, 29]. Our previous work with Vezzosi [26] has a modality for relationally **parametric** functions, which we can prove free theorems about, and a 'pointwise' modality for functions which break the relational structure, comparable to **ad hoc polymorphism**.

Example 1.1 (Parametricity)**.** The polymorphic if operator takes four arguments: a type X of type $\mathcal{U}$ (which is a type of types), a boolean, and two elements $x_1, x_2 : X$ of which it picks one, depending on the boolean.

$$if : (\mathbf{par} \shortmid X : \mathcal{U}) \to \mathsf{Bool} \to X \to X \to X \qquad (1)$$

In the above type signature, we used the annotation **par** to signal that the argument X is used parametrically, allowing us to rely on free theorems such as $f\ (if\ X\ b\ x_1\ x_2) = if\ Y\ b\ (f\ x_1)\ (f\ x_2)$, for any $f : X \to Y$, irrespective of the implementations of f and if.

Example 1.2 (Irrelevance)**.** Let $\mathsf{List}_n\ A$ be the type of lists of length less than n. Size bounds such as n can be used as a modular way of ensuring termination of recursive functions [4, 5, 14]. Possible type signatures for the constructors of lists are then:

$$\begin{aligned} \mathsf{nil} &: (\mathbf{par} \shortmid X : \mathcal{U}) \to (\mathbf{irr} \shortmid n : \mathbb{N}) \to (\mathbf{irr} \shortmid 0 < n) \to \mathsf{List}_n\ X, \\ \mathsf{cons} &: (\mathbf{par} \shortmid X : \mathcal{U}) \to (\mathbf{irr} \shortmid m\ n : \mathbb{N}) \to \\ &\qquad (\mathbf{irr} \shortmid m < n) \to X \to \mathsf{List}_m\ X \to \mathsf{List}_n\ X. \qquad (2) \end{aligned}$$

All size bounds and proofs about them are marked as irrelevant (**irr**), because they can and should be ignored during equality-checking. Indeed, we want the following lists (where _ replaces arbitrary inequality proofs) to be judgementally equal as they are both annotated versions of the list $(a :: [])$:

$$\mathsf{cons}\ A\ 2\ 5\ \_\ a\ (\mathsf{nil}\ A\ 2\ \_) \equiv \mathsf{cons}\ A\ 3\ 5\ \_\ a\ (\mathsf{nil}\ A\ 3\ \_). \qquad (3)$$

Example 1.3 (Ad hoc polymorphism)**.** The law of excluded middle

$$\mathsf{lem} : (\mathbf{hoc} \shortmid X : \mathcal{U}) \to X \uplus (X \to \mathsf{Empty}) \qquad (4)$$

breaks parametricity. Indeed, if $\mathsf{lem}\ X$ were parametric in X, then it would be a parametricity theorem that $\mathsf{lem}\ \mathsf{Unit}$ and $\mathsf{lem}\ \mathsf{Empty}$ either both give an inhabitant, or both prove emptiness. Hence, to avoid inconsistency, we need to mark lem as ad hoc polymorphic. A sufficiently syntactic model may also justify an ad hoc polymorphic typecase operator.

Modality of the codomain In a modal dependent type system, we can consider the function type $(\mu \shortmid x : A) \to B\ x$ of functions f of modality μ that map arguments $a : A$ to $f\ a : B\ a$. Here, B is a function from A to a universe (i.e. a type of types) $\mathcal{U}$, mapping $a : A$ to the type $B\ a : \mathcal{U}$. An important question is: how well-behaved does B have to be before it is sensible to even ask μ-modal behaviour of functions $f : (\mu \mid x : A) \to B\ x$? Different authors have answered this question differently.

Example 1.4 (Shape-irrelevance)**.** Example 1.2 introduced the function nil which takes an irrelevant size bound n. Pfenning [27] and Reed [29] do not support this function: they require the codomain $B :\equiv \lambda n.((\mathbf{irr} \mid 0 < n) \to \mathsf{List}_n\ X)$ of the irrelevant function $\mathsf{nil}\ X : (\mathbf{irr} \mid n : \mathbb{N}) \to B\ n$ to be also an irrelevant function. However, $\mathsf{List}_n\ X$ cannot be irrelevant in n; otherwise, list types of different size bound would be judgementally equal, allowing us to convert lists between types of different bounds, rendering the bounds meaningless.

Mishra-Linger and Sheard [23] and Barras and Bernardo [7] do not impose any restrictions on the behaviour of B, however Abel and Scherer [3, example 2.8] show that this is problematic in the presence of type-aware computation (such as η-expansion for record types or the unit type): if η-expansion of $f\ a$ is triggered by its type $B\ a$, which in turn depends on a, then computation may depend on a even if f uses its argument only in type annotations. Hence, a cannot be erased from $f\ a$ at type-checking time.

Abel introduced the idea that the argument x may appear in $B\ x$ but should be irrelevant to the *shape* of $B\ x$, e.g. x does not get to decide whether or not $B\ x$ is a record type, hence it cannot trigger η-expansion of records. **Shape-irrelevance** is currently only formally understood in the context of sized types [4] (i.e. x has to be a size index), but is available in general in the Agda programming language.

The type $\mathsf{List}_n\ X$ is shape-irrelevant in n: regardless of n, it is always a type of lists with constructors nil and cons. Similarly, $(0 < n)$ is always a type of inequality proofs. Hence, $B : (\mathbf{shi} \mid n : \mathbb{N}) \to \mathcal{U}$ is shape9-irrelevant and qualifies as a codomain for nil X.

Example 1.5 (Continuity)**.** The codomain of the if operator from Example 1.1 w.r.t. the argument X, is the function $B :\equiv \lambda X.(\mathsf{Bool} \to X \to X \to X) : \mathcal{U} \to \mathcal{U}$. This non-dependent function B cannot be parametric, because it is a parametricity theorem that parametric non-dependent functions are all constant (in System F, these have type $\forall X.T$, where T does not depend on X). So we should not require the codomain of a parametric function type to depend parametrically on its argument.

However, it does not make sense to consider parametric functions $(\mathbf{par} \mid X : \mathcal{U}) \to T(X)$ if T is defined using a typecase operator: it is then unclear what theorems parametricity should entail. This suggests that we need a modality in between parametricity and ad hoc polymorphism; which we have called **continuity** [26].

Modalities interact Suppose we want to combine irrelevance, shape-irrelevance, parametricity, continuity and ad hoc polymorphism in a single type system. This raises new questions: if f has modality μ, and g has modality ν, then what is the best that can be said about $g \circ f$ — do we need a new modality or can we fall back to one of the existing ones? What about functions whose behaviour satisfies both μ and ν?

Example 1.6 (Composition of modal functions)**.** We expect the following terms of type $\mathsf{List}_5\ A$, which are both annotated versions of $a :: (if\ b\ []\ (a' :: as))$, to be equal:

$$\mathsf{cons}\ A\ 4\ 7\ \_\ a\ (if\ (\mathsf{List}_4\ A)\ b\ (\mathsf{nil}\ A\ 4\ \_)\ (\mathsf{cons}\ A\ 3\ 4\ \_\ a'\ as)),$$
$$\mathsf{cons}\ A\ 5\ 7\ \_\ a\ (if\ (\mathsf{List}_5\ A)\ b\ (\mathsf{nil}\ A\ 5\ \_)\ (\mathsf{cons}\ A\ 3\ 5\ \_\ a'\ as)).$$

Indeed, by the parametricity theorem in Example 1.1, we can distribute the outer cons over the then- and else-clauses, after which equality becomes clear.

As it stands, though, the terms differ not only in irrelevant arguments to cons and nil, but also in the size-bound on List. This size-bound is used shape-irrelevantly by List, and $\mathsf{List}_n\ A$ is subsequently used parametrically by if. This suggests that a shape-irrelevant function, post-composed with a parametric one, should yield an irrelevant function: $\mathbf{par} \circ \mathbf{shi} = \mathbf{irr}$.

Pertinence of modalities for a given (co)domain In our previous work [26] we show that it is sound to allow dependent pattern matching when constructing parametric functions $(\mathbf{par} \mid n : \mathbb{N}) \to B\ n$ from the naturals. This raises the question whether there is any point in distinguishing between parametric, continuous and ad hoc functions when the domain is $\mathbb{N}$. Meanwhile, others [6, 15, 31] have shown that all continuous (non-ad-hoc) functions to a small type are parametric, e.g. any continuous function $(X : \mathcal{U}_0) \to X \to X$ is automatically parametric. So it seems that certain modalities become synonymous if the (co)domain is sufficiently small.

A unified framework The above exposition raises a number of questions. We have mentioned five different modalities that all have a clear use case; are these all we need or will there be more? What modalities are synonymous under what circumstances? How do we compute the modality of composed functions? How do we compute the required modality for the codomain of a modal dependent function type? How can we justify that equality checking ignores irrelevant parts? How can we prove parametricity theorems for parametric functions? Some of these questions have been answered for specific modalities or under restricted circumstances [3, 4, 6–8, 15, 17, 22–24, 26, 27, 29, 31], but to the best of our knowledge, there are no conclusive answers in the literature in the presence of all the aforementioned modalities. We provide a general theory of modalities that deal with relations and equality, including the five we mentioned, that answers all of the above questions.

The core idea is that we describe function behaviour by stating how a function affects the degree of relatedness of objects that it is applied to. In our previous work [26] we already classified functions by how they act on related inputs, with parametric functions sending them to equal outputs, continuous functions sending them to related outputs, and ad hoc functions (which there we called pointwise) sending them to potentially unrelated outputs.

Multiple relations Our current framework is based on the observation that 'related' is an insufficiently specific concept. For example, in small types such as Bool, $\mathbb{N}$ and List_n Bool, the only truly interesting relation is equality. Equality has a heterogeneous generalization which we call 0-relatedness (informally denoted $\frown_0$), e.g. a list of type $\mathsf{List}_4\ A$ is 0-related to a list of type $\mathsf{List}_6\ A$ if they have equal length and contents. This is the most obvious notion of 0-relatedness (henceforth: 0-relation) between $\mathsf{List}_4\ A$ and $\mathsf{List}_6\ A$ and also the one that our type system will consider by default.

There is no canonical 0-relation between the types $\mathbb{N}$ and Unit, but every object $\mathbb{N} \to \mathsf{Unit} \to \mathcal{U}$ gives rise to a non-canonical 0-relation R. Defining B as in Example 1.5, this gives rise to a 0-relation $B\ R$ (informal notation) between $B\ \mathbb{N}$ and B Unit, as in Reynolds' original semantics of parametricity [30]. Regardless of the choice of R, the object $if\ \mathbb{N} : B\ \mathbb{N}$ will be 0-related to if Unit : B Unit (informally denoted $if\ \mathbb{N} \frown_0^{B\,R} if$ Unit), since both functions essentially apply the same algorithm. It is a property of our type system that the default 0-relation between a type A and itself, coincides with equality, e.g. if $a, b : A$, then $a \frown_0^A b$ means that a equals b.

hoc	=	$\langle 0, 0, \dots, 0\rangle$	:	m	$\to$	n	ad hoc polym.
str	=	$\langle 0, 0, 1, \dots, n\rangle$	:	n	$\to$	$n+1$	structurality
con	=	$\langle 0, 1, \dots, n\rangle$	:	n	$\to$	n	continuity
par	=	$\langle 1, 2, \dots, n\rangle$	:	n	$\to$	$n-1$	parametricity
shi	=	$\langle 0, \top, \dots, \top\rangle$	:	m	$\to$	$n \geq 0$	shape-irr.
irr	=	$\langle \top, \top, \dots, \top\rangle$	:	m	$\to$	n	irrelevance
up	=	$\langle 0, \dots, n, n\rangle$	:	$n \geq 0$	$\to$	$n+1$	upw. cont.
dn	=	$\langle 0, \dots, n-2, n\rangle$	:	$n \geq 1$	$\to$	$n-1$	downw. cont.

Figure 1. Some important modalities

Types themselves can also be 0-related, e.g. $\mathsf{List}_{2+3}\ A \frown_0^{\mathcal{U}_0} \mathsf{List}_5\ A$. However, this is not the only interesting relation that we can consider on types. Consider the types $\mathsf{List}_2\ A$ and $\mathsf{List}_4\ A$. These types are not 0-related (equal), but there is a notion of 0-relatedness of their elements. We say that they are 1-related ($\mathsf{List}_2\ A \frown_1^{\mathcal{U}_0} \mathsf{List}_4\ A$). This is a proof-relevant property and the proofs $R : S \frown_1^{\mathcal{U}_0} T$ correspond up to isomorphism to relations $S \to T \to \mathcal{U}_0$ and are precisely what gives meaning to $(s : S) \frown_0^{R} (t : T)$. So we can prove $\mathbb{N} \frown_1^{\mathcal{U}_0} \mathsf{Unit}$ in many ways. We can also prove $\mathsf{List}_2\ A \frown_1^{\mathcal{U}_0} \mathsf{List}_4\ A$ in many ways, but there is one default way to prove it, namely by giving the default 0-relation between $\mathsf{List}_2\ A$ and $\mathsf{List}_4\ A$. As such, the type-checker will acknowledge without proof that $\mathsf{List}_2\ A \frown_1^{\mathcal{U}_0} \mathsf{List}_4\ A$, whereas it will not do so for $\mathbb{N} \frown_1^{\mathcal{U}_0} \mathsf{Unit}$ — this is analogous to the distinction between propositional and judgemental equality. Note that 0-related types are also 1-related: indeed, if $S \frown_0^{\mathcal{U}_0} T$, then both types are equal and the equality relation on S is a 0-relation between them, proving $S \frown_1^{\mathcal{U}_0} T$. In general, 0-relatedness implies 1-relatedness in the sense that proofs of the former map to proofs of the latter.

The notion of 1-relatedness extends to other large types. For example, we can define a type of monoids Mon such that two monoids M and N are 1-related ($R : M \frown_1^{\mathsf{Mon}} N$) if their underlying types $\underline{M}$ and $\underline{N}$ are 1-related ($\underline{R} : \underline{M} \frown_1^{\mathcal{U}_0} \underline{N}$) and their operations (multiplication and neutral element) are 0-related according to $\underline{R}$.

We can define a non-canonical 1-relation V between Grp and Mon by saying that a group G is 1-related to a monoid M (denoted $G \frown_1^{V} M$) if the underlying monoid N of G is ($N \frown_1^{\mathsf{Mon}} M$). This is expressed as $V : \mathsf{Grp} \frown_2^{\mathcal{U}_1} \mathsf{Mon}$, i.e. these types of algebras are 2-related as proven by giving the 1-relation V.

A modality μ is now a function ($i \cdot \mu \mapsfrom i$) mapping degrees of relatedness of the codomain to degrees of relatedness of the domain. It expresses what degree of relatedness $x \frown_{i \cdot \mu}^{S} y$ is needed in order for a μ-modal function $f : (\mu \mid S) \to T$ to map x and y to i-related objects $f\ x \frown_i^{T} f\ y$. The two most extreme modalities are now easy to define: irrelevant functions produce maximally related outputs even for unrelated inputs, so we set $i \cdot \mathbf{irr} = \top$ (where $\top$-relatedness is the trivial relation 'true') and ad hoc functions produce related outputs only for equal inputs, so we set $i \cdot \mathbf{hoc} = 0$.

Contributions We present (§3) and have proven soundness (§5, [25]) of a type system in which every dependency is annotated with a modality (§2.2) that describes its relational behaviour in a fine-grained way. The available modalities include parametricity [30], irrelevance [3, 7, 23, 27, 29], shape-irrelevance [4], ad hoc polymorphism, and continuity in the sense of our previous work [26]; and we explain each of these modalities as a certain action on relations (§2.4). We answer the question which of the available modalities are synonymous under what circumstances (§2.6) and make the type system aware of this synonymity using the concept of depth (§2.1). Depth also bridges the gap between viewing irrelevance as a property of types (as in Coq) or functions (as in Agda) (§6). We explain how the available modalities compose (§2.2), and give and explain the relation between the modalities through which a term and its type depend on the same variable (§2.3).

We support and justify type-checking time erasure of irrelevant subterms — even when irrelevance occurs as a composite of other modalities — using a fine-grained erasure system with a family erasure functions (§3.2). Using internal parametricity operators [8, 24, 26], we allow users to construct 'cheap' proof terms for 'free' theorems [33] (§3.3).

We give the first account of shape-irrelevance in which all function types, and not just those with the special domain Size, can be annotated shape-irrelevant (§2.3). We support sized (co)-inductive types and index them with shape-irrelevant natural numbers from the inductive type $\mathbb{N}$ (§4). Moreover, we support shape-irrelevant universe polymorphism (T-UNI, Fig. 3) and hence allow the erasure of universe levels, like size bounds, when they are irrelevant. Some of these features already experimentally supported by Agda.

We introduce a novel modality that we call structurality (§2.4). It expresses how algebras depend on their structure and thus allows the correct notion of parametric quantification over a type of algebras. It also replaces the ad hoc (pointwise) modality that our previous work [26] had to use in its internal parametricity operators, and unlike ad hoc polymorphism does not get in the way of iterated parametricity. This means that we are first to present a type system that combines fully iterable (i.e. self-applicable) internal parametricity operators and the identity extension lemma.

Using a special dependent if-expression in which the conditional is irrelevant, we can implement a notion of intersection and union types as irrelevant quantification over the booleans (§4).

Overview In §2, we introduce the concept of depth of a type, define the collection of modalities that we will use, and the necessary operations on them. In §3, we introduce the type system: we list core typing rules, explain the erasure system and briefly discuss the internal parametricity operators. In §4, we consider some applications: Church encoding and algebra, intersections and unions, and sized types. In §5, we sketch the denotational model and the semantics of erasure. In §6, we discuss related and future work.

2 Depths and Modalities

In this section, we define the collection of all available modalities, and investigate what modality operations a dependent type system requires and how we can compute them.

2.1 Types, Depth and Relations

An important prerequisite is the concept of **depth** of a type: a type of depth n can be thought of as being equipped with $n+1$ proof-relevant relations, numbered 0 through n, plus the trivially satisfied relation $\top$. As in Reynolds' original semantics of parametricity [30], these relations arise from an intricate interplay between user-made choices and the mechanics of the framework. Here, we give a conceptual discussion and hide higher-dimensional (cubical) relational structure; in §3.3 we sketch how the relations are exposed internally using parametricity operators and in §5 we sketch the denotational model. Conceptually, a depth n type $\Gamma \vdash_n A\ \mathbf{type}$ gives us, for every assignment γ of the variables in Γ, a set of values $A[\gamma]$. Moreover, for every $i \in \{0, \dots, n\}$, we get a proof-relevant binary relation on $A[\gamma]$ which we call the **homogeneous i-relation** and informally denote as $\frown_i^{A[\gamma]}$. These relations are reflexive, meaning that $A[\gamma]$ comes equipped with functions that map values $a : A[\gamma]$ to reflexivity proofs of $a : a \frown_i^{A[\gamma]} a$ (denoted with the same symbol). The homogeneous 0-relation $\frown_0^{A[\gamma]}$ is the strictest one and is always the equality relation — this generalizes Reynolds' identity extension lemma. As i increases, $\frown_i^{A[\gamma]}$ becomes more liberal, meaning more precisely that $A[\gamma]$ comes equipped with functions that map proofs $r : x \frown_i^{A[\gamma]} y$ to proofs $r : x \frown_j^{A[\gamma]} y$ (denoted with the same symbol) if $j \geq i$, such that all diagrams commute. Finally, $\frown_\top^{A[\gamma]}$ is always true and can always be proven in only one way. Thus, we place an order relation $0 \leq \ldots \leq n \leq \top$ on the degrees of relatedness available in a type of depth n.

Example 2.1. The type $\Gamma \vdash_0 \mathsf{Bool}\,\textbf{type}$ is a weakening of a closed type and hence its meaning is independent of the assignment γ. It represents a 2-element set Bool equipped with a homogeneous 0-relation $\frown_0^{\mathsf{Bool}}$ that is equality. The type $\Gamma \vdash_n \mathsf{List}\,A\,\textbf{type}$ yields, for every assignment γ, the set $(\mathsf{List}\,A)[\gamma]$ of lists over $A[\gamma]$, equipped with the homogeneous i-relation that relates $as \frown_i^{(\mathsf{List}\,A)[\gamma]} bs$ if as and bs have equal length and are componentwise related by $\frown_i^{A[\gamma]}$. In particular, if $\frown_0^{A[\gamma]}$ is the equality relation, then so is $\frown_0^{(\mathsf{List}\,A)[\gamma]}$. This exemplifies that the 0-relation is defined by recursion on the type, while we prove inductively that it is the equality relation.

But there is more to the semantics of $\Gamma \vdash_n A\,\textbf{type}$. For any *two* assignments γ, γ', any proof $\gamma^* : \gamma \frown_i^{\Gamma} \gamma'$ that they are i-related (meaning that the assignments for each μ-modal variable are $(i \cdot \mu)$-related) and any $j \geq i$, we obtain a **heterogeneous *j*-relation** $\frown_j^{A[\gamma^*]}$ between elements of $A[\gamma]$ and $A[\gamma']$. These heterogeneous relations need not be reflexive; indeed, they do not even relate values from the same set. Moreover, the heterogeneous 0-relation may be any proof-relevant relation and need not be the equality relation. As in the homogeneous case, the heterogeneous relations become more liberal as i increases, i.e. $A[\gamma^*]$ comes with functions that map proofs of $a \frown_j^{A[\gamma^*]} a'$ to proofs of $a \frown_k^{A[\gamma^*]} a'$ if $k \geq j$.

Example 2.2. The type $\Gamma \vdash_0 \mathsf{Bool}\,\textbf{type}$ is closed and has as its heterogeneous relation $\frown_0^{\mathsf{Bool}[\gamma^*]}$ simply the homogeneous relation $\frown_0^{\mathsf{Bool}}$. The type $\Gamma \vdash_n \mathsf{List}\,A\,\textbf{type}$ relates $as \frown_j^{(\mathsf{List}\,A)[\gamma^*]} as'$ heterogeneously if $as : (\mathsf{List}\,A)[\gamma]$ and $as' : (\mathsf{List}\,A)[\gamma']$ have equal length and are componentwise related by $\frown_j^{A[\gamma^*]}$.

In the introduction, we pretended that the depth of a type would be derivable from its universe level, e.g. that all small types would have depth 0. However, in practice it will be useful to treat universe level and depth as orthogonal properties of a type. Universes $\mathcal{U}_\ell^n$ will be annotated with a level ℓ to prevent impredicativity, and a depth n which is the depth of the types they contain. We have $\mathcal{U}_\ell^n : \mathcal{U}_{\ell'}^{n+1}$ if $\ell < \ell'$. This double indexing is not unlike the way the homotopy level is treated in HoTT [32] and allows us to consider predicative depth-truncation (using Box, see Remark 2.8) and small higher inductive types (as a future research direction).

Example 2.3. The universe $\Gamma \vdash_{n+1} \mathcal{U}_\ell^n\,\textbf{type}$, like Bool, is a closed type. Its values are the closed types $\vdash_n T\,\textbf{type}$ of depth n and level ℓ. As is required by identity extension, $\frown_0^{\mathcal{U}_\ell^n}$ means equality. A proof of homogeneous $(i+1)$-relatedness $A \frown_{i+1}^{\mathcal{U}_\ell^n} B$ wraps up ever more liberal heterogeneous relations between A and B, numbered i through n. Reflexivity of $\frown_{i+1}^{\mathcal{U}_\ell^n}$ is proven by using the homogeneous relations as heterogeneous relations. A variable $\Gamma \vdash_n X\,\textbf{type}$ will only be well-typed if Γ provides a variable $X : \mathcal{U}_\ell^n$ for parametric use, to which γ will assign a value $X[\gamma] : \mathcal{U}_\ell^n$ which contains everything needed to interpret $X[\gamma]$ as a type. A proof $\gamma^* : \gamma \frown_i^{\Gamma} \gamma'$ assigns to a parametric variable X a proof $X[\gamma^*] : X[\gamma] \frown_{i \cdot \mathbf{par}}^{\mathcal{U}_\ell^n} X[\gamma']$. Since $i \cdot \mathbf{par} = i + 1$ (as we will find in §2.4), the object $X[\gamma^*]$ gives us exactly what we need: heterogeneous relations numbered i through n. Internal parametricity operators (§3.3) give the user control over what relations they put in $X[\gamma^*]$.

A special case are types of depth -1: they are solely equipped with the trivial relation $\frown_\top$, which then also takes the role of the 0-relation and therefore expresses equality. In other words, in a -1-type, equality is trivially true, i.e. -1-types are proof-irrelevant propositions. When dealing with -1-types, we will allow ourselves to use 0 and $\top$ interchangeably, reducing the need for special cases.

2.2 Modalities

Let A be a type of depth m, and $B\,x$ a type of depth n for each $x : A$. Then the behaviour of a dependent function $f : (\mu \mid A) \to B$ is described by a modality $\mu : m \to n$ which gives us the minimal degree of relatedness $i \cdot \mu$ that is required of x and y in order to conclude $f\,x \frown_i f\,y$. Clearly, a stronger degree of relatedness of x and y is required in order to conclude a stronger degree of relatedness of $f\,x$ and $f\,y$. Also clearly, $\top \cdot \mu = \top$. Thus, we say:

Definition 2.4. A **depth** is an integer $n \geq -1$. A **modality** $\mu : m \to n$ from depth m to depth n is a monotonically increasing function $\mu : \{0 \leq \ldots \leq n\} \to \{0 \leq \ldots \leq m \leq \top\} : i \mapsto i \cdot \mu$, which we also denote in vector notation as $\mu = \langle 0 \cdot \mu, \ldots, n \cdot \mu \rangle$. By convention, we write $\top \cdot \mu = \top$.

Example 2.5. The precise signature of **hoc** and **irr** was already derived in the introduction and can be found in Fig. 1. Notice how both modalities exist for arbitrary domain and codomain, but coincide when either has depth -1 (Fig. 2).

Consider a function $f : (\mu \mid A) \to B$ of modality $\mu : m \to n$ and a function $g : (\nu \mid B) \to C$ of modality $\nu : n \to p$. Then how do we compute the modality $\nu \circ \mu$ of $g \circ f$? Clearly, $g\,(f\,x) \frown_i g\,(f\,y)$ requires $f\,x \frown_{i \cdot \nu} f\,y$, which in turn requires $x \frown_{(i \cdot \nu) \cdot \mu} y$. Thus, we have $i \cdot (\nu \circ \mu) = (i \cdot \nu) \cdot \mu$.

Meanwhile, the identity function $\mathsf{id} : X \to X$ has a modality **con** for which $i \cdot \mathbf{con} = i$. We consider **con** the default modality (hence we omit its annotation) and call it **continuity** (Fig. 1; in §2.4 we argue that this is also the modality encountered in Example 1.5). Notice how continuity coincides with ad hoc polymorphism on depth 0 types like Bool or $\mathbb{N}$ (Fig. 2).

Finally, we define a partial order relation on modalities: we say that $\mu \leq \nu$ when all ν-modal functions are also μ-modal (the direction of the ordering arises from viewing μ and ν as operations applied to a function's domain). This means that μ concludes $f\,x \frown_i f\,y$ under stronger assumptions than ν does, i.e. $i \cdot \mu \leq i \cdot \nu$ for all i. For example, an irrelevant function also satisfies every other modality (**irr** is maximal), and all functions can be seen as ad hoc polymorphic (**hoc** is minimal).

Example 2.6. If a function satisfies both modalities μ and ν, then it satisfies their least upper bound $\mu \sqcup \nu$, for which $i \cdot (\mu \sqcup \nu) = \max\{i \cdot \mu, i \cdot \nu\}$.

Proposition 2.7. *Depths and modalities form a poset-enriched category, i.e. a category whose Hom-sets are partially ordered and whose composition operation is monotonically increasing.* □

Remark 2.8. We can have a data type $\mathsf{Box}^\mu A$ with a single μ-modal constructor $\mathsf{box}^\mu : (\mu \mid A) \to \mathsf{Box}^\mu A$. Then $\mathsf{box}^\mu x \frown_i \mathsf{box}^\mu y$ can and can only be proven from $x \frown_{i \cdot \mu} y$, and functions $(\mu \mid A) \to B$ can be thought of as continuous functions from domain $\mathsf{Box}^\mu A$. This shows that we can think of modalities as operations that are applied to the domain of a function type.

2.3 Modality of the Codomain

We come back to the question of what modality ν is required of B in order to consider μ-modal functions $f : (\mu \mid x : A) \to B\,x$. Assume that A has depth m and $B\,x$ always has depth n, i.e. $\mu : m \to n$. Then B has type $(\nu \mid A) \to \mathcal{U}_\ell^n$ for some universe level ℓ. As mentioned in §2.1, in order to consider $f\,x \frown_i f\,y$, we need $(i+1)$-relatedness of their types, i.e. $B\,x \frown_{i+1} B\,y$. We know that μ asserts the former when $x \frown_{i\cdot\mu} f\,y$, so ν should assert the latter under the same circumstances, i.e. $(i+1)\cdot\nu = i\cdot\mu$. Equality of types $B\,x \frown_0 B\,y$ is only required when the arguments $x \frown_0 y$ are equal, so we make the most liberal choice and set $0\cdot\nu = 0$. Then $\nu = \langle 0, 0\cdot\mu, \dots, n\cdot\mu\rangle : m \to n+1$, i.e. the modality of a function's codomain is obtained by inserting a 0 in front of the modality of the function itself. Applying this reasoning to the signature of irrelevance, we obtain the signature of shape-irrelevance (Fig. 1), which was indeed conceived as the modality of an irrelevant function's codomain. This signature in turn shows that we can ignore shape-irrelevant subterms when checking for 1-relatedness. We also find that ad hoc functions can have ad hoc codomain.

2.4 Modalities Worthy of a Name

In this section, we take a closer look at the type signature and meaning of the modalities mentioned in the introduction. Irrelevance (§1), shape-irrelevance (§2.3) and ad hoc polymorphism (§1) have already been treated.

Continuity In §2.2, we defined continuity $\mathbf{con} : n \to n$ as the modality of the identity function, thus obtaining its signature. Here, we show that the continuous pair and function type formers are continuous, confirming what we said in Example 1.5. We have $0\cdot\mathbf{con} = 0$, and indeed if $X \frown_0^{\mathcal{U}_\ell^n} X'$ and $Y \frown_0^{\mathcal{U}_\ell^n} Y'$, we will have $X \times Y \frown_0^{\mathcal{U}_\ell^n} X' \times Y'$ and $X \to Y \frown_0^{\mathcal{U}_\ell^n} X' \to Y'$ as 0-relatedness means equality. Next, we show that the type formers respect the fact that $(i+1)\cdot\mathbf{con} = i+1$. If $X^* : X \frown_{i+1}^{\mathcal{U}_\ell^n} X'$ and $Y^* : Y \frown_{i+1}^{\mathcal{U}_\ell^n} Y'$, then we can prove $X^* \times Y^* : X \times Y \frown_{i+1}^{\mathcal{U}_\ell^n} X' \times Y'$ by taking $(x^*, y^*) : (x, y) \frown_i^{X^* \times Y^*} (x', y')$ to mean that $x^* : x \frown_i^{X^*} x'$ and $y^* : y \frown_i^{Y^*} y'$. Meanwhile, we would naively prove $X^* \to Y^* : X \to Y \frown_{i+1}^{\mathcal{U}_\ell^n} X' \to Y'$ by taking a proof $f^* : f \frown_i^{X^* \to Y^*} f'$ to be a function that maps proofs $x^* : x \frown_i^{X^*} x'$ to proofs $f^*\,x^* : f\,x \frown_i^{Y^*} f'\,x'$. However, with this definition, the relations for $X^* \to Y^*$ do not become more liberal as i increases. Thus, we require that for all $j \geq i$, the proof f^* maps proofs $x^* : x \frown_j^{X^*} x'$ to proofs $f^*\,x^* : f\,x \frown_j^{Y^*} f'\,x'$ in a coherent way.

Parametricity We want to continue Example 1.1 and investigate how *if* X depends on X. However, in order to derive the general signature of parametricity, we need to consider types X of arbitrary depth. Because we have not yet explained how to promote Bool to depths greater than zero, we will instead consider $t\,X :\equiv \lambda x.\lambda x'.x : T\,X :\equiv X \to X \to X$, which corresponds to the operation 'if true' on X. Under what circumstances can we conclude that $t\,X \frown_i^{T\,R} t\,Y$? Remember that this means that for all $j \geq i$, we have $t\,X\,x\,x' \frown_j^R t\,Y\,y\,y'$ whenever $x \frown_j^R y$ and $x' \frown_j^R y'$. But that statement is trivially true (as $t\,X\,x\,x' \equiv x$ and $t\,Y\,y\,y' \equiv y$) provided that it can be stated, i.e. provided that we can consider i-relatedness of elements of X and Y. So we need $R : X \frown_{i+1} Y$, and conclude that $i\cdot\mathbf{par} = i+1$ (Fig. 1). Inserting a 0 in front of the signature as per §2.3, we see that parametric functions indeed require a continuous codomain as was claimed in Example 1.5. Observe also that $\mathbf{par} \circ \mathbf{shi} = \mathbf{irr}$, as was conjectured in Example 1.6.

Structurality Where parametricity expresses how types can be used at the term level, the structural modality ($\mathbf{str}$, Fig. 1) expresses how terms can be used at the type level. For example, the codomain $B\,x$ of a continuous function type $(\mathbf{con} \mid x : A) \to B\,x$ depends structurally on x. In particular, structurality expresses how algebras depend on their structure. Suppose we want to define a type Mon of (lawless) monoids M, i.e. types $\underline{M}$ equipped with a nullary operation $e_M : \underline{M}$ and a binary operation $*_M : T\,\underline{M} :\equiv \underline{M} \to \underline{M} \to \underline{M}$. Monoids M and N should be 0-related, i.e. equal, when all parts are equal, i.e. $\underline{M} \frown_0^{\mathcal{U}_\ell^0} \underline{N}$, $e_M \frown_0^{\underline{M}} e_N$ and $*_M \frown_0^{T\,\underline{M}} *_N$. Meanwhile, a sensible notion of 1-relatedness $R : M \frown_1 N$ would be that the underlying sets are 1-related ($\underline{R} : \underline{M} \frown_1 \underline{N}$) and that their structure is 0-related according to $\underline{R}$. This shows that monoids depend on their structure with modality $\mathbf{str} = \langle 0, 0\rangle : 0 \to 1$, so we can define

$$\mathsf{Mon} :\equiv (X : \mathcal{U}_\ell^0) \times \mathsf{Box}^{\mathbf{str}}\,(X \times (X \to X \to X)). \tag{5}$$

Notably, structurality is the sole modality that is right inverse to parametricity: $\mathbf{par} \circ \mathbf{str} = \mathbf{con}$. This means that we can uncurry functions that take both parametric and continuous arguments. For example, the type of canonical elements that can be constructed in the same way in any monoid is

$$(\mathbf{par} \mid (X, \mathsf{box}^{\mathbf{str}}\,(e, *)) : \mathsf{Mon}) \to X \tag{6}$$

which can be curried to

$$(\mathbf{par} \mid X : \mathcal{U}_\ell^0) \to X \to (X \to X \to X) \to X. \tag{7}$$

This is the Church encoding of the type of binary tree shapes, which (up to predicativity issues) is the initial lawless monoid.

2.5 Left Division and Contramodalities

Consider modalities $\mu : m \to n$ and $\nu : n \to p$ and functions $f : (\mu \mid A) \to B$ and $g : (\nu \mid B) \to C$ and suppose we want to type-check $\lambda(\rho \mid x : A).g\,(f\,x) : (\rho \mid A) \to C$. Mathematically, it is clear that this should type-check if and only if $\rho \leq \nu \circ \mu$. However, the type-checker will go about this differently. First of all, it will put $(\rho \mid x : A)$ into the context. Then it observes that g needs an argument of type B with modality ν, so it will type-check $\nu \setminus \rho \mid x : A \vdash f\,x : B$, where $\nu \setminus \rho$ is some modality computed from ν and ρ. Type-checking now succeeds whenever $\nu \setminus \rho \leq \mu$, so this should be equivalent to the condition $\rho \leq \nu \circ \mu$ mentioned above. This means that the operation $\nu \setminus \sqcup : (m \to p) \to (m \to n)$ is left adjoint to $\nu \circ \sqcup : (m \to n) \to (m \to p)$, i.e. they form a Galois connection. It turns out that left division by ν can be computed as postcomposition with a left adjoint *contramodality* $\kappa \dashv \nu$:

Definition 2.9. A **contramodality** $\kappa : p \to n$ is a monotonically increasing function $\kappa : \{0 \leq \ldots \leq n\} \to \{\bot \leq 0 \leq \ldots \leq p\} : i \mapsto i\cdot\kappa$. **Composition** with a modality $\rho : m \to p$ yields another modality $\kappa \circ \rho : m \to n$ given by $i\cdot(\kappa \circ \rho) = (i\cdot\kappa)\cdot\rho$, where by convention $\bot\cdot\rho = 0$.

Proposition 2.10. *For every modality $\nu : n \to p$, there is a contramodality $\kappa : p \to n$ so that $\kappa \circ \sqcup : (m \to p) \to (m \to n)$ is left adjoint to $\nu \circ \sqcup : (m \to n) \to (m \to p)$. We write $\kappa \dashv \nu$ and can compute $\nu \setminus \rho$ as $\kappa \circ \rho$. Adjoint pairs $\kappa \dashv \nu$ are characterized by the fact that for all i, j (not $\bot$ or $\top$), we have $i \leq j\cdot\kappa \Leftrightarrow i\cdot\nu \leq j$.* □

Example 2.11. Parametricity is both a modality and a contramodality, and we have $\mathbf{par} \dashv \mathbf{str}$. Hence, if we want to define a monoid $(T, \mathsf{box}^{\mathbf{str}}(e, *))$ with a variable $(\mu \mid x : A)$ in the context, then e and $*$ are type-checked with $(\mathbf{par} \circ \mu \mid x : A)$ in the context.

Example 2.12. The left adjoint to parametricity is $\langle\bot, 0, \ldots, n\rangle$. Hence, $\mathbf{par} \setminus \mu$ has the signature of μ with a 0 inserted in front: it is the modality of the codomain of a μ-modal function. To see where

irr, shi, hoc	$=$	$\langle\top, \ldots, \top\rangle$	$:$	-1	$\to$	n
irr, hoc	$=$	$\langle\rangle$	$:$	m	$\to$	-1
hoc, dn $\circ$ str, con, par $\circ$ up	$=$	$\langle 0\rangle$	$:$	0	$\to$	0
irr, dn $\circ$ shi	$=$	$\langle\top\rangle$	$:$	0	$\to$	0
hoc	$=$	$\langle 0\rangle$	$:$	1	$\to$	0
dn^2 $\circ$ str, dn $\circ$ con, par	$=$	$\langle 1\rangle$	$:$	1	$\to$	0
dn $\circ$ shi, irr	$=$	$\langle\top\rangle$	$:$	1	$\to$	0
hoc, str, con $\circ$ up, par $\circ$ up^2	$=$	$\langle 0, 0\rangle$	$:$	0	$\to$	1
shi	$=$	$\langle 0, \top\rangle$	$:$	0	$\to$	1
irr	$=$	$\langle\top, \top\rangle$	$:$	0	$\to$	1
hoc	$=$	$\langle 0, 0\rangle$	$:$	1	$\to$	1
dn $\circ$ str, con	$=$	$\langle 0, 1\rangle$	$:$	1	$\to$	1
shi	$=$	$\langle 0, \top\rangle$	$:$	1	$\to$	1
par $\circ$ up	$=$	$\langle 1, 1\rangle$	$:$	1	$\to$	1
shi $\sqcup$ (par $\circ$ up)	$=$	$\langle 1, \top\rangle$	$:$	1	$\to$	1
irr	$=$	$\langle\top, \top\rangle$	$:$	1	$\to$	1

Figure 2. All modalities $\mu : m \to n$ with $m, n \leq 1$.

this comes from, consider the type ascription operator (polymorphic identity function) $\_ \ni \_ : (\textbf{par} \mid X : \mathcal{U}^n_\ell) \to X \to X$. When we check the term $\lambda(\mu \mid x : A).(B\,x \ni f\,x)$, the type ascription $B\,x$ is a parametric argument and will be checked against $(\textbf{par} \setminus \mu \mid x : A)$.

Example 2.13. The left adjoint to irrelevance is $\langle\bot, \ldots, \bot\rangle$. Hence, $\textbf{irr} \setminus \mu = \textbf{hoc}$ for all μ.

2.6 Depth as information

The concepts treated so far will not let us form $\mathsf{Bool} \times \mathcal{U}^0_\ell$, because Bool has depth 0 and $\mathcal{U}^0_\ell$ has depth 1. We first need to increase the depth of Bool; this can be done by defining $\frown_1$ as a copy of $\frown_0$, which is represented syntactically by the type $\mathsf{Box}^{\textbf{up}}\ \mathsf{Bool}$, where **up** duplicates the weakest relation (Fig. 1). Then we can form $(\mathsf{Box}^{\textbf{up}}\ \mathsf{Bool}) \times \mathcal{U}^0_\ell$. In our denotational model, **up** is interpreted as a fully faithful functor, which shows that we can really see it as an embedding, i.e. that we can see depth n types as a special case of depth $n+1$ types.

In the introduction, we raised the question whether there is any difference between parametric, continuous and ad hoc functions from $\mathbb{N}$. In our previous work [26] we have used pattern matching (intuitively an ad hoc feature) to define a parametric identity function $(\textbf{par} \mid \mathbb{N}) \to \mathbb{N}$ on the naturals. However, currently we cannot consider parametric functions $\mathbb{N} \to \mathbb{N}$ as **par** decreases the depth, but we can consider $(\textbf{par} \mid \mathsf{Box}^{\textbf{up}}\ \mathbb{N}) \to \mathbb{N}$ or equivalently $(\textbf{par} \circ \textbf{up} \mid \mathbb{N}) \to \mathbb{N}$. So we can use precomposition with **up** to decrease the depth of a modality's domain, to make it match the function type of interest (Fig. 2). In this particular case, we find that $\textbf{par} \circ \textbf{up} = \langle 1\rangle \circ \langle 0, 0\rangle = \langle 0\rangle = \textbf{con} = \textbf{hoc} : 0 \to 0$. So indeed we need not distinguish between these modalities at depth 0.

We also mentioned that others [6, 15, 31] have shown that any continuous function to a small (depth 0) type — e.g. of type $(X : \mathcal{U}^0_\ell) \to X \to X$ — is parametric. However, we cannot consider continuous functions of this type, as continuity preserves the depth and $\mathcal{U}^0_\ell$ has depth 1. But we can consider $(X : \mathcal{U}^0_\ell) \to \mathsf{Box}^{\textbf{up}}(X \to X)$. In general, a function $(\mu \mid x : A) \to \mathsf{Box}^{\textbf{up}} B[x]$ is essentially the same as a function $(\textbf{up} \setminus \mu \mid x : A) \to B[x]$, because if we type-check $\lambda x.\mathsf{box}^{\textbf{up}}\ b[x]$, then we end up checking $\textbf{up} \setminus \mu \mid x : A \vdash_n b[x] : B[x]$. Proposition 2.10 entails that the contramodality $\textbf{dn} \dashv \textbf{up}$ is itself a modality which forgets the second weakest relation (Fig. 1), so $\textbf{up} \setminus \mu = \textbf{dn} \circ \mu$ and we can use postcomposition with **dn** to decrease the depth of a modality's codomain, to make it match the function type of interest. In this case, we find that $\textbf{dn} \circ \textbf{con} = \langle 1\rangle \circ \langle 0, 1\rangle = \langle 1\rangle = \textbf{par} : 1 \to 0$, i.e. parametricity and continuity coincide.

Repeated application of $\mathsf{Box}^{\textbf{up}}$ allows us to view all types as being equipped with arbitrarily many relations, and to think of finite depths as information about these relations: a type of depth n is a type of depth ω for which all relations $\frown_i$ are equal for $i \geq n$. A modality $\mu : m \to n$ can then be thought of as an equivalence class of modalities $\omega \to \omega$ that are synonymous when used to describe the behaviour of a function with domain of depth m and codomain of depth n. Parametricity is then always the equivalence class of $\langle 1, 2, 3, \ldots\rangle : \omega \to \omega$. In Fig. 2, we give an exhaustive list of all modalities of depth at most 1. We can see e.g. that ad hoc polymorphism, structurality, continuity and parametricity coincide for functions between depth 0 types $(0 \to 0)$.

There are two good reasons to allow only types of finite depth. First, the depth gives us valuable information about function behaviour, allowing us to consider only $\frac{(m+n+2)!}{(m+1)!\,(n+1)!}$ different modalities $m \to n$ instead of infinitely many modalities $\omega \to \omega$ (we leave the combinatorics as an exercise to the reader). Secondly, Proposition 2.10 breaks when considering depth ω, for example it is not clear what $\textbf{hoc} \setminus \textbf{con} : \omega \to \omega$ should be. For this reason, our denotational model also only considers types of finite depth.

3 The Type System

In this section, we present the type system a bit more formally. The core typing rules — which are essentially the typing rules of MLTT, annotated with modalities — are listed in Fig. 3 and explained in §3.1. We omit congruence and equivalence rules for judgemental equality. For space reasons, we also omit inductive types, including the identity type. A notion of judgemental relatedness is formalized using erasure functions in §3.2. In §3.3 we briefly discuss propositional relatedness and internal parametricity.

3.1 Annotating Martin-Löf Type Theory

Judgements The core type system makes use of the following judgement forms:

$\Gamma \vdash_n \textbf{Ctx}$	Γ is a context of depth n,
$\Gamma \vdash_n T\ \textbf{type}$	T is a type of depth n in context Γ,
$\Gamma \vdash_n t : T$	t is a term of type T of depth n,
$\Gamma \vdash_n T \equiv T'\ \textbf{type}$	T and T' are judgementally equal types,
$\Gamma \vdash_n t \equiv t' : T$	t and t' are judgementally equal terms.

The latter 4 judgement forms presuppose other judgements of the same depth, e.g. $\Gamma \vdash_n t : T$ can only be considered if $\Gamma \vdash_n T\ \textbf{type}$, which in turn presupposes $\Gamma \vdash_n \textbf{Ctx}$.

Contexts and variables **C-EM:** The empty context can be formed at any depth. **C-EXT:** A context of depth n can be extended with a variable x of a type T of depth m, provided that we make it available with a well-typed modality $\mu : m \to n$. The type T must exist in context $\mu \setminus \Gamma$, so that we can substitute x with $\mu \setminus \Gamma \vdash_n t : T$ later on. **T-VAR:** In order to use a variable $(\mu \mid x : T) \in \Gamma$, we require that the identity function satisfies modality μ, i.e. $\mu \leq \textbf{con}$.

Types and universes **T-UNI:** For each depth n and each *internal* natural number ℓ, we have a universe $\mathcal{U}^n_\ell$ of level $\mathsf{s}\ \ell$ (the internal successor of ℓ) and depth $n+1$. It lives in the universe $\mathcal{U}^{n+1}_{\mathsf{s}\ \ell}$ of depth $n+1$ types, which in turn has depth $n+2$. Hence, the judgement has depth $n+2$. Note that the shape-irrelevant level polymorphism makes universes of different level but equal depth 1-related, as $1 \cdot \textbf{shi} = \top$. **TY:** An element of the universe can be used as a type. This is the only introduction rule for the type judgement. By making it parametric, we ensure that the judgement $\textbf{par} \mid X : \mathcal{U}^n_\ell, \textbf{con} \mid x : X \vdash x : X$ can use X as a type, which corresponds to our expectation that the polymorphic identity function is parametric. **T-CUMUL:** We

$$\begin{array}{l}\mu \setminus () := (),\\ \mu \setminus (\Gamma, \nu \mid x : T) :=\\ \quad (\mu \setminus \Gamma), (\mu \setminus \nu) \mid x : T.\end{array} \qquad \frac{n \geq -1}{\vdash_n \mathbf{Ctx}}\textsc{c-em} \qquad \frac{\Gamma \vdash_n \mathbf{Ctx} \quad \mu : m \to n \quad \mu \setminus \Gamma \vdash_m T\,\mathbf{type}}{\Gamma, \mu \mid x : T \vdash_n \mathbf{Ctx}}\textsc{c-ext} \qquad \frac{\Gamma \vdash_n \mathbf{Ctx} \quad (\mu \mid x : T) \in \Gamma \quad \mu \leq \mathbf{con} : n \to n}{\Gamma \vdash_n x : T}\textsc{t-var} \qquad \frac{\Gamma \vdash_{n+2} \mathbf{Ctx} \quad n \geq -1 \quad \mathbf{shi} \setminus \Gamma \vdash_0 \ell : \mathbb{N}}{\Gamma \vdash_{n+2} \mathcal{U}^n_\ell : \mathcal{U}^{n+1}_{\mathrm{s}\,\ell}}\textsc{t-Uni}$$

$$\frac{\mathbf{par} \setminus \Gamma \vdash_{n+1} T : \mathcal{U}^n_\ell}{\Gamma \vdash_n T\,\mathbf{type}}\textsc{ty} \qquad \frac{\Gamma \vdash_{n+1} T : \mathcal{U}^n_\ell}{\Gamma \vdash_{n+1} T : \mathcal{U}^n_{\mathrm{s}\,\ell}}\textsc{t-cumul} \qquad \frac{\Gamma \vdash_n t : T \quad \Gamma \vdash_n T \equiv T'\,\mathbf{type}}{\Gamma \vdash_n t : T'}\textsc{t-conv} \qquad \frac{\Gamma, \mu \mid x : A \vdash_n b : B}{\Gamma \vdash_n \lambda(\mu \mid x : A).b : (\mu \mid x : A) \to B}\textsc{t-lam} \quad \text{where} \quad \lambda(\mu \mid x : A).f\langle\mu\rangle x \equiv f \qquad \frac{\Gamma \vdash_n t, t' : T \quad \lfloor t \rfloor_0 \mathrel{\mathring{=}} \lfloor t' \rfloor_0}{\Gamma \vdash_n t \equiv t' : T}\textsc{t-eq-erase}$$

$$\frac{\mu : m \to n \quad \bar{\mu} = \mathbf{par} \setminus (\mu \circ \mathbf{par}) : m+1 \to n+1 \quad \bar{\mu} \setminus \Gamma \vdash_{m+1} A : \mathcal{U}^m_\ell \quad \Gamma, (\mathbf{par} \setminus \mu) \mid x : A \vdash_{n+1} B : \mathcal{U}^n_\ell}{\Gamma \vdash_{n+1} (\mu \mid x : A) \to B : \mathcal{U}^n_\ell \qquad \Gamma \vdash_{n+1} (\mu \mid x : A) \times B : \mathcal{U}^n_\ell}\textsc{t-Pi, t-Sigma}$$

$$\frac{\mu : m \to n \quad \Gamma \vdash_n f : (\mu \mid x : A) \to B \quad \mu \setminus \Gamma \vdash_m a : A}{\Gamma \vdash_n f\langle\mu\rangle a : B[a/x]}\textsc{t-app} \quad \text{where} \quad (\lambda(\mu \mid x : A).b)\langle\mu\rangle a \equiv b[a/x] \qquad \frac{\mu : m \to n \quad \mu \setminus \Gamma \vdash_m a : A \quad \Gamma \vdash_n b : B[a/x]}{\Gamma \vdash (\mu \mid a, b) : (\mu \mid x : A) \times B}\textsc{t-pair}$$

$$\frac{\mu : m \to n \quad \nu : n \to p \quad \Gamma, \nu \mid z : (\mu \mid x : A) \times B \vdash_p C\,\mathbf{type} \quad \Gamma, \nu \circ \mu \mid x : A, \nu \mid y : B \vdash_p c : C[(\mu \mid x, y)/z] \quad \nu \setminus \Gamma \vdash_n t : (\mu \mid x : A) \times B}{\Gamma \vdash_p \mathsf{ind}^\nu_\times(z.C, x.y.c, t) : C[t/z] \quad \text{where} \quad \mathsf{ind}^\nu_\times(z.C, x.y.c, (\mu \mid a, b)) \equiv c[a/x, b/y]}\textsc{t-indpair}$$

Figure 3. Core typing rules

can coerce types to higher-level universes. **T-CONV:** Terms can be converted between equal types.

Functions and pairs **T-Pi, T-Sigma:** We have dependent function and dependent pair types for every modality μ. A μ-modal dependent function depends μ-modally on its argument (**T-LAM**, **T-APP**), whereas a μ-modal dependent pair depends μ-modally on its first component, and continuously on the second (**T-PAIR**). We use the notations $f\langle\mu\rangle a$ and $f\ a$ interchangeably, and denote nested pairs as a single tuple. The required modality of B in x was already established in §2.3. To get a feel for the modalities for the function and pair types in A and B, we can use the parametric type ascription operator from Example 2.12 and type-check

$$\lambda(\mu \mid x : A).(B\ x \ni f\langle\mu\rangle(A \ni x)), \qquad (\mu \mid A \ni a, B\ a \ni b) \tag{8}$$

to learn that these terms depend on A with modality $\mu \circ \mathbf{par}$ and on B parametrically. Their type lives behind the colon, which according to TY is itself a parametric position, so it should depend on A with modality $\mathbf{par} \setminus (\mu \circ \mathbf{par})$ and on B with modality $\mathbf{par} \setminus \mathbf{par} = \mathbf{con}$. **T-INDPAIR:** We can use a μ-modal pair ν-modally by using its first component $(\nu \circ \mu)$-modally and its second component ν-modally. We will sometimes abbreviate uses of T-INDPAIR by binding a pair pattern instead of a variable. If the contramodality $\kappa \dashv \mu$ is a modality (which is the case precisely when $0 \cdot \mu = 0$), then we can create a κ-modal first projection, since $\kappa \circ \mu \leq \mathbf{con}$. This allows us to form the type of the second projection and hence the second projection itself. In this case, we may instead take the projections as primitives, add a definitional η-rule for them, and implement $\mathsf{ind}^\nu_\times$ in terms of them. Typing rules for $\mathsf{Box}^\mu A$ are obtained by removing the second component, i.e. $\mathsf{Box}^\mu A \cong (\mu \mid A) \times \mathsf{Unit}$. Again, a κ-modal unbox function exists if $0 \cdot \mu = 0$.

Example 3.1. A contrived example is perhaps most illuminating. Let $\mu = \langle 0, 3, 7, \top \rangle : 7 \to 3$. The function type $(\mu \mid x : A) \to B\ x$ depends on A with modality $\bar{\mu} = \mathbf{par} \setminus (\mu \circ \mathbf{par}) = \langle 0, 1, 4, 8, \top \rangle : 8 \to 4$ and continuously on B. Meanwhile, $B\ x$ depends on x with modality $\mathbf{par} \setminus \mu = \langle 0, 0, 3, 7, \top \rangle : 7 \to 4$.

3.2 Erasure

Up to this point, the only inference rules that produce equality judgements, are the β- and η-laws in Fig. 3 and the (omitted) congruence and equivalence rules. However, $\beta\eta$-equality is an unsatisfying notion of equality in the current setting: we want to also identify terms that are equal up to their irrelevant subterms. A naive solution is to modify the congruence rules corresponding to inference rules with irrelevant premises. There, we would simply not require equality of the irrelevant subterms. For example, an equality rule for T-APP would say that $f\langle\mu\rangle a \equiv f'\langle\mu\rangle a'$ if $f \equiv f'$ and $a \equiv a'$; except when $\mu = \mathbf{irr}$, then we would not require $a \equiv a'$. This could be formulated in terms of an erasure function $\lfloor \sqcup \rfloor$ for which $\lfloor f\langle\mathbf{irr}\rangle a \rfloor :\mathrel{\mathring{=}} \lfloor f \rfloor\langle\mathbf{irr}\rangle\bullet$, and $\lfloor f\langle\mu\rangle a \rfloor :\mathrel{\mathring{=}} \lfloor f \rfloor\langle\mu\rangle\lfloor a \rfloor$ for all other μ (where $\mathring{=}$ denotes syntactic equality of de Bruijn terms). We could then say that $t \equiv t'$ whenever $\lfloor t \rfloor \mathrel{\mathring{=}} \lfloor t' \rfloor$. However, in Example 1.6, we saw irrelevance arise as the composite $\mathbf{irr} = \mathbf{par} \circ \mathbf{shi}$. Without special treatment of **par** and **shi**, the irrelevant subterms of Example 1.6 *will* be checked for equality.

For this reason, we need an i-erasure function $\lfloor \sqcup \rfloor_i$ for each $i \in \{0, \dots, n, \top\}$, such that $\lfloor t \rfloor_i$ is the term t, considered up to syntactically obvious i-relatedness. Then we can set $\lfloor f\langle\mu\rangle a \rfloor_i :\mathrel{\mathring{=}} \lfloor f \rfloor_i\langle\mu\rangle\lfloor a \rfloor_{i \cdot \mu}$. If we then apply $\lfloor \sqcup \rfloor_0$ to two terms to find out if they are equal, we will end up applying $\lfloor \sqcup \rfloor_\top$ to irrelevant subterms, which we define to always yield $\bullet$ as all terms are $\top$-related. The rule **T-EQ-ERASE** then asserts that judgemental equality, and hence also the conversion rule T-CONV, only consider terms up to their irrelevant subterms.

Note that we sometimes need to compare erased terms from different types and contexts. For example, when we want to check that $(\mu \mid a, b) \equiv (\mu \mid a', b') : (\mu \mid x : A) \times B[x]$, we will check that $\lfloor a \rfloor_{0 \cdot \mu} \mathrel{\mathring{=}} \lfloor a' \rfloor_{0 \cdot \mu}$ and $\lfloor b \rfloor_0 \mathrel{\mathring{=}} \lfloor b' \rfloor_0$. Here, $b : B[a]$ and $b' : B[a']$ live in different types; however since $\lfloor a \rfloor_{0 \cdot \mu} \mathrel{\mathring{=}} \lfloor a' \rfloor_{0 \cdot \mu}$, the modality of B guarantees that $\lfloor B[a] \rfloor_1 \mathrel{\mathring{=}} \lfloor B[a'] \rfloor_1$, i.e. the types are 1-related so that 0-relatedness of the terms is meaningful. When erasing λ-expressions, we may even end up comparing erased function bodies with variables of different type in the context; again, the appropriate erasures of the types will match.

For this reason, we define the erasure functions not just on terms, but also on contexts and judgements. They are defined as follows: for terms, we simply set $\lfloor t \rfloor_\top :\mathrel{\mathring{=}} \bullet$, while $\lfloor \sqcup \rfloor_i$ is pushed through to μ-modal subterms as $\lfloor \sqcup \rfloor_{i \cdot \mu}$. For variables, we define $\lfloor x \rfloor_i :\mathrel{\mathring{=}} x$ if $i < \top$. For contexts, we define $\lfloor () \rfloor_i := ()$ and $\lfloor \Gamma, \mu \mid x : A \rfloor_i := (\lfloor \Gamma \rfloor_i, \mu \mid \lfloor x \rfloor_{i \cdot \mu} : \lfloor A \rfloor_{i \cdot \mu \cdot \mathbf{par}})$. When applying $\lfloor \sqcup \rfloor_i$ to a judgement, we apply $\lfloor \sqcup \rfloor_i$ to the context and any terms (before the colon) on the right, and $\lfloor \sqcup \rfloor_{i \cdot \mathbf{par}}$ to any types (behind the colon or in the type judgement) on the right. We remark that, if $i \leq j$, then i-relatedness implies j-relatedness and hence $\lfloor \sqcup \rfloor_j$ factors over $\lfloor \sqcup \rfloor_i$. Note that there is no type theory of erased objects; instead erased judgements should be seen as equivalence classes of non-erased judgements. The idea of dealing with judgemental i-relatedness as a relation on derivable judgements is due to Andrea Vezzosi; when this relation turned out to be an equivalence relation, we chose to present the partition as an erasure function instead.

Example 3.2. The list terms in Example 1.6 both 0-erase to

$$\mathsf{cons}\ A \bullet \bullet \bullet\ a\ (\mathit{if}\ (\mathsf{List}_\bullet\ A)\ b\ (\mathsf{nil}\ A \bullet \bullet)\ (\mathsf{cons}\ A \bullet \bullet \bullet\ a'\ as)). \tag{9}$$

Example 3.3. Continuing Example 1.2, we have:

$$\lfloor \mathbf{irr} \mid n : \mathbb{N}, x : \mathbb{N}, xs : \mathsf{List}_n\ \mathbb{N} \vdash_0 \mathsf{cons}\ \mathbb{N}\ n\ (\mathsf{s}\ n)\ \_\ x\ xs : \mathsf{List}_{\mathsf{s}\,n}\ \mathbb{N} \rfloor_0$$
$$= (\mathbf{irr} \mid \bullet : \bullet, x : \mathbb{N}, xs : \mathsf{List}_\bullet\ \mathbb{N} \vdash_0 \mathsf{cons}\ \mathbb{N} \bullet \bullet \bullet\ x\ xs : \mathsf{List}_\bullet\ \mathbb{N}).$$

3.3 Propositional Relatedness and Internal Parametricity

Just as modality-aware equality checking relies on an erasure function for every i because it needs to detect i-relatedness of subterms, modality-aware and in particular parametricity-aware equality *proving* relies on types encoding the proposition 'a_0 and a_1 are i-related', and of course operators for constructing and using elements of those types. For space reasons, we only give a high-level discussion.

Like earlier accounts of internal parametricity [8, 24, 26] and univalence [10], we make use of a special interval pseudo-type $\mathbb{I}$ containing constants $\mathbf{0}$ and $\mathbf{1}$ that are related. In our setting, $\mathbb{I}$ has depth 1 and the constants $\mathbf{0}$ and $\mathbf{1}$ are seen as 1-related, but not 0-related. Write $\underline{i} : 1 \to n$ for the modality such that $j \cdot \underline{i}$ equals 1 if $j \geq i$ and 0 otherwise, e.g. $\underline{2} = \langle 0, 0, 1, \dots, 1\rangle : 1 \to n + 2$ and $\underline{\top} = \mathbf{hoc} : 1 \to n$. One can show that $\mu \setminus \underline{i} = \underline{i \cdot \mu}$. Then a function $b : (\underline{i} \mid \alpha : \mathbb{I}) \to B\,\alpha$ (called an i-**edge**) can serve as a proof that $b\,\mathbf{0} \frown_i b\,\mathbf{1}$ and the codomain $B : (\mathbf{par} \setminus \underline{i} = \underline{i+1} \mid \mathbb{I}) \to \mathcal{U}^n_\ell$ gives us the proof of $B\,\mathbf{0} \frown^{\mathcal{U}^n_\ell}_{i+1} B\,\mathbf{1}$ according to which this is the case. Triviality of the $\top$-relation is internalized by allowing the user to create ad hoc functions from $\mathbb{I}$ by giving values only for $\mathbf{0}$ and $\mathbf{1}$. These simple observations allow us to reason about relations internally.

Example 3.4. The constant edge $\lambda\_.a : (\underline{i} \mid \mathbb{I}) \to A$ shows that every object $a : A$ is i-related to itself. If $j \geq i$, then $\underline{j} \leq \underline{i}$, so that i-edges $a : (\underline{i} \mid \alpha : \mathbb{I}) \to A\,\alpha$ weaken to j-edges $\lambda(\underline{j} \mid \alpha : \mathbb{I}).a\langle\underline{i}\rangle\alpha : (\underline{j} \mid \alpha : \mathbb{I}) \to A\,\alpha$.

Example 3.5 (Pairs)**.** Consider the i-edge of pairs $\lambda(\underline{i} \mid \alpha : \mathbb{I}).(\mu \mid a\,\alpha, b\,\alpha) : (\underline{i} \mid \alpha : \mathbb{I}) \to (\mu \mid x : A) \times B\,\alpha\,x$. Using the typing rules for pair types, we see that this is well-typed if

$$\begin{array}{ll} a : (\underline{i \cdot \mu} \mid \alpha : \mathbb{I}) \to A\,\alpha & A : (\underline{i \cdot \mu + 1} \mid \mathbb{I}) \to \mathcal{U}^m_\ell \\ b : (\underline{i} \mid \alpha : \mathbb{I}) \to B\,\alpha\,(a\,\alpha) & B : (\underline{i+1} \mid \mathbb{I}) \to (\mathbf{par} \setminus \mu \mid A\,\alpha) \to \mathcal{U}^n_\ell \end{array}$$

where we assume $\top + 1 = \top$. This is exactly as one would expect from a relational perspective.

Example 3.6 (Functions)**.** Consider an i-edge of functions $f : (\underline{i} \mid \alpha : \mathbb{I}) \to (\mu \mid x : A\,\alpha) \to B\,\alpha\,x$. For all $j \geq i$, it maps $(j \cdot \mu)$-edges $a : (\underline{j \cdot \mu} \mid \alpha : \mathbb{I}) \to A\,\alpha$ to j-edges $\lambda(\underline{j} \mid \alpha : \mathbb{I}).f\,\alpha\,(a\,\alpha)$. However, we cannot prove that a term such as f can be constructed by giving $f\,\mathbf{0}$, $f\,\mathbf{1}$ and a sensible actions on edges. A generalization of Moulin's Φ-operator [24] would prove this, but is not covered by our model.

Example 3.7 (Types)**.** An $(i+1)$-edge $A : (\underline{i+1} \mid \mathbb{I}) \to \mathcal{U}^n_\ell$ allows us to consider, for all $j \geq i$, the j-edges $(\underline{j} \mid \alpha : \mathbb{I}) \to A\,\alpha$. However, we cannot construct a term such as A by giving $A\,\mathbf{0}$, $A\,\mathbf{1}$ and notions of j-edges for $j \geq i$. A generalization of Moulin's Ψ-operator [24] would allow this, but is again not covered by our model.

As soundness of Moulin's axioms is not covered by our model, we resort to the less expressive alternative given by the Glue and Weld types in our previous work [26]. Among other things, Glue allows us to compose functions $A' \to A$ and $B' \to B$ with an i-edge $A \frown^{\mathcal{U}^n_\ell}_i B$, yielding an i-edge $A' \frown^{\mathcal{U}^n_\ell}_i B'$. Weld allows composition with functions in reverse direction.

Identity extension Finally, as in our previous work [26], we decree by axiom that all non-dependent 0-edges $(\underline{0} \mid \mathbb{I}) \to B$ are in fact constant, so that homogeneous propositional 0-relatedness becomes equivalent to propositional equality. We leave the computational content of this axiom for future work, so for now applications of the J-rule to this axiom are stuck terms.

4 Applications

We briefly discuss three applications of our type system: Church encoding and algebra, intersection and union types, and sized types.

Church encoding and algebra Let F be a level-preserving functor on types of depth n, i.e. an operator $F : \mathcal{U}^n_\ell \to \mathcal{U}^n_\ell$ for every level ℓ (irrelevant in ℓ), which also has an action on functions and which satisfies the functor laws judgementally. The inductive type with a single continuous constructor $FX \to X$ can be mimicked by the Church encoding

$$\mathsf{Mu}F_\ell :\equiv (\mathbf{par} \mid X : \mathcal{U}^n_\ell) \to (FX \to X) \to X. \tag{10}$$

As in §2.4, this can be uncurried to

$$(\mathbf{par} \mid (X, \mathsf{box}^{\mathbf{str}}(\mathsf{mk}X)) : \mathsf{Alg}_\ell F) \to X \tag{11}$$

$$\text{where}\quad \mathsf{Alg}_\ell F :\equiv (X : \mathcal{U}^n_\ell) \times \mathsf{Box}^{\mathbf{str}}(FX \to X) \tag{12}$$

is the type of F-algebras. This reveals the meaning of $\mathsf{Mu}F_\ell$: to give an element of the Church encoding, is to give a canonical (parametric) element that exists in every F-algebra of level ℓ. This clean algebraic formulation of Church encoding is possible thanks to the novel structural modality.

One can easily show that $\mathsf{Mu}F_\ell$ is itself an F-algebra; call its structure $\mathsf{mkMu}F_\ell$. Write $\mathsf{fold}_\ell\ X\ \mathsf{mk}X\ q :\equiv q\ X\ \mathsf{mk}X$. Define $\downarrow : \mathsf{Mu}F_{\mathsf{s}\,\ell} \to \mathsf{Mu}F_\ell$ by restricting the first argument to a smaller universe. Disregarding predicativity issues, we can see $\downarrow$ as the identity, in which case $\mathsf{Mu}F$ really is the initial F-algebra:

Theorem 4.1 (Initiality)**.** *Assume we have* $(\mathbf{par} \mid B : \mathcal{U}^n_{\mathsf{s}\,\ell})$*, an algebra structure* $(\mathbf{con} \mid \mathsf{mk}B : FB \to B)$ *and an algebra morphism* $(\mathbf{con} \mid f : \mathsf{Mu}F_\ell \to B)$ *for which* $f \circ \mathsf{mkMu}F_\ell \equiv \mathsf{mk}B \circ \vec{F}f$*. Then we can prove* $f \circ \downarrow =_{(\mathsf{Mu}F_{\mathsf{s}\,\ell} \to B)} \mathsf{fold}_{\mathsf{s}\,\ell}\ B\ \mathsf{mk}B$ *propositionally.*

We also have the dual theorem for co-algebras. These results are proven as in our previous work [26]. An important difference is that previously the internal parametricity operators had a pointwise dependency, corresponding in the current system to $\mathbf{hoc} : 1 \to 1$. Since this modality has no left inverse, it could not be cancelled, yielding a proof of the initiality theorem with an ad hoc dependency on $\mathsf{mk}B$ and f. In the current setting, we can instead use the finer structural modality, which is left inverted by parametricity, yielding a proof continuous in $\mathsf{mk}B$ and f, to which we can apply further parametricity arguments.

Theorem 4.2. *We have a continuous dependent eliminator*

$$\begin{aligned} \mathsf{ind}_{\mathsf{Mu}F} : (\mathbf{par} \mid C : (\mathbf{str} \mid \mathsf{Mu}F_\ell) \to \mathcal{U}^n_{\mathsf{s}\,\ell}) \to \\ \big((p^* : F\,((q : \mathsf{Mu}F_\ell) \times C\,q)) \to C\,(\mathsf{mkMu}F_\ell(\vec{F}\mathsf{fst}\,p^*))\big) \to \\ (q : \mathsf{Mu}F_{\mathsf{s}\,\ell}) \to C\,(\downarrow q). \end{aligned}$$

Observe how the eliminated q comes from a higher-level type than the one used in the elimination clause. A somewhat dual result is that we can prove, up to predicativity issues, that bisimilar Church-encoded streams are equal.

Sized types Let F be a functor as before. Write $\exists$ and $\forall$ for irrelevant quantification over the naturals. A **sized** F**-algebra** is defined to be a type family $T : (\mathbf{shi} \mid \sigma : \mathbb{N}) \to \mathcal{U}^n_\ell$ equipped with an operation $\mathsf{mk}T : \forall\sigma.F(\exists(\tau < \sigma).T\,\tau) \to T\,\sigma$. We define

$$\begin{aligned} \widehat{\mathsf{Mu}}F_\ell\,\sigma :\equiv (\mathbf{par} \mid X : (\mathbf{shi} \mid \omega : \mathbb{N}) \to \mathcal{U}^n_\ell) \to \\ (\forall\omega.F(\exists(\tau < \omega).X\,\tau) \to X\,\omega) \to X\,\sigma. \end{aligned} \tag{13}$$

Using proof techniques by Vezzosi [26], one can show that under reasonable conditions (e.g. F is a finitely branching container functor) and ignoring predicativity issues, we have $\exists\sigma.\widehat{\mathsf{Mu}}F\,\sigma \cong \mathsf{Mu}F$.

Dually, if F is any container functor, then up to predicativity issues we have $\forall\sigma.\widehat{\mathsf{Nu}}F\ \sigma \cong \mathsf{Nu}F$. The main novelty here with respect to our previous work [26] is that we now use (shape-)irrelevant quantification over $\mathbb{N}$ instead of continuous/parametric quantification over a special Size type; and that many propositional equalities become definitional thanks to the erasure system. In practice, we will instead add size-indexed inductive types to our type system as primitives. In other words, our account of shape-irrelevance supports the applications that Abel et al. [4] introduced it for.

Intersection and Union Types A line of work for which we can cite [11, 21] treats intersection ($S \cap T$) and union ($S \cup T$) types in a manner similar to the product ($S \times T$) and the disjoint union ($S \uplus T$) respectively, in order to achieve type inference in the presence of intersection and union types. Concretely, the type system is equipped with an erasure function which erases annotations required only for type inference, and $A \cap B$ is inhabited by pairs (s, t) for which $s : S$ and $t : T$ have equal erasure; the pair then also erases to the same expression. Similarly, in $S \cup T$ we find terms $(\mathsf{inl}\ s)$ and $(\mathsf{inr}\ t)$ which are indistinguishable when $s : S$ and $t : T$ have equal erasure; the tags inl and inr are erased.

Remarkably, we can mimick this approach using irrelevant quantification over the booleans, if we make the sound assumption that we can create a function $\lambda b.(\mathsf{if}\ \nu \mid b\ \mathsf{then}\ c_{\mathsf{true}}\ \mathsf{else}\ c_{\mathsf{false}}) : (\nu \mid b : \mathsf{Bool}) \to C\ b$ by giving terms $c_{\mathsf{true}} : C\ \mathsf{true}$ and $c_{\mathsf{false}} : C\ \mathsf{false}$ such that $\lfloor c_{\mathsf{true}} \rfloor_i \mathrel{\mathring{=}} \lfloor c_{\mathsf{false}} \rfloor_i$ whenever $i \cdot \nu = \top$. Define:

$$\begin{array}{rcc}
S \cap T & :\equiv & (\mathbf{irr} \mid b : \mathsf{Bool}) \to (\mathsf{if}\ \mathbf{shi} \mid b\ \mathsf{then}\ S\ \mathsf{else}\ T), \\
(s, t) & :\equiv & \lambda(\mathbf{irr} \mid b : \mathsf{Bool}).(\mathsf{if}\ \mathbf{irr} \mid b\ \mathsf{then}\ s\ \mathsf{else}\ t), \\
\mathsf{fst}\ p & :\equiv & p\langle\mathbf{irr}\rangle\mathsf{true} \\
\mathsf{snd}\ p & :\equiv & p\langle\mathbf{irr}\rangle\mathsf{false} \\
S \cup T & :\equiv & (\mathbf{irr} \mid b : \mathsf{Bool}) \times (\mathsf{if}\ \mathbf{shi} \mid b\ \mathsf{then}\ S\ \mathsf{else}\ T), \\
\mathsf{inl}\ s & :\equiv & (\mathbf{irr} \mid \mathsf{true}, s), \\
\mathsf{inr}\ t & :\equiv & (\mathbf{irr} \mid \mathsf{false}, t).
\end{array}$$

Clearly then, $\cap$ and $\cup$ do not satisfy any definitional commutativity, associativity, distributivity or idempotency laws. Moreover, types need to have equal 1-erasure before we can take their intersection or union, e.g. we cannot consider $\mathbb{N} \cup \mathsf{Bool}$ as the shape-irrelevant if-expression that implements $\cup$ will reject these operands for having different 1-erasure, but we can take the union of $\mathsf{List}_4\ A \to \mathsf{List}_4\ A$ and $\mathsf{List}_6\ A \to \mathsf{List}_6\ A$. We see this as a feature: it would be equally sensible to expect $\mathbb{N} \cup \mathsf{Bool} \cong \mathbb{N} \uplus \mathsf{Bool}$ as it is to expect $\mathbb{N} \cup \mathsf{Bool} \cong \mathsf{Unit}$. We prefer to err on the safe side and reject the type altogether. When we consider terms, we see that if $\lfloor s \rfloor_0 \mathrel{\mathring{=}} \lfloor t \rfloor_0$, then $\mathsf{inl}\ s \equiv \mathsf{inr}\ t$ as both terms 0-erase to $(\mathbf{irr} \mid \bullet, \lfloor s \rfloor_0)$ via their implementation. Unfortunately $\lfloor S \cup T \rfloor_1 \not\mathrel{\mathring{=}} \lfloor S \rfloor_1$ and $\lfloor \mathsf{inl}\ s \rfloor_0 \not\mathrel{\mathring{=}} \lfloor s \rfloor_0$, i.e. comparison between unions and non-unions is not presently supported as it is more difficult to prove it sound. Dually, we obtain similar results.

5 Soundness: The Presheaf Model

We give here a high-level discussion of the denotational model, and refer to our technical report [25] for details.

We define a **depth n reflexive graph** Γ to be a diagram in Set of the form

$$\begin{tikzcd}
\Gamma_\bot \arrow[r, "\mathbf{r}"] & \Gamma_0 \arrow[r, "\mathbf{w}_0^1"] & \Gamma_1 \arrow[r, "\mathbf{w}_1^2"] & \ldots \arrow[r, "\mathbf{w}_{n-1}^n"] & \Gamma_n \arrow[llll, bend right, "\mathbf{s}"'] \arrow[llll, bend left, "\mathbf{t}"]
\end{tikzcd} \tag{14}$$

which need not be commutative but should satisfy $\mathbf{s} \circ \mathbf{w}_{n-1}^n \circ \ldots \circ \mathbf{w}_0^1 \circ \mathbf{r} = \mathsf{id}$ and similar for $\mathbf{t}$. For $n = 0$, this is just a reflexive graph. A depth -1 reflexive graph Γ is defined to be just any set $\Gamma_\bot$. An element of $\Gamma_\bot$ is called a **point**, an element of Γ_i an ***i*-edge**. A depth n reflexive graph can be seen as a (contravariant) presheaf over a category RG_n with $n + 2$ objects $\bot, 0, 1, \ldots, n$. By considering presheaves over a somewhat bigger category that also contains monoidal products such as $0 \otimes 2 \otimes 1$ with $\bot$ as the monoidal unit, we can define various notions of **depth n cubical sets**. The reasons for using cubical sets rather than graphs are all technical; for the sake of the exposition here one may think of cubical sets as graphs.

Every presheaf category constitutes a model of dependent type theory with basic type formers and universes [12, 13]. We model judgements of depth n in the category of depth n cubical sets. Modalities and contramodalities are interpreted as functors that preserve the structural rules of type theory. Their definition is conceptually straightforward: for a (contra)modality μ, we set $(\mu\Gamma)_\bot = \Gamma_\bot$ and $(\mu\Gamma)_i := \Gamma_{i \cdot \mu}$ where $\Gamma_\top := \Gamma_\bot \times \Gamma_\bot$.

We show that the entire type system can be modelled using only semantic types that are *discrete*. A graph A is discrete if $\mathbf{r} : A_\bot \to A_0$ (or, in the depth -1 case $(\mathsf{id}, \mathsf{id}) : A_\bot \to A_\bot \times A_\bot$) is an isomorphism. This is generalized to 'dependent graphs' using a lifting property.

Erasure Unfortunately, we do not expect that the erasure functions can be given meaning directly in the current model. For this reason, we take an intermediate step where we synthesize a proof of $a \frown_i b$ whenever $\lfloor a \rfloor_i \mathrel{\mathring{=}} \lfloor b \rfloor_i$ in a slightly extended but erasure-free type system. This system can then be properly interpreted in the model.

6 Related and Future Work

Relation to some other type systems **MLTT** without universe cumulativity embeds into our system if we interpret $\mathcal{U}_\ell$ as $\mathcal{U}_\ell^\ell$ and annotate every dependency with continuity (adjusted using **up** and **dn** as in §2.6). Functions to small types are then automatically parametric, as several authors showed [6, 15, 31]. The core type system of **ParamDTT** [26] with cumulativity but without function extensionality, embeds if we interpret $\mathcal{U}_\ell$ as $\mathsf{Box}^{\langle 0,1\rangle}\ \mathcal{U}_\ell^1$, i.e. all types have depth 1 and the universe is forced to have depth 1 as $\langle 0, 1\rangle : 2 \to 1$ forgets the 2-relation. Moreover, the judgement $\Gamma \vdash T\ \mathbf{type}$ is mapped to $\mathbf{hoc} \setminus \Gamma \vdash_1 T\ \mathbf{type}$, which entails $\mu \setminus \Gamma \vdash_1 T\ \mathbf{type}$ for every $\mu : 1 \to 1$. The pointwise, continuous and parametric modalities map to $\mathbf{hoc} = \langle 0, 0\rangle$, $\mathbf{con} = \langle 0, 1\rangle$, $\mathbf{up} \circ \mathbf{par} = \langle 1, 1\rangle : 1 \to 1$ respectively. **Predicative System Fω** (see [16] for predicative System F) embeds into our system if we give types depth 0 and kinds depth 1, interpreting $*_\ell$ as $\mathcal{U}_\ell^0$, and annotating every dependency with $\mathbf{con} = \mathbf{hoc} : 0 \to 0$, $\mathbf{con} : 1 \to 1$ or $\mathbf{dn} = \mathbf{par} : 1 \to 0$. Church-style versions of the **implicit calculus of constructions** [7, 22, 23] do not embed into our system if we want to map the implicit modality to irrelevance, as the type may not be shape-irrelevant, but we may map implicitness to parametricity if we disable the special conversion rule which allows type-checking time erasure. **Pfenning's** notion of irrelevance [27] does map to irrelevance in our system, but we do not support intensionality.

Modalities and modes Modalities describing function behaviour in type theory have been applied to: modal logic (eponymously) [28], variance of functors [1, 2, 17], intensionality vs. extensionality [27], irrelevance [3, 4, 7, 22, 23, 27, 29], shape-irrelevance [4], parametricity [26], globality [18] and more.

The treatment in terms of order, composition and left division has been developed by Pfenning [27] and Abel [1, 2] and is the basis for the Agda implementation of (shape)-irrelevance. A generalization to a multi-mode system, where every type gets assigned a mode (in our case, a depth) which defines the set of available modalities for functions from/to that type, is described by Licata et al. [19, 20].

Irrelevance and erasure We are aware of two approaches to type-checking time erasure of irrelevant data. One is to view irrelevance as a property of dependencies, i.e. as a modality; see the previous paragraphs for related work and Example 1.4 for a more detailed comparison. The other is to view irrelevance as a property of types, leading to a universe of propositions (as in Coq) whose proofs may be erased (which Coq does not currently do); this corresponds to our types of depth -1. Note that there is no analogue of shape-irrelevance for the latter approach. *Compile-time* erasure, which uses different techniques to achieve different goals (efficiency vs. ease of proving) is beyond the scope of this section.

Parametricity in and of dependent type theory It is known [6, 9, 15, 31] that all functions in dependent type theory (DTT) preserve relatedness — in our terminology: that DTT is continuous. Takeuti [31] and Atkey et al. [6] moreover prove the identity extension lemma for small types, a result that can be phrased in our setting as: all continuous functions with small (depth 0) codomain are parametric (see Fig. 2). Bernardy, Coquand, and Moulin [8, 24] devise internal operators for building 'cheap' proofs of 'free' theorems [33]; however these operators only exploit continuity, not parametricity. In our previous work [26] we used a parametricity modality to allow for internal operators that do exploit parametricity. Atkey et al. [6] prove their results in a reflexive graph model, which has been enhanced by Bernardy, Coquand, and Moulin [8, 24] to a cubical set model (cubical sets can be seen as iterated reflexive graphs). We previously further enhanced the model by annotating every edge with whether it witnesses 0-relatedness (a 'path') or 1-relatedness (a 'bridge') [26]; now instead we annotate every edge with the degree of relatedness it witnesses.

Computation, and cubical homotopy type theory (HoTT) Our type system is a sound proof environment, but not a good programming language: when we apply the J-rule to the identity extension axiom (§3.3), we get a stuck term. In order to achieve a canonicity result, we likely need to use operators similar to the path composition operator from cubical HoTT [10], that allow us to compute transport along equality proofs obtained from the identity extension axiom. From there, we believe it should be possible to merge our system with cubical HoTT into a system for relational HoTT.

Acknowledgments

Thanks to Andreas Abel, Paolo Capriotti, Luigi Liquori, Claude Stolze and Andrea Vezzosi for related discussions, and to the anonymous reviewers for valuable feedback. Andreas Nuyts and Dominique Devriese hold a Ph.D. Fellowship and a Postdoctoral Mandate (resp.) from the Research Foundation - Flanders (FWO).

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. *MSCS* 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* 8, 1 (2012), 1–36. https://doi.org/10.2168/LMCS-8(1:29)2012 TYPES'10 special issue.

[4] Andreas Abel, Andrea Vezzosi, and Theo Winterhalter. 2017. Normalization by Evaluation for Sized Dependent Types. *Proc. ACM Program. Lang.* 1, ICFP, Article 33 (Aug. 2017), 30 pages. https://doi.org/10.1145/3110277

[5] Andreas M. Abel and Brigitte Pientka. 2013. Wellfounded Recursion with Copatterns: A Unified Approach to Termination and Productivity. In *ICFP '13*. ACM, New York, NY, USA, 185–196. https://doi.org/10.1145/2500365.2500591

[6] Robert Atkey, Neil Ghani, and Patricia Johann. 2014. A Relationally Parametric Model of Dependent Type Theory. In *POPL '14*. https://doi.org/10.1145/2535838.2535852

[7] Bruno Barras and Bruno Bernardo. 2008. The Implicit Calculus of Constructions as a Programming Language with Dependent Types. In *FOSSACS '08*. https://doi.org/10.1007/978-3-540-78499-9_26

[8] Jean-Philippe Bernardy, Thierry Coquand, and Guilhem Moulin. 2015. A Presheaf Model of Parametric Type Theory. *Electron. Notes in Theor. Comput. Sci.* 319 (2015), 67 – 82. https://doi.org/10.1016/j.entcs.2015.12.006

[9] Jean-Philippe Bernardy, Patrik Jansson, and Ross Paterson. 2012. Proofs for Free — Parametricity for Dependent Types. *Journal of Functional Programming* 22, 02 (2012), 107–152. https://doi.org/10.1017/S0956796812000056

[10] Cyril Cohen, Thierry Coquand, Simon Huber, and Anders Mörtberg. 2015. Cubical Type Theory: A Constructive Interpretation of the Univalence Axiom. In *TYPES '15*. https://doi.org/10.4230/LIPIcs.TYPES.2015.5

[11] Daniel J. Dougherty, Ugo de'Liguoro, Luigi Liquori, and Claude Stolze. 2016. A Realizability Interpretation for Intersection and Union Types. In *APLAS 2016*. 187–205. https://doi.org/10.1007/978-3-319-47958-3_11

[12] Martin Hofmann. 1997. *Syntax and semantics of dependent types*. Springer London, London, Chapter 4, 13–54. https://doi.org/10.1007/978-1-4471-0963-1_2

[13] Martin Hofmann and Thomas Streicher. 1997. Lifting Grothendieck Universes. Unpublished note. (1997).

[14] John Hughes, Lars Pareto, and Amr Sabry. 1996. Proving the Correctness of Reactive Systems Using Sized Types. In *POPL '96*. ACM, New York, NY, USA, 410–423. https://doi.org/10.1145/237721.240882

[15] Neelakantan R. Krishnaswami and Derek Dreyer. 2013. Internalizing Relational Parametricity in the Extensional Calculus of Constructions. In *CSL 2013 (LIPIcs)*, Vol. 23. Dagstuhl, Germany, 432–451. https://doi.org/10.4230/LIPIcs.CSL.2013.432

[16] Daniel Leivant. 1991. Finitely stratified polymorphism. *Information and Computation* 93, 1 (1991), 93 – 113. https://doi.org/10.1016/0890-5401(91)90053-5

[17] Daniel R. Licata and Robert Harper. 2011. 2-Dimensional Directed Type Theory. *Electr. Notes Theor. Comput. Sci.* 276 (2011), 263–289. https://doi.org/10.1016/j.entcs.2011.09.026

[18] Daniel R. Licata, Ian Orton, Andrew M. Pitts, and Bas Spitters. 2018. Internal Universes in Models of Homotopy Type Theory. In *FSCD '18*.

[19] Daniel R. Licata and Michael Shulman. 2016. Adjoint Logic with a 2-Category of Modes. In *LFCS '16*. https://doi.org/10.1007/978-3-319-27683-0_16

[20] Daniel R. Licata, Michael Shulman, and Mitchell Riley. 2017. A Fibrational Framework for Substructural and Modal Logics. In *FSCD '17*. 25:1–25:22. https://doi.org/10.4230/LIPIcs.FSCD.2017.25

[21] Luigi Liquori and Simona Ronchi Della Rocca. 2007. Intersection-types à la Church. *Inf. Comput.* (2007). https://doi.org/10.1016/j.ic.2007.03.005

[22] Alexandre Miquel. 2001. The Implicit Calculus of Constructions. In *TLCA*. 344–359. https://doi.org/10.1007/3-540-45413-6_27

[23] Nathan Mishra-Linger and Tim Sheard. 2008. *Erasure and Polymorphism in Pure Type Systems*. 350–364. https://doi.org/10.1007/978-3-540-78499-9_25

[24] Guilhem Moulin. 2016. *Internalizing Parametricity*. Ph.D. Dissertation. Chalmers University of Technology, Sweden.

[25] Andreas Nuyts. 2018. *Presheaf Models of Relational Modalities in Dependent Type Theory*. Technical Report. KU Leuven, Belgium.

[26] Andreas Nuyts, Andrea Vezzosi, and Dominique Devriese. 2017. Parametric quantifiers for dependent type theory. *PACMPL* 1, ICFP (2017), 32:1–32:29. https://doi.org/10.1145/3110276

[27] Frank Pfenning. 2001. Intensionality, Extensionality, and Proof Irrelevance in Modal Type Theory. In *LICS '01*. https://doi.org/10.1109/LICS.2001.932499

[28] Frank Pfenning and Rowan Davies. 2001. A judgmental reconstruction of modal logic. *MSCS* 11, 4 (2001), 511–540. https://doi.org/10.1017/S0960129501003322

[29] Jason Reed. 2003. Extending Higher-Order Unification to Support Proof Irrelevance. In *TPHOLs 2003*. 238–252. https://doi.org/10.1007/10930755_16

[30] John C. Reynolds. 1983. Types, Abstraction and Parametric Polymorphism.. In *IFIP Congress*. 513–523.

[31] Izumi Takeuti. 2001. *The Theory of Parametricity in Lambda Cube*. Technical Report 1217. Kyoto University.

[32] The Univalent Foundations Program. 2013. *Homotopy Type Theory: Univalent Foundations of Mathematics*. http://homotopytypetheory.org/book, IAS.

[33] Philip Wadler. 1989. Theorems for Free!. In *FPCA '89*. ACM, New York, NY, USA, 347–359. https://doi.org/10.1145/99370.99404