

GLUING MODELS OF TYPE THEORY ALONG FLAT FUNCTORS

JONATHAN STERLING AND CARLO ANGIULI

ABSTRACT. We extend the theory of *Artin gluing* to strict dependent type theory: given a flat functor $\mathcal{C} \xrightarrow{F} \mathcal{E}$ from the category of contexts of a model of Martin-Löf type theory into a Grothendieck topos \mathcal{E} , we may construct the *F-computability families* model of type theory. Our theorem extends to MLTT with a (strict, weak) universe à la Tarski if \mathcal{E} may be equipped with a (strict, weak) universe à la Tarski.

We introduce a more tractable approach to the gluing models of type theory, working primarily within a suitably enlarged category of computability families which are not all tracked by syntactical entities, but which is densely generated by the subcategory of such computability families.

1. INTRODUCTION

Most important properties of sufficiently complex languages, including canonicity, normalization, decidability of judgmental equality, and parametricity, cannot be proved by induction on the syntax of terms or derivations. *Artin gluing* is a unifying technique which equips a language with a much stronger *semantic* induction principle well-suited to proving all these metatheorems.

1.1. Artin gluing and the method of computability. In the history of logic and structural proof theory, the struggle for these semantic induction principles begins with Tait’s pioneering proof of strong normalization for a rewriting presentation of the simply typed λ -calculus [Tai67]; the method employed, now variously called *Tait’s method*, *the method of computability/reducibility*, *logical/computability/reducibility predicates/relations*, etc., was further developed by Girard [Gir71; Gir72] and Martin-Löf [Mar75].

Artin gluing was invented by the Grothendieck school of algebraic geometry in the early 1970s [AGV72]. Given an accessible and finite limit-preserving functor $\mathcal{X} \xrightarrow{F} \mathcal{E}$ between Grothendieck topoi, one may construct a new Grothendieck topos whose objects are *families* $E \rightarrow F(X) : \mathcal{E}$ with $X : \mathcal{X}$, and whose morphisms are commutative squares. This glued topos $\mathcal{E} \downarrow F$ comes with a projection functor $\mathcal{E} \downarrow F \rightarrow \mathcal{X}$ preserving all the structure of a topos, equipped with both left and right adjoints.

The closure of Grothendieck topoi under gluing was subsequently generalized by Tierney to the case of *elementary* topoi [Wra74], which include topoi that may not have a description in terms of generators and relations. This essential step enabled Freyd to use the theory of gluing to prove that the terminal object of the free topos¹ with a natural numbers object is connected and projective [Fre78]. Considering the correspondence

Date: January 2020.

This material is based on research sponsored by Air Force Office of Scientific Research through MURI grant FA9550-15-1-0053. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the AFOSR..

¹Here we mean the initial object in the category of elementary topoi with n.n.o.s and *logical* morphisms.

between the free topos with a n.n.o. and intuitionistic higher-order predicate logic, the result immediately implies both the disjunction and existence properties [nLa18a; nLa17].

Proof (ir)relevance. Freyd’s construction involves gluing the free topos \mathcal{F} along the *global sections* functor $\mathcal{F} \xrightarrow{\Gamma_{\mathcal{F}}} \mathbf{Set}$; the objects of the resulting “Freyd cover” $\mathbf{Set} \downarrow \Gamma_{\mathcal{F}}$ are families of sets indexed in the global elements of \mathcal{F} -objects. As observed by Mitchell and Scedrov [MS93], the Freyd cover is *almost* the same as the Tait-style models used to prove the *canonicity* property of type theories, except that the Tait-style models are restricted to proof-irrelevant predicates over global elements rather than general families.

A great deal of the literature in programming languages and type theory has been oriented toward artificially enforcing the same proof-irrelevance constraint in the case of *non-Freyd* gluing, the paradigmatic example being the gluing along a nerve functor; this method, called *Kripke logical predicates of varying arity* in the type-theoretic literature [JT93; MS93; FS99], corresponds exactly to the method of Kripke logical predicates/relations which has become central to the study of programming languages.

More recently, Coquand has argued that the proof irrelevance of the classical method of computability is a disadvantage in comparison with the more natural gluing construction in terms of families [Coq19]. Gluing, or “proof-relevant computability,” enjoys a direct and elementary application to the metatheory of dependent type theory with hierarchies of universes, in contrast to the proof-irrelevant version which appeared to require a highly technical detour through partial equivalence relations over rewriting systems on raw terms.

Computability predicates for universes. The main difficulty in the proof-irrelevant case is that the computability predicate for a *universe* \mathcal{U} must somehow associate to each $A : \mathcal{U}$ a computability predicate for its decoding $\text{el}(A)$. Because the predicate for \mathcal{U} cannot store those predicates directly, we instead simply store a “permission” to access a separately-maintained global lookup table.

These lookup tables, called *type systems* by Allen [All87], have contributed a critical component to the metatheory of dependent type theories [Har92; KPB15; AR14; AHH17; BGM17; SH18; WB18; GSB19; Ang19]. Their construction, however, relies on rather subtle fixed point theorems, and unfortunately, their basic closure properties must all be established globally by simultaneous induction. For this reason, modular proofs of (e.g.) canonicity and normalization for dependent type theories have been elusive.

In contrast, proof-relevant computability *families* enable a more direct approach to universes: the computability family for a universe may store, as data, the computability families for its elements. Then, rather than interpreting $\text{el}(A)$ using a global lookup table, one may simply project the computability family from the computability data attached to A .

1.2. Compact and general computability families. A typical Martin-Löf-style type theory is a fairly wild object, suffering from a paucity of limits and colimits, and is consequently closed under only some dependent sums and products.² Another characteristic of type theory poorly adapted for categorical manipulation is its strictness: the constructs of type theory are closed under a *substitution* operation which is coherent in the sense that it commutes strictly with chosen representatives of connectives of type theory, e.g. $[M/z]\Pi_{x:A}B \equiv \Pi_{x:[M/z]A}[\hat{x}^*M/z]B$.

²Usually the ones determined as adjoints to weakening.

Because of the strictness of the substitution action, it is not appropriate to treat types in the semantics of (strict) type theory as just some morphisms in a category \mathcal{C} for which the change of base has left and right adjoints. Instead, models of type theory are arranged in a more intricate way: the types are rendered as elements of a *universe* \mathbf{T} which by necessity lies not in \mathcal{C} but in the category of presheaves $\mathbf{Pr}(\mathcal{C})$. Then, the substitution action on types for morphisms $\Delta \xrightarrow{y} \Gamma : \mathcal{C}$ is given by the functoriality of the presheaf $\mathbf{T} : \mathbf{Pr}(\mathcal{C})$.

The yoga of dense generation. While the category of contexts \mathcal{C} is somewhat irregular, its free co-completion $\mathbf{Pr}(\mathcal{C})$ is particularly tame. In addition to the universe of types, most components of models of type theory are native to $\mathbf{Pr}(\mathcal{C})$, including the maps which model the rules of type theory (e.g. dependent product introduction, elimination, etc.).

This situation may be contextualized within the history of mathematical ideas, considering the dual Yoneda lemma which exposes \mathbf{T} (and any other presheaf) as a formal piecing-together of objects from \mathcal{C} . That is, writing $\mathcal{C} \hookrightarrow \mathbf{Pr}(\mathcal{C})$ for the Yoneda embedding:

$$\text{(Density)} \quad \mathbf{T} \cong \text{colim}_{\mathcal{C} \rightarrow \mathbf{T}} \mathcal{C} \rightarrow \mathbf{T}$$

The dense generation of $\mathbf{Pr}(\mathcal{C})$ by \mathcal{C} and the attendant advantages of studying objects in \mathcal{C} by means of their (fully faithful) avatars in the tame category $\mathbf{Pr}(\mathcal{C})$ is an old story in mathematics first told by Grothendieck in the context of *scheme theory*, a program of regularization which initiated the era of modern algebraic geometry.

Example 1 (Affine schemes). *Algebraic geometry is classically the study of zeroes of families of polynomials, a situation generalized by the notion of affine schemes, the formal duals of commutative rings. Although the category \mathbf{Aff} of affine schemes is not closed under many operations, it densely generates the more regular Zariski topos $\mathbf{Sh}(\mathbf{Aff})$ in which the affine schemes become the representables, and the general schemes are the sheaves which are covered by affines via open maps [Gro60; nLa18b].* 🍷

Example 2 (Smooth manifolds). *Synthetic differential geometry is concerned with the study of smooth spaces and smooth maps between them; categories of manifolds being somewhat irregular, one may consider the fully faithful embedding which sends a smooth manifold to its (finitely generated) smooth algebra. The category of sheaves on the formal duals of (finitely generated) smooth algebras is, then, a suitable environment for studying smooth spaces [Koc06; nLa16].* 🍷

Compact computability families. Computability à la Tait is concerned with structures (or properties) that lie over syntactical entities from some type theory; the canonical example of a computability family is the family over $\mathbf{term}[\mathbf{nat}]$ whose fiber at $\vdash t : \mathbf{nat}$ is the collection of natural numbers $n \in \mathbb{N}$ such that $\vdash [n] = t : \mathbf{nat}$ in the type theory.

These “compact” computability families, meaning those which lie over individual contexts or types, are not well-adapted on their own for the metatheory of *dependent* type theories, where one must consider not only the computable elements of types but also the computable types and families thereof. This has led to the proliferation of *ad hoc* treatments of computability for dependent type theory, in which one defines in a duplicative way the (spiritually) identical notions of a “computable map between computability families” and a “computable family of computability families”.

Because the collection of types is not a context in dependent type theory, the spirit of Examples 1 and 2 invites us to enlarge the collection of computability families to include structures and properties that lie over more general objects than the contexts of the type theory. Taking note of the role of presheaves in defining the collections of types and elements, it is particularly appropriate to consider a category \mathcal{G} of “general” computability

families which lie over objects of $\mathbf{Pr}(\mathcal{C})$, densely generated by a subcategory \mathcal{G}_K of “compact” computability families which lie over objects of \mathcal{C} .

1.3. Summary of contributions. In this paper, we develop the theory of Artin gluing in the context of strict type theory; by working in a category \mathcal{G} of “general computability families” which are not *a priori* tracked by syntactical entities from the type theory, we obtain a smoother and more synthetic construction of the model of type theory (glued natural model, glued connectives, etc.) which avoids the *analytic* quantification over contexts and substitutions plaguing prior work on gluing for strict type theory.

Theorem 3. *Let \mathcal{C} be the category of contexts of a model $\mathcal{M}_{\mathcal{C}}$ of Martin-Löf type theory, and let $\mathcal{C} \xrightarrow{F} \mathcal{E}$ be a flat functor into a Grothendieck topos \mathcal{E} (equivalently, a functor whose Yoneda extension $\mathbf{Pr}(\mathcal{C}) \xrightarrow{\tilde{F}} \mathcal{E}$ is left exact). Then we have a model $\mathcal{M}_{\mathcal{E} \downarrow F}$ together with a homomorphism of models $\mathcal{M}_{\mathcal{E} \downarrow F} \rightarrow \mathcal{M}_{\mathcal{C}}$ which is tracked by the codomain projection of the comma category $\mathcal{E} \downarrow F \rightarrow \mathcal{C}$.*

Most of the instances of gluing for type theory that we are aware of correspond to flat functors into Grothendieck topoi.³

Example 4 (Global canonicity). *The global sections functor $\mathcal{C} \xrightarrow{\Gamma_{\mathcal{C}}} \mathbf{Set}$ of a type theory is flat; the corresponding gluing model of type theory may be used to establish strict canonicity.* 🐼

Example 5 (Cubical canonicity). *Awodey observed that an interval object induces a nerve functor from cubical type theory into cubical sets; it is easy to check that this nerve is moreover flat. The corresponding gluing model of type theory may be employed to establish strict canonicity in interval-only contexts, as in the work of Sterling, Angiuli, and Gratzner [SAG19] and unpublished work by Awodey and Fiore.*

From the perspective of cubical sets, this cubical nerve is just the “global sections” functor; it appears as such in the work of Coquand, Huber, and Sattler [CHS19] where it is used to prove homotopy canonicity for a weak cubical type theory. 🐼

Example 6 (Normalization). *The interpretation of formal renamings of variables as substitutions induces a flat nerve functor $\mathcal{C} \xrightarrow{N} \mathbf{Pr}(\mathbf{Ren})$.⁴ The resulting gluing model of type theory may be used to prove normalization and decidability of judgmental equality, as in the work of Fiore [Fio02], Altenkirch, Hofmann, and Streicher [AHS95], and Coquand [Coq19].* 🐼

1.4. Related work. The prior works on gluing for type theory can be roughly divided into those which target *strict* type theory (type theory in which substitution commutes with type-formers on the nose) and those which target *weak* type theory (type theory in which substitution is defined up to isomorphism).

1.4.1. Gluing for strict type theory. Kaposi, Huber, and Sattler [KHS19] have contributed the most general gluing theorem for models of (strict) type theory to date: given two models of type theory and a *pseudo-morphism* of cwfs/natural models between them, they obtain a glued model of type theory displayed over the first model. The work of Kaposi, Huber, and Sattler is essentially syntactic, and uses the strict model theory of quotient inductive-inductive types [AK16; KKA19], a type-theoretic reformulation of Cartmell’s generalized algebraic theories [Car86]

³By a “flat functor”, we mean what is sometimes referred to in the literature as an *internally flat* functor.

⁴This nerve is sometimes billed as a “Yoneda embedding” (e.g. Kaposi, Huber, and Sattler [KHS19]), but we find this perspective highly misleading considering that it is not even fully faithful. Only fully faithful nerves, equivalently nerves generated by dense functors, can seriously be referred to as “Yoneda embeddings”.

The present authors have been curious whether the results of Kaposi, Huber, and Sattler [KHS19] could be cast into a more tractable categorical language which leverages the classical theory of Artin gluing, and in which laborious syntactic computations may in part be replaced by *mathematical* abstractions which are less sensitive to the exact equality of objects.

The current paper is a first step along the road toward a mathematical version of gluing for strict type theory, employing a 2-categorical model theory due to Uemura [Uem19] in which morphisms need preserve type-theoretic constructs only up to isomorphism. This weak kind of morphism, which generalizes pseudo-morphisms of cwfs/natural models to the rest of the connectives of type theory, enables the use of categorical tools (which leverage universal properties) to prove metatheoretic results.

Whereas the input of the theorem of Kaposi, Huber, and Sattler is a pseudo-morphism of natural models, we have restricted our attention to a special case which covers all metatheorems for type theory that we are aware of: a *flat* functor from the category of contexts of a model of type theory into a Grothendieck topos \mathcal{E} . We implicitly use the canonical model of type theory in \mathcal{E} to carry out our result, but we have found it most natural to work from the perspective of flat functors (functors whose Yoneda extensions are left exact) without bringing any algebraic notions into the picture.

Avoiding quantification over substitutions. In previous work on gluing for strict type theory, it was apparently necessary to construct the presheaf of computable types and the presheaves of computable elements at the level of sets, working explicitly with computable contexts and computable substitutions [Coq19; SAG19; KHS19; CHS19]. This analytic approach, which shatters the notion of a computability family into its fibers at each context, is particularly unwieldy: when fundamental definitions are given at the level of sets and not characterized abstractly, all remaining constructions must likewise be carried out at the level of sets.

The analytic definition of these presheaves on (compact) computability families of contexts leads to an avalanche of technical functoriality and naturality obligations which are in practice never explicitly discharged, and whose proofs evince no conceptual insight. It is perhaps reasonable to question whether the economy accorded by the use of proof-relevant computability families and quotiented/abstract terms is not in fact negated by these obligations.

In contrast, our category \mathcal{G} of general computability families provides a suitable environment in which to construct the fundamental objects of the computability model of type theory in a *synthetic* manner; yet, to obtain a model of type theory, it is still necessary to ultimately transfer these objects to the presheaf category $\mathbf{Pr}(\mathcal{G}_K)$ on compact computability families. This *post hoc* shattering of general computability families into presheaves on computable contexts is achieved parsimoniously by a *nerve* functor $\mathcal{G} \xrightarrow{\text{Nk}} \mathbf{Pr}(\mathcal{G}_K)$ which is well-behaved by virtue of the density of \mathcal{G}_K in \mathcal{G} .

1.4.2. *Gluing for weak type theory.* An alternative is to consider gluing for *weak* type theory, in which substitution commutes with type formers only up to isomorphism. Weak type theory is closer to the type-theoretic situations arising organically in the mathematical landscape, such as locally cartesian closed categories [See84; Hof95; CD14; CGH14], Joyal's clans and tribes [Joy17], etc.

In general, developing the metatheory of weak dependent type theory is considerably easier than for the strict version; this is mainly because, when laws are taken up to

isomorphism (such as the commutation of substitution and type-theoretic connectives embodied in *Beck-Chevalley*), the potential for coherence problems is mostly eliminated.

This is why, in our estimation, the theory of Artin gluing for *weak* type theory has been comparatively better developed in terms of both abstractness and generality, as in the work of Shulman [Shu15], Uemura [Uem17], and Kapulkin and Sattler [KS19].

Our aim in the present work is to develop a tractable and *categorical* account of gluing in the same style, but which may be applied to prove the strict metatheorems about strict type theory that are needed to justify the correctness of proof assistants like Agda [Nor07; Nor09; VMA19], Coq [Coq16], Lean [Mou+15], and `redtt` [Red18; Ang+18].


Acknowledgments. We are very thankful to Mathieu Anel, Steve Awodey, Lars Birkedal, Evan Cavallo, Thierry Coquand, Marcelo Fiore, Jonas Frey, Daniel Gratzer, Robert Harper, Alex Kavvos, Michael Shulman, Bas Spitters, Thomas Streicher, and Andrew Swan for helpful conversations and seminars on the model theory of type theory, Artin gluing, and the method of computability.


2. FUNCTORIAL SEMANTICS OF MARTIN-LÖF’S TYPE THEORY

Lawvere’s method of functorial semantics [Law63] replaces syntactic presentations of theories with an invariant presentation, the *classifying category*, such that models of a theory are just structure-preserving functors out of its classifying category. Recently, Uemura [Uem19] has extended functorial semantics to dependent type theory; in Uemura’s framework, a “type theory” is just a small *representable map category* \mathbb{T} , and a model of this type theory is a structure-preserving functor $\mathbb{T} \rightarrow \mathbf{Pr}(\mathcal{C})$ for some category \mathcal{C} with a terminal object.

Many standard doctrines (e.g. finite products, finite limits, etc.) arise as different type theories in the above sense; then, Uemura’s concept of a *theory over a type theory* subsumes the standard notions of theory in each doctrine (e.g. algebraic theory, essentially algebraic theory, etc.). Thus, a type theory in the sense of Uemura [Uem19] is more like a *logical framework* in the sense of Nordström, Peterson, and Smith [NPS90] and Harper, Honsell, and Plotkin [HHP93].

2.1. An overview of the functorial semantics. In this section, we will recall enough of Uemura’s definitions and results to make our constructions self-contained; for more details, we recommend reading Uemura [Uem19].

Notation 1. We will write $\Delta^n : \mathbf{Cat}$ for the n -simplex regarded as a category; for instance, Δ^0 is the point $\{*\}$, and Δ^1 is the walking arrow $\{0 \rightarrow 1\}$. 

Notation 2. In light of Notation 1, $[\Delta^1, \mathcal{C}]$ is the category of arrows and commuting squares in \mathcal{C} , and $[\Delta^1, \mathcal{C}]_{\text{cart}}$ is the category of arrows and *cartesian* squares (pullbacks) in \mathcal{C} . We write $[\Delta^1, \mathcal{C}] \xrightarrow{\partial_1} \mathcal{C}$ for the *codomain opfibration* of \mathcal{C} which projects the codomain of a map; when \mathcal{C} has pullbacks, this is furthermore a fibration $[\Delta^1, \mathcal{C}] \xrightarrow{\partial_1} \mathcal{C}$. 

Type theories as representable map categories. A representable map category in the sense of Uemura [Uem19] is a finitely complete category \mathbb{T} (writing $\diamond : \mathbb{T}$ for the terminal object), together with a wide subcategory $\mathcal{R}_{\mathbb{T}} \hookrightarrow \mathbb{T}$ which is closed under pullback along arbitrary arrows, such that \mathbb{T} has right adjoints $\mathbb{T}/_A \xrightarrow{f_*} \mathbb{T}/_B$ to pullback along any map $A \xrightarrow{f} B \in \mathcal{R}_{\mathbb{T}}$. A map in $\mathcal{R}_{\mathbb{T}}$ is called a *representable* map.

The closure under pullback makes the subcategory of representable maps $[\Delta^1, \mathcal{R}_T] \hookrightarrow [\Delta^1, \mathbb{T}]$ replete. A representable map functor is a left exact functor which preserves the representable arrows, as well as the pushforwards f_* for representable $A \xrightarrow{f} B$.

Remark 3. The notion of a representable map category is the same as that of a category equipped with a universe of “small maps” [Str05] closed under dependent sums, such that “large” dependent products along small families exist. \heartsuit

Representable map categories capture the judgmental situation of type theories in the following way:

- (1) A form of judgment is a map/family in \mathbb{T} ; e.g., the $\langle A \text{ type} \rangle$ judgment is a family $\mathbb{T} \rightarrow \diamond$.
- (2) The *presuppositions* of a form of judgment, in the sense of Schroeder-Heister [Sch87], are rendered as the base of this family; for instance, the judgment $\langle M : A \rangle$ which presupposes $\langle A \text{ type} \rangle$ is a family $\dot{\mathbb{T}} \xrightarrow{\tau} \mathbb{T}$. The fiber $\tau[A]$ collects the elements of the type $A : \mathbb{T}$.
- (3) A form of judgment which may be hypothesized is a *representable* map, and hypothetical judgments are given by pushforwards of those maps. For instance, by making $\dot{\mathbb{T}} \xrightarrow{\tau} \mathbb{T}$ representable, we support the hypothetical judgments $\langle J[x] \ (x : \tau[A]) \rangle$; in *polymorphic* type theory [Jac99], one might additionally make $\mathbb{T} \rightarrow \diamond$ representable.

The canonical representable map category. Given a category \mathcal{C} , there is a “canonical” representable map category structure on the category of presheaves $\mathbf{Pr}(\mathcal{C})$, in which a natural transformation $X \xrightarrow{f} Y : \mathbf{Pr}(\mathcal{C})$ is *representable* iff every fiber of f over a representable presheaf is a representable presheaf (in the sense of the Yoneda embedding). This notion of representability agrees with the classical one, and a representable map $\dot{\mathbb{T}} \xrightarrow{\tau} \mathbb{T} \in \mathcal{R}_{\mathbf{Pr}(\mathcal{C})}$ is exactly a *natural model* over \mathcal{C} in the sense of Awodey [Awo18].

Therefore, a *model* of a type theory \mathbb{T} should be a representable map functor $\mathbb{T} \rightarrow \mathbf{Pr}(\mathcal{C})$ for some \mathcal{C} ; it is easy to see that, in the case of the type theory $\mathbb{T} = \{\dot{\mathbb{T}} \xrightarrow{\tau} \mathbb{T}\}$, this notion of model restricts exactly to that of natural models [Awo18] or (equivalently) categories with families [Dyb96].

Uemura promotes a more symmetric perspective on this canonical representable map category, using the equivalent category of discrete fibrations over \mathcal{C} instead of $\mathbf{Pr}(\mathcal{C})$.

Notation 4. We write \mathbf{DF} for the category of discrete fibrations, i.e., the full subcategory of \mathbf{Fib} consisting of fibrations whose fibers are all sets. Then $\mathbf{Fib} \xrightarrow{\pi_{\mathbf{Fib}}} \mathbf{Cat}$ extends to a fibration $\mathbf{DF} \xrightarrow{\pi_{\mathbf{DF}}} \mathbf{Cat}$, and we write $\mathbf{DF}_{\mathcal{C}}$ for $\pi_{\mathbf{DF}}[\mathcal{C}]$, the category of discrete fibrations over the category \mathcal{C} . \heartsuit

Any presheaf $X : \mathbf{Pr}(\mathcal{C})$ may be turned into a discrete fibration $\mathcal{C}/X \rightarrow \mathcal{C}$, writing \mathcal{C}/X for the category of elements of X ; this assignment extends to an equivalence of categories. However, representability is considerably simpler to state in $\mathbf{DF}_{\mathcal{C}}$: a representable map $X \xrightarrow{f} Y : \mathbf{DF}_{\mathcal{C}}$ is one which, considered as a functor between total categories, has a right adjoint $Y \xrightarrow{q_f} X : \mathbf{Cat}$ [Awo+14; Awo18].

Definition 5. A model of a type theory \mathbb{T} is a functor of representable map categories $\mathbb{T} \xrightarrow{\mathcal{M}_{\mathcal{C}}} \mathbf{DF}_{\mathcal{C}}$, for some category \mathcal{C} with a terminal object. We will write $X_{\mathcal{C}}$ (resp., $f_{\mathcal{C}}$) for $\mathcal{M}_{\mathcal{C}}(X)$ (resp., $\mathcal{M}_{\mathcal{C}}(f)$) for each $X : \mathbb{T}$ and $X \xrightarrow{f} Y : \mathbb{T}$. \heartsuit

Remark 6. The notation \mathbf{q}_f is chosen suggestively: in a model $\mathcal{M}_\mathcal{C}$ of the “walking natural model” $\{\dot{\mathbf{T}} \xrightarrow{\tau} \mathbf{T}\}$, the right adjoint $\mathbf{T}_\mathcal{C} \xrightarrow{\mathbf{q}_f} \dot{\mathbf{T}}_\mathcal{C}$ takes a type $A : \mathbf{T}_\mathcal{C}$ in context Γ to the variable term $\mathbf{q}_f^\Gamma(A) : \dot{\mathbf{T}}_\mathcal{C}$ in context $\Gamma.A$. \heartsuit

A 2-category of models. In contrast to the *generalized algebraic* approach to the model theory of type theory initiated by Cartmell [Car86] and promoted by a number of recent authors [ACD08; Awo18; New18; KHS19; SAG19; Ste18], Uemura’s framework evinces a (strict) 2-category of models for each type theory, equipped with a *bi-initial* object (the term model). This higher-dimensional approach is better adapted to the methods of categorical semantics than the purely 1-categorical approach, in which homomorphisms must commute with all structure strictly: such strict homomorphisms are rarely found in nature, and even when they do exist, they defy mathematical abstraction and are consequently extremely difficult to construct.

Definition 7. Let $\mathbb{T} \xrightarrow{\mathcal{M}_\mathcal{C}} \mathbf{DF}_\mathcal{C}$ and $\mathbb{T} \xrightarrow{\mathcal{M}_\mathcal{D}} \mathbf{DF}_\mathcal{D}$ be models of a type theory \mathbb{T} . A *morphism* from $\mathcal{M}_\mathcal{C}$ to $\mathcal{M}_\mathcal{D}$ is a functor $\mathcal{C} \xrightarrow{F} \mathcal{D}$ preserving the terminal object, together with a natural transformation $\mathcal{M}_\mathcal{C} \xrightarrow{\mathcal{M}_F} \mathcal{M}_\mathcal{D} : [\mathbb{T}, \mathbf{DF}]$ where each component $X_\mathcal{C} \xrightarrow{X_F} X_\mathcal{D} : \mathbf{DF}$ lies over $\mathcal{C} \xrightarrow{F} \mathcal{D}$ in $\mathbf{DF} \xrightarrow{\pi_{\mathbf{DF}}} \mathbf{Cat}$, subject to an additional condition. Consider the image in \mathbf{DF} of the naturality square of any representable map $X \xrightarrow{f} Y \in \mathcal{R}_\mathbb{T}$, filling in the right adjoints to the representable maps:

$$\begin{array}{ccc} X_\mathcal{C} & \xrightarrow{X_F} & X_\mathcal{D} \\ \downarrow f_\mathcal{C} & & \downarrow f_\mathcal{D} \\ Y_\mathcal{C} & \xrightarrow{Y_F} & Y_\mathcal{D} \end{array} \quad \begin{array}{c} \mathbf{q}_{f_\mathcal{C}} \curvearrowright \\ \mathbf{q}_{f_\mathcal{D}} \curvearrowleft \end{array}$$

There is a canonical *comparison map* from the composite $X_F \circ \mathbf{q}_{f_\mathcal{C}}$ to the composite $\mathbf{q}_{f_\mathcal{D}} \circ Y_F$:

$$\begin{array}{l} \frac{\text{identity}}{\mathbf{q}_{f_\mathcal{C}} \rightarrow \mathbf{q}_{f_\mathcal{C}}} \\ \frac{f_\mathcal{C} \circ \mathbf{q}_{f_\mathcal{C}} \rightarrow \mathbf{id}_{Y_\mathcal{C}}}{Y_F \circ f_\mathcal{C} \circ \mathbf{q}_{f_\mathcal{C}} \rightarrow Y_F} \\ \frac{Y_F \circ f_\mathcal{C} \circ \mathbf{q}_{f_\mathcal{C}} \rightarrow Y_F}{f_\mathcal{D} \circ X_F \circ \mathbf{q}_{f_\mathcal{C}} \rightarrow Y_F} \\ \frac{f_\mathcal{D} \circ X_F \circ \mathbf{q}_{f_\mathcal{C}} \rightarrow Y_F}{X_F \circ \mathbf{q}_{f_\mathcal{C}} \rightarrow \mathbf{q}_{f_\mathcal{D}} \circ Y_F} \end{array} \quad \begin{array}{l} \\ \\ \text{naturality} \\ f_\mathcal{D} \dashv \mathbf{q}_{f_\mathcal{D}} \end{array}$$

We require that this comparison map be an isomorphism; this condition is called *Beck-Chevalley*. \heartsuit

Remark 8. The Beck-Chevalley condition on the action of morphisms of models on representable maps can be glossed in an intuitive way: hypothesizing a judgment and then doing a change-of-model is the same as first doing a change-of-model and then hypothesizing. \heartsuit

Example 9. In the case of the type theory $\mathbb{T} = \{\dot{\mathbf{T}} \xrightarrow{\tau} \mathbf{T}\}$, the Beck-Chevalley condition corresponds exactly to the requirement in weak morphisms of natural models [CD14; New18; Clo+18] that the action of the induced transformation of natural models commutes with context extension. \heartsuit

We will often write \mathcal{M}_F to refer to a morphism of models, when we really mean the pair (F, \mathcal{M}_F) .

Definition 10. Let $\mathcal{M}_F, \mathcal{M}_G$ be two morphisms between models $\mathcal{M}_{\mathcal{C}}, \mathcal{M}_{\mathcal{D}}$ of \mathbb{T} . Then, a 2-morphism $\mathcal{M}_F \xrightarrow{\mathcal{M}_\alpha} \mathcal{M}_G$ is a natural transformation $F \xrightarrow{\alpha} G : [\mathcal{C}, \mathcal{D}]$ which lies underneath some (necessarily unique) natural transformation $X_F \xrightarrow{X_\alpha} X_G : [X_{\mathcal{C}}, X_{\mathcal{D}}]$ for each $X : \mathbb{T}$. \heartsuit

Together, all these notions assemble into a 2-category $\mathbf{Mod}_{\mathbb{T}}$ of models of a type theory \mathbb{T} .

The bi-initial model. Uemura [Uem19] shows that we have a *bi-initial* object $\mathbb{T} \xrightarrow{\mathcal{M}_J} \mathbf{DF}_J$ in each 2-category of models, formed by taking the *heart* of the Yoneda embedding $\mathbb{T} \xrightarrow{\mathcal{K}_{\mathbb{T}}} \mathbf{DF}_{\mathbb{T}}$. The heart of a model is the core whose category of contexts is *democratic*, in the sense of being totally generated by context extensions at representable maps.

The universal property of the bi-initial object ensures for any model $\mathcal{M}_{\mathcal{C}} : \mathbf{Mod}_{\mathbb{T}}$, a contractible space of morphisms $\mathcal{M}_J \rightarrow \mathcal{M}_{\mathcal{C}}$; in other words, there is at least one such morphism, and for any two morphisms $\mathcal{M}_J \xrightarrow{F, G} \mathcal{M}_{\mathcal{C}}$, we are guaranteed a *unique* invertible 2-morphism $\mathcal{M}_F \xrightarrow{\alpha} \mathcal{M}_G$.

2.2. Martin-Löf type theory. We will give a modular treatment of Martin-Löf type theory, defining various possible extensions to the “walking natural model” $\mathbb{T}_0 = \{\dot{\mathbb{T}} \xrightarrow{\tau} \mathbb{T}\}$, each corresponding to the closure of the type theory under different connectives. A type theory \mathbb{T} which includes \mathbb{T}_0 will be called a “Martin-Löf type theory”.

Using pullback, pushforward, and composition it is possible to define families in \mathbb{T} which express the generic *dependent (product, sum)* situation for τ .⁵ First, we define the bases of the families $\dot{\mathbb{T}}^\Sigma \xrightarrow{\tau^\Sigma} \mathbb{T}^\Sigma$ and $\dot{\mathbb{T}}^\Pi \xrightarrow{\tau^\Pi} \mathbb{T}^\Pi$:

$$\mathbb{T}^\Sigma, \mathbb{T}^\Pi = (A : \mathbb{T}) \times \tau[A] \Rightarrow \mathbb{T}$$

Then, we define the rest of the family fiberwise:

$$\begin{aligned} \tau^\Sigma[(A, B)] &= (a : \tau[A]) \times \tau[B \ a] \\ \tau^\Pi[(A, B)] &= (a : \tau[A]) \Rightarrow \tau[B \ a] \end{aligned}$$

Lemma 11. *The assignments $\tau \mapsto \tau^\Pi$ and $\tau \mapsto \tau^\Sigma$ extend to endofunctors \bullet^Π and \bullet^Σ on $[\Delta^1, \mathbb{T}]_{\text{cart}}$.*

Awodey [Awo18] and Newstead [New18] describe how to treat the closure of a type theory under certain connectives as existence of certain cartesian maps.

Definition 12 (Dependent products). A Martin-Löf type theory \mathbb{T} has dependent products iff it is equipped with an algebra $\tau^\Pi \rightarrow \tau : [\Delta^1, \mathbb{T}]_{\text{cart}}$. We will write $\dot{\mathbb{T}}^\Pi \xrightarrow{\text{lam}} \dot{\mathbb{T}}$ and $\mathbb{T}^\Pi \xrightarrow{\text{pi}} \mathbb{T}$ for the upstairs and downstairs components of this algebra respectively. \heartsuit

Definition 13 (Dependent sums). A Martin-Löf type theory \mathbb{T} has dependent sums iff it is equipped with an algebra $\tau^\Sigma \rightarrow \tau : [\Delta^1, \mathbb{T}]_{\text{cart}}$. We will write $\dot{\mathbb{T}}^\Sigma \xrightarrow{\text{pair}} \dot{\mathbb{T}}$ and $\mathbb{T}^\Sigma \xrightarrow{\text{sg}} \mathbb{T}$ for the upstairs and downstairs components of this algebra respectively. \heartsuit

Definition 14 (Universe à la Tarski). Universes à la Tarski are given by an element $\diamond \xrightarrow{\mathbf{u}} \mathbb{T}$ together with a decoding map $\tau[\mathbf{u}] \xrightarrow{\text{dec}} \mathbb{T}$; we will write \mathbf{U} for $\tau[\mathbf{u}]$. (Further structures will be imposed in Definition 15.) \heartsuit

⁵The reader may consult Newstead [New18] and Awodey [Awo18] for more categorical presentations in terms of operations on polynomial endofunctors.

From a universe à la Tarski, we obtain a new representable map ϖ by pullback:

$$\begin{array}{ccc} \dot{\mathbf{U}} & \longrightarrow & \dot{\mathbf{T}} \\ \varpi \downarrow & \lrcorner & \downarrow \tau \\ \mathbf{U} & \xrightarrow{\text{dec}} & \mathbf{T} \end{array}$$

Definition 15. A universe à la Tarski is *weakly* closed under dependent product when it is equipped with an algebra $\varpi^\Pi \rightarrow \varpi : [\Delta^1, \mathbb{T}]_{\text{cart}}$. We write $\mathbf{U}^\Pi \xrightarrow{\widehat{\text{pi}}} \mathbf{U}$ for the downstairs component of this algebra. The closure is called *strict* when the following square commutes on the nose in \mathbb{T} :

$$\begin{array}{ccc} \mathbf{U}^\Pi & \xrightarrow{\widehat{\text{pi}}} & \mathbf{U} \\ \text{dec}^\Pi \downarrow & & \downarrow \text{dec} \\ \mathbf{T}^\Pi & \xrightarrow{\text{pi}} & \mathbf{T} \end{array} \quad \bullet$$

Remark 16. By Lemma 11, we see that the leftmost map and thence the composite map are cartesian below:

$$\varpi^\Pi \xrightarrow{\text{dec}^\Pi} \tau^\Pi \xrightarrow{\text{pi}} \tau$$

Because ϖ^Π is *also* the pullback of τ along $\widehat{\text{pi}} \circ \text{dec}$, we obtain an isomorphism between the elements of the decoding of the Π -code and the elements of the dependent product of the decoding of the underlying family. The role of the diagram in Definition 15 is, then, to make this identification descend to the level of codes. \bullet

We may analogously define the (weak, strict) closure of a universe à la Tarski under dependent sums. Henceforth, we write \mathbb{T} for the least Martin-Löf type theory which has both dependent product and sum types, as well as a universe à la Tarski closed under dependent product and sum. We will work parametrically in whether this universe is strictly or weakly closed under dependent product and sum.

3. THE SEMANTIC GLUING CONSTRUCTION

In most instances of Artin gluing, one begins with a functor $\mathcal{C} \xrightarrow{F} \mathcal{E}$; then, the “gluing of \mathcal{C} along F ” is the comma category $\mathcal{E} \downarrow F$. Artin gluing theorems show that if \mathcal{C}, \mathcal{E} are (e.g.) topoi and F satisfies certain conditions (such as preservation of pullbacks), then the gluing $\mathcal{E} \downarrow F$ satisfies the same conditions as \mathcal{C} and these conditions are preserved by the canonical projection $\mathcal{E} \downarrow F \rightarrow \mathcal{C}$. Artin gluing has been developed for many kinds of categories, summarized in Table 1.

It is reasonable to wonder what conditions are required on a functor $\mathcal{C} \xrightarrow{F} \mathcal{E}$ from the category of contexts \mathcal{C} of a model of type theory $\mathcal{M}_{\mathcal{C}}$ to obtain a good gluing, in the sense that the gluing category $\mathcal{E} \downarrow F$ can be equipped with the structure of a model of type theory. In this section and Section 4, we show that *flatness* is a sufficient condition on F to evince a good notion of gluing for models of type theory.⁶

⁶Some authors use the qualified term “internally flat” for this condition.

Domain conditions	Codomain conditions	Morphism conditions
Grothendieck topos	Grothendieck topos	accessible & preserves finite limits [AGV72]
elementary topos	elementary topos	preserves finite limits [Wra74]
quasitopos	quasitopos	preserves pullbacks [CJ95]
cartesian closed	cartesian closed, has pullbacks	preserves finite products. [CJ95]
supports \mathbb{T} -model	supports \mathbb{T} -model	pseudo-morphism of natural models [KHS19]
supports \mathbb{T}-model	Grothendieck topos	flat (this paper)

TABLE 1. If \mathcal{C} satisfies the domain conditions, \mathcal{E} satisfies the codomain conditions, and F satisfies the morphism conditions, then the gluing $\mathcal{E} \downarrow F$ satisfies the domain conditions and the canonical projection $\mathcal{E} \downarrow F \rightarrow \mathcal{C}$ preserves them. In each case, the condition that the morphism preserve finite limits may be weakened to preservation of pullbacks [CJ95].

The topo-logic of flat functors. Considering that the categories of contexts of type theories possess very few finite limits, one does not expect that finite continuity would be a strong enough condition for a functor $\mathcal{C} \xrightarrow{F} \mathcal{E}$ to induce a new model of type theory by gluing, in contrast to the case where F is a functor between finitely complete categories. It is therefore appropriate to demand that F preserve not only the finite limits that exist, but additionally even the finite limits that don't exist, an intuition which is formalized by the condition of *flatness* [nLa19a; Bor10].

Anel and Joyal advance a dual *algebraic* view on topoi, speaking of the category $\mathcal{Q}\text{ogos} \simeq \mathcal{T}\text{opos}^{\text{op}}$ — a logoi is the same as a topos, but one emphasizes the inverse image part of a geometric morphism rather than the direct image [AJ19]. A morphism of logoi is accordingly called an *algebraic* morphism, by analogy with *geometric* morphisms of topoi.

Now, an algebraic morphism of logoi is nothing more than a left exact and cocontinuous functor between the underlying categories. Diaconescu's theorem [Bor10] states that $\mathbf{DF}_{\mathcal{C}}$ is the *classifying topos* (or “co-classifying logoi”) of the theory of flat functors, so an algebraic morphism $F : \mathbf{DF}_{\mathcal{C}} \rightarrow \mathcal{E}$ corresponds exactly to an essentially unique flat functor $F_{\mathcal{C}} = F \circ \mathcal{F}_{\mathcal{C}} : \mathcal{C} \rightarrow \mathcal{E}$.

Therefore, if we adopt the perspective that a “model of type theory” happens not in \mathcal{C} but rather in $\mathbf{DF}_{\mathcal{C}}$, it is appropriate to speak of “gluing models of type theory along algebraic morphisms of logoi”.

Gluing $\mathbf{DF}_{\mathcal{C}}$ along an algebraic morphism. Given $\mathcal{M}_{\mathcal{C}}$, a model of type theory whose category of contexts is \mathcal{C} , for any algebraic morphism $\mathbf{DF}_{\mathcal{C}} \xrightarrow{F} \mathcal{E} : \mathcal{Q}\text{ogos}$, we may show that the comma logoi $\mathcal{G} = \mathcal{E} \downarrow F$ supports a universe closed under the type-theoretic connectives specified by \mathbb{T} .

Unfortunately, we do *not* obtain from this semantic construction a projection functor $\mathcal{G} \rightarrow \mathcal{C}$, considering that the gluing fibration $\mathcal{G} \xrightarrow{\text{gl}} \mathbf{DF}_{\mathcal{C}}$ is over the category of discrete fibrations on \mathcal{C} rather than \mathcal{C} itself. However, we will show in Section 4 that from our semantic constructions in \mathcal{G} , we obtain a model of type theory $\mathcal{M}_{\mathcal{G}_K} : \mathbf{Mod}_{\mathbb{T}}$, whose category of contexts $\mathcal{G}_K = \mathcal{E} \downarrow F_{\mathcal{C}}$ is the gluing of \mathcal{C} along the flat functor corresponding to the algebraic morphism $\mathbf{DF}_{\mathcal{C}} \xrightarrow{F} \mathcal{E}$.

Concretely, the gluing category $\mathcal{G} = \mathcal{E} \downarrow F$ may be obtained from the following (strict) pullback in \mathbf{Cat} , writing $[\Delta^1, \mathcal{E}] \xrightarrow{\partial_1} \mathcal{E}$ for the codomain fibration of \mathcal{E} :

$$\begin{array}{ccc} \mathcal{G} & \longrightarrow & [\Delta^1, \mathcal{E}] \\ \text{gl} \downarrow & \lrcorner & \downarrow \partial_1 \\ \mathbf{DF}_c & \xrightarrow{F} & \mathcal{E} \end{array}$$

By pulling back further along the Yoneda embedding, we obtain a category of “compact” computability families which lie over genuine contexts $\mathcal{C} : \mathcal{C}$:

$$\begin{array}{ccccc} \mathcal{G}_k & \xrightarrow{K} & \mathcal{G} & \longrightarrow & [\Delta^1, \mathcal{E}] \\ \text{gl}_k \downarrow & \lrcorner & \downarrow \text{gl} & \lrcorner & \downarrow \partial_1 \\ \mathcal{C} & \xrightarrow{\mathcal{J}_c} & \mathbf{DF}_c & \xrightarrow{F} & \mathcal{E} \\ & \searrow & \text{---} & \nearrow & \\ & & F_c & & \end{array}$$

Definition 1 (Computability family). We refer to the objects $E_P \xrightarrow{P} F(X) : \mathcal{E}$ of \mathcal{G} as *computability families*; it is important to note that these computability families lie over arbitrary discrete fibrations X on \mathcal{C} , not just those which are either representable or otherwise correspond to type-theoretic structure in the model \mathcal{M}_c .

A computability family which lies over a representable object (i.e. a computability family in the essential image of K) is called *compact*. \clubsuit

Proposition 2 (Artin, Grothendieck, and Verdier [AGV72]). *The category \mathcal{G} of computability families is a Grothendieck logos and therefore cocomplete, with $\mathcal{G} \xrightarrow{\text{gl}} \mathbf{DF}_c$ a logical morphism of logoi. Moreover, the gluing fibration has both left and right adjoints [Tay99] which are each sections. \clubsuit*

A representable map category over \mathcal{G} . We now equip \mathcal{G} with the structure of a representable map category. Define a wide subcategory \mathcal{R}_G as follows: an arrow $X \xrightarrow{f} Y : \mathcal{G}$ is in \mathcal{R}_G iff every fiber of f at a compact computability family is compact. In other words, fixing $K(\Gamma) \xrightarrow{x} X$ we require that the pullback of f along x is compact:

$$\begin{array}{ccc} K(Y[x]) & \longrightarrow & Y \\ K(p_x) \downarrow & \lrcorner & \downarrow f \\ K(\Gamma) & \xrightarrow{x} & X \end{array}$$

It is a simple matter to verify that this class of maps satisfies the axioms of a representable map category, and that $\mathcal{G} \xrightarrow{\text{gl}} \mathbf{DF}_c$ preserves and reflects representable maps. The rest of this section will be oriented toward constructing a representable map functor $\mathbb{T} \xrightarrow{\mathcal{G}[-]} \mathcal{G}$ which can be used to construct a *gluing model* of type theory in Section 4. We will

additionally need the following triangle to commute up to a canonical natural isomorphism χ_\bullet in Cat :

$$\begin{array}{ccc} \mathbb{T} & \xrightarrow{\mathcal{G}[-]} & \mathcal{G} \\ & \searrow \mathcal{M}_e & \swarrow \mathcal{G}! \\ & \text{DF}_e & \end{array} \quad \chi_\bullet$$

Because \mathbb{T} can be seen to be generated by a signature of clauses (e.g. “There is a representable map $\dot{\mathbb{T}} \xrightarrow{\tau} \mathbb{T}$ ”), we may construct such a pair $(\mathcal{G}[-], \chi_\bullet)$ by considering only these generating clauses, following Theorem 5.17 of Uemura [Uem19].

3.1. A universe of computability families. Because \mathcal{E} is a Grothendieck logoi, we may find a large enough universe $\dot{\mathcal{U}}_{\mathcal{E}} \xrightarrow{\text{el}_{\mathcal{U}_{\mathcal{E}}}} \mathcal{U}_{\mathcal{E}} : \mathcal{E}$ closed under dependent product, dependent sum, subobjects, quotients, etc. [Str05], containing in addition a code for $F(\tau_e)$. We will use this to define $\mathcal{G}[\dot{\mathbb{T}} \xrightarrow{\tau} \mathbb{T}]$ lying directly over $(\dot{\mathbb{T}} \xrightarrow{\tau} \mathbb{T})_e$ (which is to say, we may define $\chi_{\dot{\mathbb{T}}} = \text{id}_{\dot{\mathbb{T}}}$ and $\chi_{\mathbb{T}} = \text{id}_{\mathbb{T}}$, etc.).

Notation 3. To save space, we write $\mathbb{T} \xrightarrow{\perp\!\!\!\lrcorner} \mathcal{E}$ for the composite of $\mathbb{T} \xrightarrow{\mathcal{M}_e} \text{DF}_e$ with $\text{DF}_e \xrightarrow{F} \mathcal{E}$. \heartsuit

Remark 4. Let $\mathcal{X} \xrightarrow{G} \mathcal{E}$ be any functor, and let $f^!g : \mathcal{X}/Z$ be the dependent sum of $X \xrightarrow{g} Y$ along $Y \xrightarrow{f} Z$; then, we have $G(f^!g) = G(f)^!G(g)$. This is because dependent sums are formed by composition, and functors obviously preserve composition. \heartsuit

Lemma 5. Let $\mathcal{X} \xrightarrow{G} \mathcal{E}$ be any left exact functor, and let $f_*g : \mathcal{X}/Z$ be the dependent product of $X \xrightarrow{g} Y$ along $Y \xrightarrow{f} Z$. Then we have the following canonical comparison map:

$$G(f_*g) \dashrightarrow G(f)_*G(g)$$

Proof. It is simple enough to calculate the comparison map in the “adjoint sequent calculus”:

$$\begin{array}{c} \overline{\overline{f_*g \rightarrow f_*g}} \text{ identity} \\ \overline{f^*f_*g \rightarrow g} \quad f^* \dashv f_* \\ \overline{G(f^*f_*g) \rightarrow G(g)} \quad G(-) \\ \overline{G(f)^*G(f_*g) \rightarrow G(g)} \quad G \text{ left exact} \\ \overline{G(f_*g) \rightarrow G(f)_*G(g)} \quad G(f)^* \dashv G(f)_* \end{array} \quad \square$$

Corollary 6. We have the following canonical comparison maps, not generally invertible:

$$\begin{array}{ccc} [\dot{\mathbb{T}}^\Pi] \xrightarrow{(-)} [\dot{\mathbb{T}}]^\Pi & & [\dot{\mathbb{T}}^\Sigma] \xrightarrow{(-)} [\dot{\mathbb{T}}]^\Sigma \\ \downarrow [\tau^\Pi] & & \downarrow [\tau^\Sigma] \\ [\mathbb{T}^\Pi] \xrightarrow{(-)} [\mathbb{T}]^\Pi & & [\mathbb{T}^\Sigma] \xrightarrow{(-)} [\mathbb{T}]^\Sigma \end{array}$$

Proof. By Remark 4 and Lemma 5, using the fact that $\mathbb{T} \xrightarrow{\perp\!\!\!\lrcorner} \mathcal{E}$ is left exact. \square

Notation 7. In light of Corollary 6, we fix the following notations in the fibers.

- (1) Given $G : [\mathbf{T}^\Pi]$ and $g : [\tau^\Pi][G]$, we will write $(g) : [\tau]^\Pi[(G)]$, i.e. $(g) : (a : [\tau][[(G)_0]] \Rightarrow [\tau][[(G)_1] a])$.
- (2) Given $G : [\mathbf{T}^\Sigma]$ and $g : [\tau^\Sigma][G]$, we will write $(g) : [\tau]^\Sigma[(G)]$, i.e. $(g) : (a : [\tau][[(G)_0]] \times [\tau][[(G)_1] a])$. \clubsuit

Construction 8 (Universe of computability families). We define a computability family $\mathcal{G}[\mathbf{T}]$ over $\mathbf{T}_\mathcal{E}$ fiberwise in the internal language of \mathcal{E} . Fixing $A : [\mathbf{T}]$, we define the fiber $\mathcal{G}[\mathbf{T}][A]$ in \mathcal{E} to be the function space $[\tau][A] \Rightarrow \mathbf{U}_\mathcal{E}$.

The family $\mathcal{G}[\dot{\mathbf{T}}]$ which we are now defining must lie over $\dot{\mathbf{T}}_\mathcal{E}$; considering that we must eventually define both maps $E_{\mathcal{G}[\dot{\mathbf{T}}]} \xrightarrow{\mathcal{G}[\tau]} E_{\mathcal{G}[\mathbf{T}]}$, $E_{\mathcal{G}[\dot{\mathbf{T}}]} \xrightarrow{\mathcal{G}[\dot{\mathbf{T}}]} [\dot{\mathbf{T}}]$, we do so simultaneously by considering the fiber over the pullback of $\mathcal{G}[\mathbf{T}]$ along $[\tau]$:

$$(3.9) \quad \begin{array}{ccc} & E_{\mathcal{G}[\dot{\mathbf{T}}]} & \\ & \searrow \mathcal{G}[\tau] & \swarrow \mathcal{G}[\tau] \\ & E_{\mathcal{G}[\mathbf{T}]} \times_{[\mathbf{T}]} [\dot{\mathbf{T}}] & \rightarrow E_{\mathcal{G}[\mathbf{T}]} \\ \mathcal{G}[\dot{\mathbf{T}}] & \downarrow \lrcorner & \downarrow \mathcal{G}[\mathbf{T}] \\ & [\dot{\mathbf{T}}] & \xrightarrow{[\tau]} [\mathbf{T}] \end{array}$$

Fixing $\mathfrak{A} : \mathcal{G}[\mathbf{T}][A]$, $a : [\tau][A]$, we define $\mathcal{G}[\tau][A, \mathfrak{A}, a]$ to be the type $\mathfrak{A} a : \mathbf{U}_\mathcal{E}$. As in Diagram 3.9, this determines both $E_{\mathcal{G}[\dot{\mathbf{T}}]} \xrightarrow{\mathcal{G}[\tau]} E_{\mathcal{G}[\mathbf{T}]}$ and $E_{\mathcal{G}[\dot{\mathbf{T}}]} \xrightarrow{\mathcal{G}[\dot{\mathbf{T}}]} [\dot{\mathbf{T}}]$ by composition. \clubsuit

3.2. Dependent product and sum. The representable maps $\mathcal{G}[\tau^\Pi]$, $\mathcal{G}[\tau^\Sigma]$ are defined compositionally as $\mathcal{G}[\tau]^\Pi$, $\mathcal{G}[\tau]^\Sigma$ respectively. The local cartesian closure of $\mathcal{G} \xrightarrow{\mathfrak{gl}} \mathbf{DF}_\mathcal{E}$ determines canonical isomorphisms $\chi_{\mathbf{T}^\Pi}$, $\chi_{\dot{\mathbf{T}}^\Pi}$, $\chi_{\mathbf{T}^\Sigma}$, $\chi_{\dot{\mathbf{T}}^\Sigma}$ and the appropriate naturality squares relative to $\tau_\mathcal{E}^\Pi$, $\tau_\mathcal{E}^\Sigma$.

In order to close the universe $\mathcal{G}[\dot{\mathbf{T}}] \xrightarrow{\mathcal{G}[\tau]} \mathcal{G}[\mathbf{T}]$ under dependent products and sums, it will be useful to *compute* the families $\mathcal{G}[\dot{\mathbf{T}}]^\Pi \xrightarrow{\mathcal{G}[\tau]^\Pi} \mathcal{G}[\mathbf{T}]^\Pi$ and $\mathcal{G}[\dot{\mathbf{T}}]^\Sigma \xrightarrow{\mathcal{G}[\tau]^\Sigma} \mathcal{G}[\mathbf{T}]^\Sigma$ in the language of \mathcal{E} ; this computation is displayed in Figure 1.

Construction 10 (Dependent product). We define a cartesian map $\mathcal{G}[\tau]^\Pi \rightarrow \mathcal{G}[\tau] : [\Delta^1, \mathcal{G}]_{\text{cart}}$ which lies over the algebra $\tau_\mathcal{E}^\Pi \rightarrow \tau_\mathcal{E}$ modulo the canonical isomorphisms mentioned above; we accomplish this directly by defining computability families *strictly* over the desired constituents of $\mathcal{M}_\mathcal{E}$, which may then be adjusted by χ_\bullet appropriately. Unfolding into the language of \mathcal{E} , we must construct the upstairs and downstairs arrows

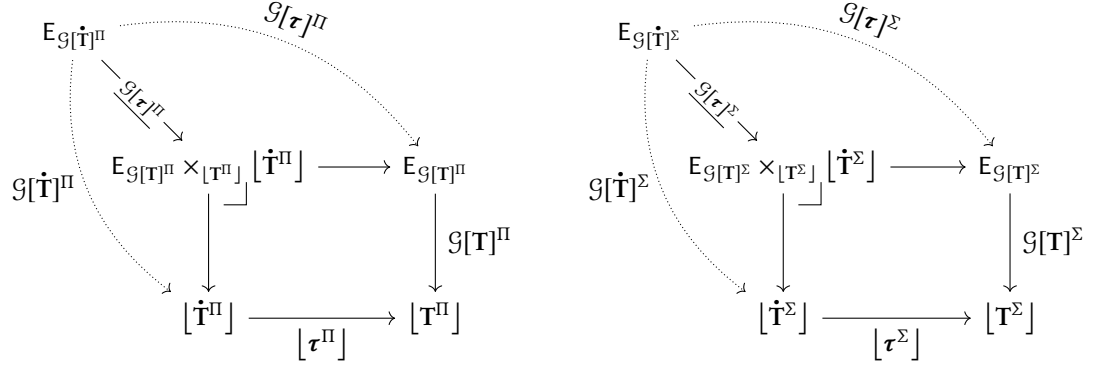
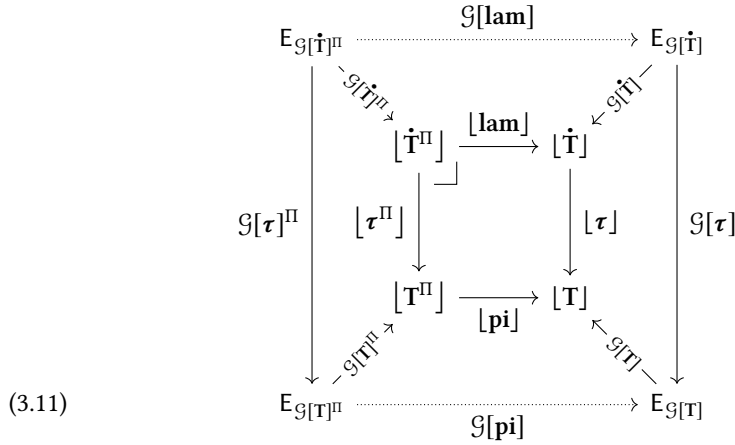


FIGURE 1. Computation of the generic dependent product and sum from \mathcal{G} in \mathcal{E} . Note that this is not a definition, but rather a construction of a family which is universally determined up to isomorphism.

of Diagram 3.11 below.



To define $\mathcal{G}[\mathbf{pi}]$ over $\lfloor \mathbf{pi} \rfloor$, we fix $G : [\mathbf{T}^\Pi]$ and $\mathcal{G} : \mathcal{G}[\mathbf{T}^\Pi]$ over G , and must choose an element of $\mathcal{G}[\mathbf{T}][\lfloor \mathbf{pi} \rfloor G]$, i.e. a $\mathbf{U}_\mathcal{E}$ -valued family indexed in $\lfloor \tau \rfloor[\lfloor \mathbf{pi} \rfloor G]$. Fixing $g : \lfloor \tau \rfloor[\lfloor \mathbf{pi} \rfloor G]$, we choose the following type in $\mathbf{U}_\mathcal{E}$:

$$\mathcal{G}[\mathbf{pi}][G] \mathcal{G} g \triangleq (a : \lfloor \tau \rfloor[\langle G \rangle_0]) (\alpha : \mathcal{G}_0 a) \Rightarrow \mathcal{G}_1 a \alpha (g a)$$

Next, we must define $\mathcal{G}[\mathbf{lam}]$ over $\lfloor \mathbf{lam} \rfloor$. Fixing $G : [\mathbf{T}^\Pi]$, $g : \lfloor \tau^\Pi \rfloor[G]$, $\mathcal{G} : \mathcal{G}[\mathbf{T}^\Pi]$ over G , and $g : \mathcal{G}[\tau]^\Pi[\mathcal{G}]$ over g , and considering Construction 8, we must choose an element

of the following collection:

$$\begin{aligned} & \underline{\mathcal{G}[\tau][[\text{pi}]G, \mathcal{G}[\text{pi}][G] \mathfrak{G}, [\text{lam}]g]} \\ & \cong \mathcal{G}[\text{pi}][G] \mathfrak{G} ([\text{lam}]g) \\ & \cong (a : [\tau][(\mathcal{G})_0]) (a : \mathfrak{G}_0 a) \Rightarrow \mathfrak{G}_1 a a (([\text{lam}]g) a) \end{aligned}$$

We choose the following element:

$$\mathcal{G}[\text{lam}][G, g] \mathfrak{G} g a a \triangleq g a a \quad \heartsuit$$

Construction 12 (Dependent sum). The universe of computability families is closed under dependent sums lying essentially over the corresponding algebra in $\mathcal{M}_{\mathcal{C}}$. As in Construction 10, we must construct the upstairs and downstairs arrows of Diagram 3.13 below.

$$(3.13) \quad \begin{array}{ccc} & \mathcal{G}[\text{pair}] & \\ E_{\mathcal{G}[\dot{\mathbf{T}}]^\Sigma} & \xrightarrow{\quad} & E_{\mathcal{G}[\dot{\mathbf{T}}]} \\ \downarrow \mathcal{G}[\tau]^\Sigma & \begin{array}{c} \dashrightarrow \mathcal{G}[\dot{\mathbf{T}}]^\Sigma \rightarrow [\dot{\mathbf{T}}]^\Sigma \xrightarrow{[\text{pair}]} [\dot{\mathbf{T}}] \xleftarrow{\mathcal{G}[\dot{\mathbf{T}}]} \\ \downarrow [\tau]^\Sigma \quad \downarrow [\tau] \\ [\mathbf{T}]^\Sigma \xrightarrow{[\text{sg}]} [\mathbf{T}] \xleftarrow{\mathcal{G}[\mathbf{T}]} \end{array} & \downarrow \mathcal{G}[\tau] \\ & \mathcal{G}[\text{sg}] & \\ E_{\mathcal{G}[\mathbf{T}]^\Sigma} & \xrightarrow{\quad} & E_{\mathcal{G}[\mathbf{T}]} \end{array}$$

To define $\mathcal{G}[\text{sg}]$ over $[\text{sg}]$, we fix $G : [\mathbf{T}^\Sigma]$ and $\mathfrak{G} : \mathcal{G}[\mathbf{T}]^\Sigma$ over G , and must choose an element of $\mathcal{G}[\mathbf{T}][[\text{sg}]G]$, i.e. a U_ε -valued family indexed in $[\tau][[\text{sg}]G]$. Fixing $g : [\tau][[\text{sg}]G]$, we choose the following type in U_ε :

$$\mathcal{G}[\text{sg}][G] \mathfrak{G} g \triangleq (a : \mathfrak{G}_0 (g)_0) \times \mathfrak{G}_1 (g)_0 a (g)_1$$

Next, we must define $\mathcal{G}[\text{pair}]$ over $[\text{pair}]$. Fixing $G : [\mathbf{T}^\Sigma]$, $g : [\tau]^\Sigma[G]$, $\mathfrak{G} : \mathcal{G}[\mathbf{T}]^\Sigma$ over G , $g : \mathcal{G}[\tau]^\Sigma[\mathfrak{G}]$ over g , and considering Construction 8, we must choose an element of $\underline{\mathcal{G}[\tau][[\text{sg}]G, \mathcal{G}[\text{sg}][G] \mathfrak{G}, [\text{pair}]g]}$. We choose the following:

$$\mathcal{G}[\text{pair}][G, g] \mathfrak{G} g \triangleq (g_0, g_1) \quad \heartsuit$$

3.3. Universe à la Tarski.

Assumption 14. In this section we assume a universe à la Tarski ($u_\varepsilon : U_\varepsilon, \text{dec}_\varepsilon : \text{el}_{U_\varepsilon}[u_\varepsilon] \rightarrow U_\varepsilon$) closed under dependent sum and product in the sense of Definition 15, and large enough to contain a code for $[\tau]$. \heartsuit

In Lemma 17, we will prove that the (strict, weak) closure of u_ε under dependent product and sum can be lifted to a glued universe which is (strictly, weakly) closed under dependent product and sum.

Remark 15. Following Streicher [Str05], we *always* have such universes à la Tarski for Grothendieck logoi, weakly closed under connectives; there is some uncertainty in the literature whether these universes may be *strictly* closed under connectives for general

sheaf logoi, but the strictness is always easy to obtain by a direct construction in the case of presheaves (which covers most instances of gluing for metatheoretic purposes, e.g. Examples 4 to 6).

Alternatively, (small) induction-recursion may always be used to obtain a suitably strict universe in a Grothendieck logoi, as employed by Sterling, Angiuli, and Gratzter [SAG19]; small induction-recursion is a formal schema for the construction of certain initial algebras [Han+13], which are known to exist in sheaf logoi following Moerdijk and Palmgren [MP00] (using the fact that small induction recursion can be reduced to indexed inductive families). \heartsuit

Construction 16 (Glued universe à la Tarski). We will define a global element $\mathcal{G}[\diamond] \xrightarrow{\mathcal{G}[\mathbf{u}]} \mathcal{G}[\mathbf{T}]$ lying strictly over $(\diamond \xrightarrow{\mathbf{u}} \mathbf{T})_{\mathcal{C}}$ and a decoding map $\mathcal{G}[\mathbf{U}] \xrightarrow{\mathcal{G}[\mathbf{dec}]} \mathcal{G}[\mathbf{T}]$ lying over $(\tau[\mathbf{u}] \xrightarrow{\mathbf{dec}} \mathbf{T})_{\mathcal{C}}$ which, as a universe à la Tarski, is (strictly, weakly) closed under dependent product and sum. (Note that $\mathcal{G}[\mathbf{U}]$ is $\mathcal{G}[\tau][\mathcal{G}[\mathbf{u}]]$ because $\mathcal{G}[-]$ preserves pullbacks.) These are formulated in the internal language of \mathcal{E} .

The computability family of the universe $\mathcal{G}[\mathbf{u}]$ is an element of $\mathcal{G}[\mathbf{T}][[\mathbf{u}]] = [\tau][[\mathbf{u}]] \Rightarrow \mathbf{U}_{\mathcal{E}}$. Considering that $[\tau][[\mathbf{u}]]$ is $[\tau[\mathbf{u}]]$, we define $\mathcal{G}[\mathbf{u}] \hat{A}$ to be the function space $[\tau][[\mathbf{dec}]\hat{A}] \Rightarrow u_{\mathcal{E}} : \mathbf{U}_{\mathcal{E}}$. We define the decoding of $\hat{\mathfrak{A}} : \mathcal{G}[\mathbf{u}] \hat{A}$ to be $\mathbf{dec}(\hat{\mathfrak{A}}) \triangleq \mathbf{dec}_{\mathcal{E}} \circ \hat{\mathfrak{A}}$. \heartsuit

Lemma 17 (Closure under dependent product and sum). *If $(u_{\mathcal{E}}, \mathbf{dec}_{\mathcal{E}})$ is (strictly, weakly) closed under dependent product and sum relative to $\mathbf{U}_{\mathcal{E}}$, then so is the glued universe à la Tarski $(\mathcal{G}[\mathbf{u}], \mathcal{G}[\mathbf{dec}])$ relative to $\mathcal{G}[\mathbf{T}]$.*

Proof. To close it under dependent products, we must define $\mathcal{G}[\hat{\mathbf{pi}}]$ lying over $\hat{\mathbf{pi}}_{\mathcal{C}}$. Fixing a code $\hat{G} : [\mathbf{U}^{\Pi}]$ over $G : [\mathbf{T}^{\Pi}]$ and a computability family $\hat{\mathfrak{G}} : \mathcal{G}[\mathbf{U}]^{\Pi}$ over \hat{G} , we must construct a function $[\tau][[\hat{\mathbf{pi}}] G] \Rightarrow u_{\mathcal{E}}$ which is (equal, pointwise isomorphic) to $\mathcal{G}[\hat{\mathbf{pi}}][G]$ ($\mathcal{G}[\mathbf{dec}]^{\Pi} \hat{\mathfrak{G}}$) when followed by $\mathbf{dec}_{\mathcal{E}}$. We send each $g : [\tau][[\hat{\mathbf{pi}}] G]$ to the following:

$$(a : [\tau][[(G)_0]]) (a : \hat{\mathfrak{G}}_0 a) \Rightarrow \hat{\mathfrak{G}}_1 a \ a \ (g \ a) : u_{\mathcal{E}}$$

The case for dependent sums is analogous: fixing $\hat{G} : [\mathbf{U}^{\Sigma}]$ over $G : [\mathbf{T}^{\Sigma}]$ and $\hat{\mathfrak{G}} : \mathcal{G}[\mathbf{U}]^{\Sigma}$ over \hat{G} , we construct a function $[\tau][[\hat{\mathbf{sg}}] G] \Rightarrow u_{\mathcal{E}}$ which is (equal, pointwise isomorphic) to $\mathcal{G}[\hat{\mathbf{sg}}][G]$ ($\mathcal{G}[\mathbf{dec}]^{\Sigma} \hat{\mathfrak{G}}$) when followed by $\mathbf{dec}_{\mathcal{E}}$. For each $g : [\tau][[\hat{\mathbf{sg}}] G]$, we choose:

$$(a : \hat{\mathfrak{G}}_0 (g)_0) \times \hat{\mathfrak{G}}_1 (g)_0 \ a \ (g)_1 : u_{\mathcal{E}} \quad \square$$

In summary, we have closed our semantic universe à la Tarski under algebras $\mathcal{G}[\mathbf{U}]^{\Pi} \xrightarrow{\mathcal{G}[\hat{\mathbf{pi}}]} \mathcal{G}[\mathbf{U}]$, $\mathcal{G}[\mathbf{U}]^{\Sigma} \xrightarrow{\mathcal{G}[\hat{\mathbf{sg}}]} \mathcal{G}[\mathbf{U}]$ which lie over $(\mathbf{U}^{\Pi} \xrightarrow{\hat{\mathbf{pi}}} \mathbf{U})_{\mathcal{C}}$ and $(\mathbf{U}^{\Sigma} \xrightarrow{\hat{\mathbf{sg}}} \mathbf{U})_{\mathcal{C}}$ respectively such that $\mathcal{G}[\mathbf{dec}] \circ \mathcal{G}[\hat{\mathbf{pi}}] = \mathcal{G}[\hat{\mathbf{pi}}] \circ \mathcal{G}[\mathbf{dec}]^{\Pi}$ and $\mathcal{G}[\mathbf{dec}] \circ \mathcal{G}[\hat{\mathbf{sg}}] = \mathcal{G}[\hat{\mathbf{sg}}] \circ \mathcal{G}[\mathbf{dec}]^{\Sigma}$.

4. THE COMPUTABILITY MODEL OF TYPE THEORY

In this section, we will transform the semantic gluing construction from Section 3 into a model of type theory. Starting from the Artin gluing situation for the algebraic morphism $\mathbf{DF}_{\mathcal{C}} \xrightarrow{F} \mathcal{E}$, we recall that we may restrict further along the Yoneda embedding to construct the gluing of \mathcal{C} along the essentially unique flat functor $\mathcal{C} \xrightarrow{F_{\mathcal{C}}} \mathcal{E}$.

It is justified to refer to the restricted gluing category $\mathcal{G}_{\mathcal{K}}$ as the category of *compact computability families*; indeed, the embedding functor $\mathcal{G}_{\mathcal{K}} \xrightarrow{\hookrightarrow} \mathcal{G}$ is *dense*, a consequence of the cocontinuity of F and the universality of colimits in \mathcal{E} .

Lemma 1 (Density). *Every computability family $X : \mathcal{G}$ is canonically a colimit of computability families which lie in the image of \mathbf{K} , i.e. compact computability families.*

The main result of this section will be to exhibit a *gluing model* of type theory $\mathcal{M}_{\mathcal{G}_K} : \mathbf{Mod}_{\mathbb{T}}$ over the compact computability families (Construction 4), together with a homomorphism of models $\mathcal{M}_{\mathcal{G}_K} \rightarrow \mathcal{M}_{\mathcal{C}} : \mathbf{Mod}_{\mathbb{T}}$ tracked by the fibration $\mathcal{G}_K \xrightarrow{\mathbf{gl}_K} \mathcal{C}$ (Lemma 5).

The nerve of a computability family. The dense subcategory $\mathcal{G}_K \xrightarrow{\mathbf{K}} \mathcal{G}$ induces a fully faithful *nerve* operation $\mathcal{G} \xrightarrow{\mathbf{N}_K} \mathbf{DF}_{\mathcal{G}_K}$ which takes a computability family and “splays it out” as a discrete fibration over compact computability families; the fiber of the nerve $\mathbf{N}_K(X)$ of a computability family at a compact computability family Γ is the set of morphisms of computability families from $\mathbf{K}(\Gamma)$ to X :

$$\mathbf{DF}_{\mathcal{G}_K}[\mathbf{K}\Gamma, \mathbf{N}_K(X)] \cong \mathcal{G}[\mathbf{K}(\Gamma), X]$$

Moreover, $\mathcal{G} \xrightarrow{\mathbf{N}_K} \mathbf{DF}_{\mathcal{G}_K}$ has a left adjoint $\mathbf{DF}_{\mathcal{G}_K} \xrightarrow{|\cdot|_K} \mathcal{G}$, the *realization* of a discrete fibration in computability families. The adjunction $|\cdot|_K \dashv \mathbf{N}_K$ restricts to an equivalence between categories of “generating” objects, namely the representable presheaves and the compact computability families. We note a further consequence of the density of compact computability families:

Lemma 2. *The nerve is locally cartesian closed.*

Our idea is to transfer our semantic gluing constructions from \mathcal{G} to $\mathbf{DF}_{\mathcal{G}_K}$ along the nerve functor, obtaining a *gluing model* of type theory over \mathcal{G}_K .

Lemma 3. *The nerve functor $\mathcal{G} \xrightarrow{\mathbf{N}_K} \mathbf{DF}_{\mathcal{G}_K}$ is a representable map functor.*

Construction 4 (The gluing model). By composing with the nerve, we obtain a representable map functor (thence, a model of type theory) $\mathbb{T} \xrightarrow{\mathcal{M}_{\mathcal{G}_K}} \mathbf{DF}_{\mathcal{G}_K}$:

$$\begin{array}{ccc} \mathbb{T} & \xrightarrow{\mathcal{G}[-]} & \mathcal{G} \\ & \searrow \mathcal{M}_{\mathcal{G}_K} & \downarrow \mathbf{N}_K \\ & & \mathbf{DF}_{\mathcal{G}_K} \end{array}$$

A morphism of models. The gluing model $\mathcal{M}_{\mathcal{G}_K}$ must lie over $\mathcal{M}_{\mathcal{C}}$, in the sense that we have a morphism $\mathcal{M}_{\mathcal{G}_K} \rightarrow \mathcal{M}_{\mathcal{C}} : \mathbf{Mod}_{\mathbb{T}}$. We will use the gluing fibration $\mathcal{G}_K \xrightarrow{\mathbf{gl}_K} \mathcal{C}$ for the underlying functor between categories of contexts; it remains to exhibit a natural transformation $\mathcal{M}_{\mathcal{G}_K} \xrightarrow{\mathcal{M}_{\mathbf{gl}_K}} \mathcal{M}_{\mathcal{C}} : [\mathbb{T}, \mathbf{DF}]$ whose components lie over \mathbf{gl}_K and which satisfies the Beck-Chevalley condition on representable maps.

We must construct $\mathcal{M}_{\mathcal{G}_K} \xrightarrow{\mathcal{M}_{\mathbf{gl}_K}} \mathcal{M}_{\mathcal{C}} : [\mathbb{T}, \mathbf{DF}]$ lying pointwise over \mathbf{gl}_K , which is in fact the same as constructing a map $\mathcal{M}_{\mathcal{G}_K} \rightarrow \mathbf{gl}_K^* \circ \mathcal{M}_{\mathcal{C}} : [\mathbb{T}, \mathbf{DF}_{\mathcal{G}_K}]$ by the universal property of the cartesian lift in the fibration $\mathbf{DF} \xrightarrow{\pi_{\mathbf{DF}}} \mathbf{Cat}$. We can see that this map is given essentially by the action of the gluing fibration on morphisms; working fiberwise, this action can be

written as follows for each $J : \mathbb{T}$, noting that $\text{gl} \circ K = \mathfrak{Z} \circ \text{gl}_K$:

$$\begin{array}{c}
 J_{\mathcal{G}_K}(\Gamma) \cong \mathcal{G}[K(\Gamma), \mathcal{G}[J]] \\
 \downarrow \text{gl} \\
 \text{DF}_{\mathcal{C}}[\mathfrak{Z}\text{gl}_K(\Gamma), \text{gl}(\mathcal{G}[J])] \\
 \downarrow (\chi_J)_* \\
 \text{DF}_{\mathcal{C}}[\mathfrak{Z}\text{gl}_K(\Gamma), J_{\mathcal{C}}] \cong J_{\mathcal{C}}(\text{gl}_K(\Gamma))
 \end{array}$$

The above extends to a natural transformation in $J : \mathbb{T}$ by virtue of the naturality of χ_\bullet :

$$\mathcal{M}_{\mathcal{G}_K} = N_K \circ \mathcal{G}[-] \xrightarrow{\widetilde{\mathcal{M}}_{\text{gl}_K}} \text{gl}_K^* \circ \mathcal{M}_{\mathcal{C}} : [\mathbb{T}, \text{DF}_{\mathcal{G}_K}]$$

By transposing, we therefore have a natural transformation $\mathcal{M}_{\mathcal{G}_K} \xrightarrow{\mathcal{M}_{\text{gl}_K}} \mathcal{M}_{\mathcal{C}} : [\mathbb{T}, \text{DF}]$ such that each $X_{\mathcal{G}_K} \xrightarrow{X_{\text{gl}_K}} X_{\mathcal{C}}$ is tracked by $\mathcal{G}_K \xrightarrow{\text{gl}_K} \mathcal{C}$. It remains to check the Beck-Chevalley conditions for the naturality squares at representable maps, which we omit for reasons of space.

Lemma 5 (Morphism of models). *The natural transformation $\mathcal{M}_{\mathcal{G}_K} \xrightarrow{\mathcal{M}_{\text{gl}_K}} \mathcal{M}_{\mathcal{C}} : [\mathbb{T}, \text{DF}]$ exhibits a morphism of models in $\mathbf{Mod}_{\mathbb{T}}$, tracked by $\mathcal{G}_K \xrightarrow{\text{gl}_K} \mathcal{C}$.*

4.1. Display of the gluing model. We will now unfold what the universal property of the bi-initial model $\mathbb{T} \xrightarrow{\mathcal{M}_J} \text{DF}_J$ gives us in relation to the gluing projection. We will write $\mathcal{M}_J \xrightarrow{\mathcal{M}_{\text{ev}}} \mathcal{M}_{\mathcal{G}_K} : \mathbf{Mod}_{\mathbb{T}}$ lying over $J \xrightarrow{\text{ev}} \mathcal{G}_K$ for the essentially unique evaluation map from the bi-initial model into the gluing model.

Diagram 4.6 below commutes in $\mathbf{Mod}_{\mathbb{T}}$ up to a *unique* invertible 2-cell $\mathcal{M}_{\text{id}_J} \xrightarrow{\mathcal{M}_{\alpha}} \mathcal{M}_{\text{gl}_K} \circ \mathcal{M}_{\text{ev}}$.

$$\begin{array}{ccc}
 \mathcal{M}_J & \xrightarrow{\mathcal{M}_{\text{ev}}} & \mathcal{M}_{\mathcal{G}_K} \\
 \searrow \mathcal{M}_{\text{id}_J} & \mathcal{M}_{\alpha} & \swarrow \mathcal{M}_{\text{gl}_K} \\
 & \mathcal{M}_J &
 \end{array}$$

(4.6)

We will write $\mathcal{M}_{\overline{\alpha}}$ for the inverse to this canonical 2-cell.

Display of contexts. Unfolding slightly, the 2-cell \mathcal{M}_{α} is a natural isomorphism between the underlying functors id_J and $\text{gl}_K \circ \text{ev}$ in $\text{Cat}[J, J]$. This means that for every context $\Gamma : J$, we have an isomorphism $\Gamma \xrightarrow{\alpha_{\Gamma}} \text{gl}_K(\text{ev}(\Gamma))$.

Display of judgments. For each $X : \mathbb{T}$, both X_{id_J} and $X_{\text{gl}_K} \circ X_{\text{ev}}$ are fibered endofunctors on $X_J : \text{DF}$ lying over id_J and $\text{gl}_K \circ \text{ev}$ respectively. The canonical natural isomorphism $\text{id}_J \xrightarrow{\alpha} \text{gl}_K \circ \text{ev}$ tracks some (uniquely determined) natural isomorphism between fibered endofunctors:

$$(4.7) \quad X_{\text{id}_J} \xrightarrow{X_{\alpha}} X_{\text{gl}_K} \circ X_{\text{ev}}$$

Fixing $x : X_{\mathcal{J}}$, we consider the component of the natural transformation in Equation 4.7 as an isomorphism in $X_{\mathcal{J}}$ tracked by the appropriate component of α :

$$\begin{array}{ccc}
 x & \xrightarrow{X_{\alpha}^x} & X_{\text{gl}_k}(X_{\text{ev}}(x)) & X_{\mathcal{J}} \\
 \downarrow & & \downarrow & \downarrow \pi \\
 \pi(x) & \xrightarrow{\alpha_{\pi(x)}} & \text{gl}_k(\text{ev}(\pi(x))) & \mathcal{J}
 \end{array}$$

An isomorphism in a discrete fibration is nothing more than a *condition*:

$$(4.8) \quad x = \alpha_{\pi(x)}^* X_{\text{gl}_k}(X_{\text{ev}}(x))$$

$$(4.9) \quad \bar{\alpha}_{\pi(x)}^* x = X_{\text{gl}_k}(X_{\text{ev}}(x))$$

Slogan 10. The natural transformations X_{α} ensure that the difference between type-theoretic structures is totally determined by the action of α on contexts. \clubsuit

Realignment. As we noted above, the glued evaluation $\text{ev}(\Gamma) : \mathcal{G}_K$ lies not over $\Gamma : \mathcal{J}$ but (tautologically) rather over $\text{gl}_k(\text{ev}(\Gamma)) : \mathcal{J}$. By cartesian lift, we obtain a computability family $\llbracket \Gamma \rrbracket$ which lies strictly over the context Γ (though this assignment is only pseudo-functorial):

$$(4.11) \quad \begin{array}{ccc}
 \llbracket \Gamma \rrbracket & \xrightarrow{\alpha_{\Gamma}^{\dagger}} & \text{ev}(\Gamma) & \mathcal{G}_K \\
 \downarrow & & \downarrow & \downarrow \text{gl}_k \\
 \Gamma & \xrightarrow{\alpha_{\Gamma}} & \text{gl}_k(\text{ev}(\Gamma)) & \mathcal{J}
 \end{array}$$

A similar phenomenon occurs in the discrete fibrations: given a judgment $X : \mathbb{T}$ and element $\mathfrak{A}\Gamma \xrightarrow{x} X_{\mathcal{J}} : \mathbf{DF}_{\mathcal{J}}$, the glued evaluation $\mathfrak{A}\text{ev}(\Gamma) \xrightarrow{X_{\text{ev}}(x)} X_{\mathcal{G}_K} : \mathbf{DF}_{\mathcal{G}_K}$ is sent by X_{gl_k} not to x itself but to $x \circ \mathfrak{A}\bar{\alpha}_{\Gamma}$, considering Equation 4.9.

Computing the nerve of a computability family at a representable fiber, we know that $X_{\text{ev}}(x)$ is in fact determined canonically by an element $K(\text{ev}(\Gamma)) \xrightarrow{\widetilde{X_{\text{ev}}(x)}} \mathcal{G}[X] : \mathcal{G}$ in the following configuration:

$$\begin{array}{ccc}
 K(\text{ev}(\Gamma)) & \xrightarrow{\widetilde{X_{\text{ev}}(x)}} & \mathcal{G}[X] & \mathcal{G} \\
 \downarrow & & \downarrow & \downarrow \text{gl} \\
 \mathfrak{A}\text{gl}_k(\text{ev}(\Gamma)) & \xrightarrow{x \circ \mathfrak{A}\bar{\alpha}_{\Gamma}} & X_{\mathcal{J}} & \mathbf{DF}_{\mathcal{J}}
 \end{array}$$

Using α_1^\dagger from Diagram 4.11, we obtain a glued element which lies *strictly* over x :

$$\begin{array}{ccc}
 \mathsf{K}(\llbracket \Gamma \rrbracket) & \xrightarrow{\llbracket x \rrbracket} & \mathcal{G}[X] \\
 \downarrow & & \downarrow \\
 \mathfrak{K}\Gamma & \xrightarrow{x} & X_{\mathcal{J}}
 \end{array}
 \qquad
 \begin{array}{c}
 \mathcal{G} \\
 \downarrow \mathfrak{g}^! \\
 \mathsf{DF}_{\mathcal{J}}
 \end{array}$$

5. CASE STUDY: CLOSED CANONICITY

We demonstrate an initial application of our model construction, to prove the *canonicity* metatheorem for Martin-Löf type theory. In order to state the canonicity theorem, we must have a *base type* whose elements are observable. We could extend \mathbb{T} by the booleans, but it is equally illustrative to simply add a type \mathbf{O} of *observables* which contains two constants $\{y, \mathbf{n}\}$ and no elimination form, following Harper [Har16]. We therefore extend \mathbb{T} with the following maps:

$$\diamond \xrightarrow{\mathbf{O}} \mathbb{T} \qquad \diamond \xrightarrow{y} \tau[\mathbf{O}] \qquad \diamond \xrightarrow{\mathbf{n}} \tau[\mathbf{O}]$$

Extending the gluing model. We will glue along the global sections functor $\mathcal{J} \xrightarrow{\Gamma} \mathbf{Set}$, which is clearly flat because its Yoneda extension $\mathsf{DF}_{\mathcal{J}} \xrightarrow{\text{Lan}_{\mathcal{G}} \Gamma} \mathbf{Set}$ can be seen to be an algebraic morphism of logoi. We must extend the constructions from Sections 3 and 4 to account for the type of observables.

To that end, we define a computability family $\mathcal{G}[\diamond \xrightarrow{\mathbf{O}} \mathbb{T}]$ lying over $(\diamond \xrightarrow{\mathbf{O}} \mathbb{T})_{\mathcal{J}}$. From the perspective of $\mathcal{E} = \mathbf{Set}$, it suffices to construct a family of (small) sets lying over $[\tau[\mathbf{O}]]$. We choose the family $\{0, 1\} \rightarrow [\tau[\mathbf{O}]]$ sending 0 to $y_{\mathcal{J}}$ and 1 to $\mathbf{n}_{\mathcal{J}}$. In other words, a computable element over $o : [\tau[\mathbf{O}]]$ is a witness that $o \in \{y_{\mathcal{J}}, \mathbf{n}_{\mathcal{J}}\}$.

We model constants $\mathcal{G}[\diamond \xrightarrow{y, \mathbf{n}} \tau[\mathbf{O}]]$ lying over the closed terms $(\diamond \xrightarrow{y, \mathbf{n}} \tau[\mathbf{O}])_{\mathcal{J}}$, choosing 0 and 1 respectively.

Theorem 1 (Canonicity). *Let $\mathfrak{K}1_{\mathcal{J}} \xrightarrow{o} \tau[\mathbf{O}]_{\mathcal{J}}$ be an observable in empty context. Then either $o = y_{\mathcal{J}}$ or $o = \mathbf{n}_{\mathcal{J}}$.*

Proof. Using the evaluation morphism and realigning by α , we have the following situation:

$$\begin{array}{ccc}
 1_{\mathcal{G}} \cong \mathsf{K}(\llbracket 1_{\mathcal{J}} \rrbracket) & \xrightarrow{\llbracket o \rrbracket} & \mathcal{G}[\tau[\mathbf{O}]] \\
 \downarrow & & \downarrow \\
 \mathfrak{K}1_{\mathcal{J}} & \xrightarrow{o} & \tau[\mathbf{O}]_{\mathcal{J}}
 \end{array}
 \qquad
 \begin{array}{c}
 \mathcal{G} \\
 \downarrow \mathfrak{g}^! \\
 \mathsf{DF}_{\mathcal{J}}
 \end{array}$$

Considering that $\mathcal{G}[\tau[\mathbf{O}]] \cong \mathcal{G}[\tau[\mathcal{G}[\mathbf{O}]]]$, by unfolding the definition of $\mathcal{G}[\mathbf{O}]$ we immediately obtain $o \in \{y_{\mathcal{J}}, \mathbf{n}_{\mathcal{J}}\}$. \square

APPENDIX A. APPENDIX

A.1. Type theory.

Lemma 11. *The assignments $\tau \mapsto \tau^{\Pi}$ and $\tau \mapsto \tau^{\Sigma}$ extend to endofunctors \bullet^{Π} and \bullet^{Σ} on $[\Delta^1, \mathbb{T}]_{\text{cart}}$.*

Proof. We consider the case of dependent products; the case of dependent sums is analogous. Fix $\mathcal{S} \xrightarrow{\alpha} \tau : [\Delta^1, \mathbb{T}]_{\text{cart}}$, which is the following cartesian square:

$$(A.1) \quad \begin{array}{ccc} \dot{\mathcal{S}} & \xrightarrow{\alpha_0} & \dot{\mathcal{T}} \\ \mathcal{S} \downarrow \lrcorner & & \downarrow \tau \\ \mathcal{S} & \xrightarrow{\alpha_1} & \mathcal{T} \end{array}$$

We will first exhibit a functorial action $\mathcal{S}^\Pi \xrightarrow{\alpha^\Pi} \tau^\Pi$, and then we will show that \mathcal{S}^Π is cartesian.

$$(A.2) \quad \begin{array}{ccc} \dot{\mathcal{S}}^\Pi & \xrightarrow{\alpha_0^\Pi} & \dot{\mathcal{T}}^\Pi \\ \mathcal{S}^\Pi \downarrow \lrcorner & & \downarrow \tau^\Pi \\ \mathcal{S}^\Pi & \xrightarrow{\alpha_1^\Pi} & \mathcal{T}^\Pi \end{array}$$

We may define α_1^Π as follows, using the canonical isomorphism $\chi_A : \tau[\alpha_1 A] \cong \zeta[A]$ (which we have because Diagram A.1 is cartesian):

$$\alpha_1^\Pi (G : \mathcal{S}^\Pi) = (\alpha_1(G_0), \alpha_1 \circ G_1 \circ \chi_{G_0})$$

We define the upstairs map α_0^Π fiberwise over each $G : \mathcal{S}^\Pi$ to produce a dependent map $\alpha_0^\Pi[G]$ of the following type:

$$\begin{aligned} \mathcal{S}^\Pi[G] &\Rightarrow \tau^\Pi[\alpha_1^\Pi G] \\ &\cong \mathcal{S}^\Pi[G] \Rightarrow (x : \tau[\alpha_1 G_0]) \Rightarrow \tau[\alpha_1(G_1(\chi_{G_0}(x)))] \\ &\cong \mathcal{S}^\Pi[G] \Rightarrow (x : \zeta[G_0]) \Rightarrow \tau[\alpha_1(G_1(x))] \\ &\cong \mathcal{S}^\Pi[G] \Rightarrow (x : \zeta[G_0]) \Rightarrow \zeta[G_1(x)] \\ &\cong \mathcal{S}^\Pi[G] \Rightarrow \mathcal{S}^\Pi[G] \end{aligned}$$

Therefore, each fiber $\alpha_0^\Pi[G]$ is the following isomorphism:

$$\alpha_0^\Pi[G](g : \mathcal{S}^\Pi[G]) = \lambda x : \tau[\alpha_1 G_0] \cdot \chi_{G_1(\chi_{G_0}(x))}^{-1}(g(\chi_{G_0}(x)))$$

Next, we will check that Diagram A.2 is cartesian.

$$(A.3) \quad \begin{array}{ccc} X & \xrightarrow{g} & \dot{\mathbf{T}}^\Pi \\ \downarrow \text{dotted} & \searrow \alpha_0^\Pi & \downarrow \tau^\Pi \\ \dot{\mathbf{S}}^\Pi & \xrightarrow{\quad} & \dot{\mathbf{T}}^\Pi \\ \downarrow \text{dotted} & \searrow \alpha_1^\Pi & \downarrow \\ \mathbf{S}^\Pi & \xrightarrow{\quad} & \mathbf{T}^\Pi \\ \downarrow G & & \downarrow \\ \mathbf{S}^\Pi & \xrightarrow{\quad} & \mathbf{T}^\Pi \end{array}$$

The universal map of Diagram A.3 is obtained from g and the inverse to the isomorphism $\alpha_0^\Pi[G]$. \square

A.2. Semantic gluing.

Lemma 4. *The family $\mathcal{G}[\dot{\mathbf{T}}] \xrightarrow{\mathcal{G}[\tau]} \mathcal{G}[\mathbf{T}] : \mathcal{G}$ is representable.*

Proof. We fix a dependent computability family in a compact context $\mathbf{K}(\Gamma) \xrightarrow{\mathfrak{A}} \mathcal{G}[\mathbf{T}]$, and must verify that the fiber product below is compact:

$$(A.5) \quad \begin{array}{ccc} \mathcal{G}[\dot{\mathbf{T}}] \times_{\mathcal{G}[\mathbf{T}]} \mathbf{K}(\Gamma) & \rightarrow & \mathcal{G}[\dot{\mathbf{T}}] \\ \downarrow \lrcorner & & \downarrow \mathcal{G}[\tau] \\ \mathbf{K}(\Gamma) & \xrightarrow{\mathfrak{A}} & \mathcal{G}[\mathbf{T}] \end{array}$$

A computability family is compact iff it lies over a representable discrete fibration. Therefore, it suffices to check that the image of the fiber product from Diagram A.5 under gl is representable in \mathbf{DF}_c . Noting that $\mathcal{G}[\dot{\mathbf{T}}] \xrightarrow{\mathcal{G}[\tau]} \mathcal{G}[\mathbf{T}]$ lies over $(\dot{\mathbf{T}} \xrightarrow{\tau} \mathbf{T})_c$ and $\text{gl}(\mathbf{K}(\Gamma)) = \mathfrak{A} \text{gl}_{\mathbf{K}}(\Gamma)$, this follows from the fact that $\mathbb{T} \xrightarrow{\mathfrak{M}_c} \mathbf{DF}_c$ must preserve representable maps. \square

Corollary 6. *We have the following canonical comparison maps, not generally invertible:*

$$\begin{array}{ccc} [\dot{\mathbf{T}}^\Pi] & \xrightarrow{(-)} & [\dot{\mathbf{T}}]^\Pi \\ \downarrow [\tau^\Pi] & & \downarrow [\tau]^\Pi \\ [\mathbf{T}^\Pi] & \xrightarrow{(-)} & [\mathbf{T}]^\Pi \end{array} \quad \begin{array}{ccc} [\dot{\mathbf{T}}^\Sigma] & \xrightarrow{(-)} & [\dot{\mathbf{T}}]^\Sigma \\ \downarrow [\tau^\Sigma] & & \downarrow [\tau]^\Sigma \\ [\mathbf{T}^\Sigma] & \xrightarrow{(-)} & [\mathbf{T}]^\Sigma \end{array}$$

Proof. We explicitly construct the map $[\mathbf{T}^\Pi] \rightarrow [\mathbf{T}]^\Pi$, since the other cases are analogous.

$$\begin{array}{c}
\frac{\frac{\overline{\dot{\mathbf{T}}^* \mathbf{T}} \rightarrow \dot{\mathbf{T}}^* \mathbf{T} \quad \text{identity}}{\dot{\mathbf{T}}^! \dot{\mathbf{T}}^* \mathbf{T} \rightarrow \mathbf{T}}}{\frac{\overline{[\dot{\mathbf{T}}^! \dot{\mathbf{T}}^* \mathbf{T}] \rightarrow [\mathbf{T}]} \quad [-]}{[\dot{\mathbf{T}}^! \dot{\mathbf{T}}^* \mathbf{T}] \rightarrow [\mathbf{T}]} \quad \text{Remark 4}}}{\frac{\overline{[\dot{\mathbf{T}}^* \mathbf{T}] \rightarrow [\dot{\mathbf{T}}]^* [\mathbf{T}]} \quad [\dot{\mathbf{T}}]^! \dashv [\dot{\mathbf{T}}]^*}{[\boldsymbol{\tau}]_* [\dot{\mathbf{T}}^* \mathbf{T}] \rightarrow [\boldsymbol{\tau}]_* [\dot{\mathbf{T}}]^* [\mathbf{T}]} \quad [\boldsymbol{\tau}]_* (-)}{\frac{\overline{[\boldsymbol{\tau}]_* [\dot{\mathbf{T}}^* \mathbf{T}] \rightarrow [\boldsymbol{\tau}]_* [\dot{\mathbf{T}}]^* [\mathbf{T}]} \quad \text{Lemma 5}}{[\mathbf{T}]_! (-)}{[\mathbf{T}]_! [\boldsymbol{\tau}]_* [\dot{\mathbf{T}}^* \mathbf{T}] \rightarrow [\mathbf{T}]_! [\boldsymbol{\tau}]_* [\dot{\mathbf{T}}]^* [\mathbf{T}]} \quad \text{Remark 4}}}{\frac{\overline{[\mathbf{T}]_! [\boldsymbol{\tau}]_* [\dot{\mathbf{T}}^* \mathbf{T}] \rightarrow [\mathbf{T}]_! [\boldsymbol{\tau}]_* [\dot{\mathbf{T}}]^* [\mathbf{T}]} \quad \text{by def.}}{[\mathbf{T}]^\Pi \rightarrow [\mathbf{T}]^\Pi} \quad \square}
\end{array}$$

A.3. Nerve and realization. Let \mathcal{E} be a lex and cocomplete category; \mathcal{E} is said to have *universal colimits* when colimits are stable under pullback; in particular, every locally cartesian closed category has universal colimits. A particularly useful reformulation of the universality of colimits is suggested by Anel and Joyal [AJ19]: a cocone is universal if it is cartesian over a universal cocone.

Lemma 6 (Characterization of universal colimits). *Let $\mathcal{D} \xrightarrow{C_\bullet} \mathcal{E}$ and $\mathcal{D} \xrightarrow{A_\bullet} \mathcal{E}$ be two diagrams such that $A_\bullet \xrightarrow{\alpha_\bullet} \{A_\infty\}$ is a universal cocone and $C_\bullet \xrightarrow{\gamma_\bullet} \{C_\infty\}$ is an arbitrary cocone. Let $\gamma_\bullet \xrightarrow{f_\bullet} \alpha_\bullet$ be a cartesian morphism of cocones, i.e. one in which the following diagram is cartesian for each $i : \mathcal{D}$:*

$$\begin{array}{ccc}
C_i & \xrightarrow{\gamma_i} & C_\infty \\
\downarrow f_i & \lrcorner & \downarrow f_\infty \\
A_i & \xrightarrow{\alpha_i} & A_\infty
\end{array}$$

Then the cocone γ_\bullet is universal.

Proof. Consider the following pullback situation:

$$\begin{array}{ccc}
f_\infty^* A_\infty & \longrightarrow & A_\infty \\
\downarrow \lrcorner & & \downarrow \text{id}_{A_\infty} \\
C_\infty & \xrightarrow{f_\infty} & A_\infty
\end{array}$$

The ordinary sense of the universality of the colimit A_∞ asserts that the pullback $f_\infty^* A_\infty$ is in fact $\text{colim}_i (f_\infty^* A_i)$; recalling our assumption $C_i = f_\infty^* A_i$, we therefore have the following

pullback square:

$$\begin{array}{ccc}
 \operatorname{colim} C_{\bullet} & \longrightarrow & A_{\infty} \\
 \downarrow \lrcorner & & \downarrow \mathbf{id}_{A_{\infty}} \\
 C_{\infty} & \xrightarrow{f_{\infty}} & A_{\infty}
 \end{array}$$

But any pullback of an isomorphism is an isomorphism, so C_{∞} is in fact the desired colimit. \square

Lemma 1 (Density). *Every computability family $X : \mathcal{G}$ is canonically a colimit of computability families which lie in the image of K , i.e. compact computability families.*

We thank Mathieu Anel for suggesting this argument.

Proof. Let $X : \mathcal{G}$ be a computability family. We need to check that X is the colimit of the following diagram $K \downarrow \{X\} \rightarrow \mathcal{G}$:

$$(A.7) \quad K \downarrow \{X\} \xrightarrow{\pi_K} \mathcal{G}_K \xrightarrow{K} \mathcal{G}$$

First, we note by the dual Yoneda lemma that $\operatorname{gl}(X) : \mathbf{DF}_{\mathcal{C}}$ is canonically the colimit of Diagram A.8 below:

$$(A.8) \quad \mathcal{K} \downarrow \{\operatorname{gl}(X)\} \xrightarrow{\pi_{\mathcal{K}}} \mathcal{C} \xrightarrow{\mathcal{K}} \mathbf{DF}_{\mathcal{C}}$$

Each leg $\mathcal{K}C_i \xrightarrow{\alpha_i} \operatorname{gl}(X)$ of the colimiting cocone for Diagram A.8 exhibits a cartesian lift at X in the gluing fibration:

$$\begin{array}{ccc}
 \alpha_i^* X & \xrightarrow{\alpha_i^\dagger X} & X \\
 \downarrow & & \downarrow \\
 \mathcal{K}C_i & \xrightarrow{\alpha_i} & \operatorname{gl}(X)
 \end{array}
 \quad
 \begin{array}{c}
 \mathcal{G} \\
 \downarrow \\
 \operatorname{gl} \\
 \downarrow \\
 \mathbf{DF}_{\mathcal{C}}
 \end{array}$$

The resulting cocone $\{\alpha_i^* X \xrightarrow{\alpha_i^\dagger X} X\}$ in \mathcal{G} can be seen to be colimiting from the universality of colimits [AJ19] in the topos \mathcal{E} , using the cocontinuity of F : the image of the cocone $\{\alpha_i^\dagger X\}$ in \mathcal{E} is cartesian over the colimiting cocone $\{F(\alpha_i)\}$ in the configuration required by Lemma 6.

In fact, we needed to show that X was the colimit of Diagram A.7, whose vertices are arbitrary compact computability families, not just those which are exhibited as cartesian lifts. However, each $K(\Gamma_i) \xrightarrow{f_i} X$ factors vertically through some $\alpha_i^* X \xrightarrow{\alpha_i^\dagger X} X$ over

$\mathfrak{J}g|_K(\Gamma_i) \xrightarrow{\alpha_i} gl(X)$ in a unique way by the universal property of the cartesian lift:

$$\begin{array}{ccc}
 K(\Gamma_i) & \xrightarrow{f_i} & X \\
 \downarrow \beta_i & \searrow & \downarrow \\
 \alpha_i^* X & \xrightarrow{\alpha_i^\dagger} & X \\
 \downarrow & & \downarrow \\
 \mathfrak{J}g|_K(\Gamma_i) & \xrightarrow{\alpha_i} & gl(X)
 \end{array}
 \quad
 \begin{array}{c}
 \mathcal{G} \\
 \downarrow gl \\
 \mathbf{DF}_e
 \end{array}$$

Finally, we observe that the cocone $\{K(\Gamma_i) \xrightarrow{\alpha_i^\dagger X \circ \beta_i} X\}$ is still colimiting. Given a cocone $\{K(\Gamma_i) \xrightarrow{Y_i} Y\}$ we must exhibit a universal map $Y \rightarrow X$; but this cocone contains within it a cocone $\{\alpha_i^* X \rightarrow Y\}$ because the cartesian lifts $\alpha_i^* X$ are all compact. Therefore, our universal map is already determined by the restriction of the cocone to cartesian lifts. \square

Corollary 9. *The nerve $\mathcal{G} \xrightarrow{N_K} \mathbf{DF}_{\mathcal{G}_K}$ is fully faithful.*

Proof. This is equivalent to the conclusion of Lemma 1. \square

Remark 10. The density of the subcategory of compact computability families may be re-stated in terms of the nerve functor, using the tensor calculus of functors [Mac98; nLa19b]. For each computability family $X : \mathcal{G}$, we have:

$$X \cong N_K(X) \otimes_{\mathcal{G}_K} K \quad \heartsuit$$

Construction 11 (Realization). The nerve has a left adjoint $\mathbf{DF}_{\mathcal{G}_K} \xrightarrow{|\cdot|_K} \mathcal{G}$, the *realization* of a discrete fibration on compact computability families. The realization functor is formed by Kan extension:

$$|P|_K \cong (\text{Lan}_{\mathfrak{J}\mathcal{G}_K} K)P \cong P \otimes_{\mathcal{G}_K} K \quad \heartsuit$$

Remark 12. Via Corollary 9, the density of $\mathcal{G}_K \xleftarrow{K} \mathcal{G}$ is also equivalent to the counit $|\cdot|_K \circ N_K \xrightarrow{\epsilon} \text{id}_{\mathcal{G}}$ being a natural isomorphism. Indeed, this is easy to see by a computation in the tensor calculus:

$$\begin{array}{ll}
 |N_K(X)|_K \cong N_K(X) \otimes_{\mathcal{G}_K} K & \text{Construction 11} \\
 \cong X & \text{Remark 10} \quad \heartsuit
 \end{array}$$

The density of the inclusion K allows us to characterize the equivalent subcategories determined by the adjunction.

Lemma 13. *The nerve–realization adjunction $|\cdot|_K \dashv N_K$ restricts to an equivalence between the subcategories of compact computability families and representable presheaves.*

Proof. Let Γ be a compact computability family; we first check that the nerve of Γ is a representable presheaf.

$$\begin{array}{ll}
 N_K(K(\Gamma)) \cong \mathcal{G}[K(\bullet), K(\Gamma)] & \text{by def.} \\
 \cong \mathcal{G}_K[\bullet, \Gamma] & K \text{ fully faithful} \\
 \cong \mathfrak{J}\Gamma & \text{Yoneda lemma}
 \end{array}
 \tag{A.14}$$

Next, we check that the realization of the representable presheaf $\mathfrak{A}\Gamma$ is the corresponding compact computability family.

$$\begin{aligned} |\mathfrak{A}\Gamma|_{\mathbb{K}} &\cong |\mathbb{N}_{\mathbb{K}}(\mathbb{K}(\Gamma))|_{\mathbb{K}} && \text{Equation A.14} \\ &\cong \mathbb{K}(\Gamma) && \text{Remark 12} \quad \square \end{aligned}$$

Remark 15. In the same way that a presheaf category $\mathbf{DF}_{\mathcal{C}}$ is densely generated by \mathcal{C} under the Yoneda embedding, the slices $(\mathbf{DF}_{\mathcal{C}})_{/X}$ are densely generated by the total category $X : \mathbf{Cat}$ of the discrete fibration $X : \mathbf{DF}_{\mathcal{C}}$. As a subcategory of $(\mathbf{DF}_{\mathcal{C}})_{/X}$, this consists in the arrows $\mathfrak{A}c \rightarrow X : \mathbf{DF}_{\mathcal{C}}$. \heartsuit

Lemma 2. *The nerve is locally cartesian closed.*

Proof. As a right adjoint, the nerve is clearly left exact; we need to check that it preserves dependent products. Fixing $Y \xrightarrow{f} X$ and $Z \xrightarrow{g} Y$ in \mathcal{G} , we must check that $\mathbb{N}_{\mathbb{K}}(f)_* \mathbb{N}_{\mathbb{K}}(g)$ is $\mathbb{N}_{\mathbb{K}}(f_*g)$. Following Remark 15, it suffices to probe these objects at some $\mathfrak{A}\Gamma \xrightarrow{x} \mathbb{N}_{\mathbb{K}}(X)$.

Of course, we have a transpose $|\mathfrak{A}\Gamma|_{\mathbb{K}} \xrightarrow{\tilde{x}} X : \mathcal{G}$, and thence by functoriality we have $\mathbb{N}_{\mathbb{K}}(|\mathfrak{A}\Gamma|_{\mathbb{K}}) \xrightarrow{\mathbb{N}_{\mathbb{K}}(\tilde{x})} \mathbb{N}_{\mathbb{K}}(X) : \mathbf{DF}_{\mathcal{G}_{\mathbb{K}}}$. Because $|-|_{\mathbb{K}} \dashv \mathbb{N}_{\mathbb{K}}$ restricts to an equivalence on the dense generators, we can see that $\mathbb{N}_{\mathbb{K}}(\tilde{x}) \cong x : (\mathbf{DF}_{\mathcal{G}_{\mathbb{K}}})_{/\mathbb{N}_{\mathbb{K}}(X)}$. Therefore, the following calculation is justified:

$$\begin{aligned} &(\mathbf{DF}_{\mathcal{G}_{\mathbb{K}}})_{/\mathbb{N}_{\mathbb{K}}(X)}[x, \mathbb{N}_{\mathbb{K}}(f)_* \mathbb{N}_{\mathbb{K}}(g)] \\ &\cong (\mathbf{DF}_{\mathcal{G}_{\mathbb{K}}})_{/\mathbb{N}_{\mathbb{K}}(X)}[\mathbb{N}_{\mathbb{K}}(\tilde{x}), \mathbb{N}_{\mathbb{K}}(f)_* \mathbb{N}_{\mathbb{K}}(g)] \\ &\cong (\mathbf{DF}_{\mathcal{G}_{\mathbb{K}}})_{/\mathbb{N}_{\mathbb{K}}(Y)}[\mathbb{N}_{\mathbb{K}}(f)_* \mathbb{N}_{\mathbb{K}}(\tilde{x}), \mathbb{N}_{\mathbb{K}}(g)] \\ &\cong (\mathbf{DF}_{\mathcal{G}_{\mathbb{K}}})_{/\mathbb{N}_{\mathbb{K}}(Y)}[\mathbb{N}_{\mathbb{K}}(f^* \tilde{x}), \mathbb{N}_{\mathbb{K}}(g)] \\ &\cong \mathcal{G}_{/Y}[f^* \tilde{x}, g] \\ &\cong \mathcal{G}_{/X}[\tilde{x}, f_*g] \\ &\cong (\mathbf{DF}_{\mathcal{G}_{\mathbb{K}}})_{/\mathbb{N}_{\mathbb{K}}(X)}[\mathbb{N}_{\mathbb{K}}(\tilde{x}), \mathbb{N}_{\mathbb{K}}(f_*g)] \\ &\cong (\mathbf{DF}_{\mathcal{G}_{\mathbb{K}}})_{/\mathbb{N}_{\mathbb{K}}(X)}[x, \mathbb{N}_{\mathbb{K}}(f_*g)] \quad \square \end{aligned}$$

Lemma 3. *The nerve functor $\mathcal{G} \xrightarrow{\mathbb{N}_{\mathbb{K}}} \mathbf{DF}_{\mathcal{G}_{\mathbb{K}}}$ is a representable map functor.*

Proof. Because the nerve functor is locally cartesian closed (Lemma 2), it is in particular left exact and preserves pushforwards along representable maps; it remains to see that the nerve preserves representable maps. Fix a representable (i.e. compact) map $Y \xrightarrow{f} X \in \mathcal{R}_{\mathcal{G}}$; we must check that the nerve of this map $\mathbb{N}_{\mathbb{K}}(Y) \xrightarrow{\mathbb{N}_{\mathbb{K}}(f)} \mathbb{N}_{\mathbb{K}}(X)$ is in representable in $\mathbf{DF}_{\mathcal{C}}$. This is most easily checked in the ‘‘Grothendieck style’’, requiring that the following fiber product lies in the image of the Yoneda embedding:

$$\begin{array}{ccc} \mathbb{N}_{\mathbb{K}}(Y) \times_{\mathbb{N}_{\mathbb{K}}(X)} \mathfrak{A}\Gamma & \triangleright & \mathbb{N}_{\mathbb{K}}(Y) \\ \downarrow \lrcorner & & \downarrow \mathbb{N}_{\mathbb{K}}(f) \\ \mathfrak{A}\Gamma & \xrightarrow{x} & \mathbb{N}_{\mathbb{K}}(X) \end{array}$$

Because the nerve–realization adjunction restricts to an equivalence between generating objects, we may rewrite the square as follows:

$$\begin{array}{ccc}
 N_{\mathbb{K}}(Y) \times_{N_{\mathbb{K}}(X)} N_{\mathbb{K}}(|\mathfrak{K}\Gamma|_{\mathbb{K}}) & \xrightarrow{N_{\mathbb{K}}(f)} & N_{\mathbb{K}}(Y) \\
 \downarrow & \lrcorner & \downarrow \\
 N_{\mathbb{K}}(|\mathfrak{K}\Gamma|_{\mathbb{K}}) & \xrightarrow{N_{\mathbb{K}}(\tilde{x})} & N_{\mathbb{K}}(X)
 \end{array}$$

The nerve is left exact and fully faithful and moreover $|-|_{\mathbb{K}} \circ \mathfrak{K} \cong \mathbb{K}$ (Lemma 13), so it suffices to check that the left-hand map below is compact:

$$\begin{array}{ccc}
 Y \times_X K(\Gamma) & \longrightarrow & Y \\
 \downarrow & \lrcorner & \downarrow f \\
 K(\Gamma) & \xrightarrow{\tilde{x}} & X
 \end{array}$$

But this follows from the compactness of f . \square

A.4. The gluing model.

Action of right adjoints to representable maps. In order to check the Beck-Chevalley condition, it will be useful to have a concrete characterization of the objects of the total categories $J_{\mathcal{G}_{\mathbb{K}}}$, and the actions of the right adjoints $\mathbf{q}f_{\mathcal{G}_{\mathbb{K}}}$ on these objects, fixing a representable map $J \xrightarrow{f} I \in \mathcal{R}_{\mathbb{T}}$.

- (1) Recalling that $J_{\mathcal{G}_{\mathbb{K}}}$ is just $N_{\mathbb{K}}(\mathcal{G}[J])$, we see that an object of the total category is a map $\mathfrak{K}\Gamma \rightarrow N_{\mathbb{K}}(\mathcal{G}[J]) : \mathbf{DF}_{\mathcal{G}_{\mathbb{K}}}$, which is the same as a map $K(\Gamma) \rightarrow \mathcal{G}[J] : \mathcal{G}$.
- (2) The functor $I_{\mathcal{G}_{\mathbb{K}}} \xrightarrow{\mathbf{q}f_{\mathcal{G}_{\mathbb{K}}}} J_{\mathcal{G}_{\mathbb{K}}} : \mathbf{Cat}$ can be seen to have the following action from the perspective of \mathcal{G} , using the fact that $N_{\mathbb{K}}$ is left exact:

$$\begin{array}{ccccc}
 K(\Gamma) & & \mathcal{G}[J] \times_{\mathcal{G}[I]} K(\Gamma) & \xrightarrow{\mathbf{q}f_{\mathcal{G}_{\mathbb{K}}}(i)} & \mathcal{G}[J] \\
 \downarrow i & \nearrow \mathbf{q}f_{\mathcal{G}_{\mathbb{K}}} & \downarrow & \lrcorner & \downarrow \mathcal{G}[f] \\
 \mathcal{G}[I] & & K(\Gamma) & \xrightarrow{i} & \mathcal{G}[I]
 \end{array}$$

Lemma 5 (Morphism of models). *The natural transformation $\mathcal{M}_{\mathcal{G}_{\mathbb{K}}} \xrightarrow{\mathcal{M}_{\mathfrak{gl}_{\mathbb{K}}}} \mathcal{M}_{\mathcal{C}} : [\mathbb{T}, \mathbf{DF}]$ exhibits a morphism of models in $\mathbf{Mod}_{\mathbb{T}}$, tracked by $\mathcal{G}_{\mathbb{K}} \xrightarrow{\mathfrak{gl}_{\mathbb{K}}} \mathcal{C}$.*

Proof. We must check the Beck-Chevalley condition for each representable map $J \xrightarrow{f} I \in \mathcal{R}_{\mathbb{T}}$. In particular, the following square must commute up to canonical isomorphism.

$$\begin{array}{ccc} I_{\mathcal{G}_K} & \xrightarrow{I_{\text{gl}_K}} & I_e \\ \mathbf{q}_{f_{\mathcal{G}_K}} \downarrow & \cong & \downarrow \mathbf{q}_{f_e} \\ J_{\mathcal{G}_K} & \xrightarrow{J_{\text{gl}_K}} & J_e \end{array}$$

We fix an object $K(\Gamma) \xrightarrow{i} \mathcal{G}[I] \in \text{ob } I_{\mathcal{G}_K}$, which is taken by I_{gl_K} to $\mathfrak{K}\text{gl}_K(\Gamma) \xrightarrow{\chi_I \circ \text{gl}(i)} I_e$; this is taken by \mathbf{q}_{f_e} to the dotted map in the following pullback square in DF_e :

$$(A.16) \quad \begin{array}{ccc} \mathfrak{K}(\dots) & \xrightarrow{\mathbf{q}_{f_e}(I_{\text{gl}_K}(i))} & J_e \\ \downarrow \lrcorner & & \downarrow f_e \\ \mathfrak{K}\text{gl}_K(\Gamma) & \xrightarrow{I_{\text{gl}_K}(i) = \chi_I \circ \text{gl}(i)} & I_e \end{array}$$

Alternatively, i is taken by $\mathbf{q}_{f_{\mathcal{G}_K}}$ to the dotted map in the following pullback square in \mathcal{G} :

$$(A.17) \quad \begin{array}{ccc} K(\dots) & \xrightarrow{\mathbf{q}_{f_{\mathcal{G}_K}}(i)} & \mathcal{G}[J] \\ \downarrow \lrcorner & & \downarrow \mathcal{G}[f] \\ K(\Gamma) & \xrightarrow{i} & \mathcal{G}[I] \end{array}$$

The map $\mathbf{q}_{f_{\mathcal{G}_K}}(i)$ is taken by J_{gl_K} to the composite upstairs map of Diagram A.18, noting that the left-hand pullback square below is the image of Diagram A.17 under gl and the right-hand square comes from the naturality of χ_\bullet .

$$(A.18) \quad \begin{array}{ccccc} \mathfrak{K}\text{gl}_K(\dots) & \rightarrow & \text{gl}(\mathcal{G}[J]) & \xrightarrow{\chi_J} & J_e \\ \downarrow \lrcorner & & \downarrow \text{gl}(\mathcal{G}[f]) & & \downarrow f_e \\ \mathfrak{K}\text{gl}_K(\Gamma) & \xrightarrow{\text{gl}(i)} & \text{gl}(\mathcal{G}[I]) & \xrightarrow{\chi_I} & I_e \end{array}$$

We need to check that the composite upstairs map of Diagram A.18 is canonically isomorphic to the upstairs map of Diagram A.16. But the upstairs and downstairs maps of the right-hand square are isomorphisms, so the outer square is a pullback. \square

REFERENCES

- [ACD08] Andreas Abel, Thierry Coquand, and Peter Dybjer. “Verifying a Semantic $\beta\eta$ -Conversion Test for Martin-Löf Type Theory”. In: *Mathematics of Program Construction*. Ed. by Philippe Audebaud and Christine Paulin-Mohring. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 29–56. ISBN: 978-3-540-70594-9 (cit. on p. 8).
- [AGV72] Michael Artin, Alexander Grothendieck, and Jean-Louis Verdier. *Théorie des topos et cohomologie étale des schémas*. Séminaire de Géométrie Algébrique du Bois-Marie 1963–1964 (SGA 4), Dirigé par M. Artin, A. Grothendieck, et J.-L. Verdier. Avec la collaboration de N. Bourbaki, P. Deligne et B. Saint-Donat, Lecture Notes in Mathematics, Vol. 269, 270, 305. Berlin: Springer-Verlag, 1972 (cit. on pp. 1, 11, 12).
- [AHH17] Carlo Angiuli, Kuen-Bang Hou (Favonia), and Robert Harper. *Computational Higher Type Theory III: Univalent Universes and Exact Equality*. 2017. arXiv: 1712.01800 (cit. on p. 2).
- [AHS95] Thorsten Altenkirch, Martin Hofmann, and Thomas Streicher. “Categorical reconstruction of a reduction free normalization proof”. In: *Category Theory and Computer Science*. Ed. by David Pitt, David E. Rydeheard, and Peter Johnstone. Berlin, Heidelberg: Springer Berlin Heidelberg, 1995, pp. 182–199. ISBN: 978-3-540-44661-3 (cit. on p. 4).
- [AJ19] Mathieu Anel and André Joyal. *Topologie*. Preprint. Mar. 2019. URL: <http://mathieu.anel.free.fr/mat/doc/Anel-Joyal-Topologie.pdf> (cit. on pp. 11, 24, 25).
- [AK16] Thorsten Altenkirch and Ambrus Kaposi. “Type Theory in Type Theory Using Quotient Inductive Types”. In: *Proceedings of the 43rd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*. POPL ’16. St. Petersburg, FL, USA: ACM, 2016, pp. 18–29. ISBN: 978-1-4503-3549-2. DOI: 10.1145/2837614.2837638 (cit. on p. 4).
- [All87] Stuart Frazier Allen. “A Non-Type-Theoretic Definition of Martin-Löf’s Types”. In: *Proceedings of the Symposium on Logic in Computer Science (LICS ’87)*. Ithaca, NY: IEEE, June 1987, pp. 215–221 (cit. on p. 2).
- [Ang+18] Carlo Angiuli, Evan Cavallo, Kuen-Bang Hou (Favonia), Robert Harper, Anders Mörtberg, and Jonathan Sterling. *redtt: implementing Cartesian cubical type theory*. Dagstuhl Seminar 18341: Formalization of Mathematics in Type Theory. 2018. URL: <http://www.jonmsterling.com/pdfs/dagstuhl.pdf> (cit. on p. 6).
- [Ang19] Carlo Angiuli. “Computational Semantics of Cartesian Cubical Type Theory”. PhD thesis. Carnegie Mellon University, 2019 (cit. on p. 2).
- [AR14] Abhishek Anand and Vincent Rahli. “Towards a Formally Verified Proof Assistant”. In: *Interactive Theorem Proving: 5th International Conference, ITP 2014, Held as Part of the Vienna Summer of Logic, VSL 2014, Vienna, Austria, July 14-17, 2014. Proceedings*. Ed. by Gerwin Klein and Ruben Gamboa. Cham: Springer International Publishing, 2014, pp. 27–44. ISBN: 978-3-319-08970-6 (cit. on p. 2).
- [Awo+14] Steve Awodey, Carsten Butz, Alex Simpson, and Thomas Streicher. “Relating first-order set theories, toposes and categories of classes”. In: *Annals of Pure and Applied Logic* 165.2 (2014), pp. 428–502. ISSN: 0168-0072. DOI: <https://doi.org/10.1016/j.apal.2013.06.004> (cit. on p. 7).

- [Awo18] Steve Awodey. “Natural models of homotopy type theory”. In: *Mathematical Structures in Computer Science* 28.2 (2018), pp. 241–286. DOI: [10.1017/S0960129516000268](https://doi.org/10.1017/S0960129516000268) (cit. on pp. 7–9).
- [BGM17] P. Bahr, H. B. Grathwohl, and R. E. Møgelberg. “The clocks are ticking: No more delays!” In: *Proceedings of the 32nd Annual ACM/IEEE Symposium on Logic in Computer Science*. Reykjavik, Iceland: IEEE Press, June 2017, pp. 1–12. ISBN: 978-1-5090-3018-7 (cit. on p. 2).
- [Bor10] Francis Borceux. *Handbook of Categorical Algebra 3 – Categories of Sheaves*. Cambridge University Press, 2010 (cit. on p. 11).
- [Car86] John Cartmell. “Generalised Algebraic Theories and Contextual Categories”. In: *Annals of Pure and Applied Logic* 32 (1986), pp. 209–243. ISSN: 0168-0072 (cit. on pp. 4, 8).
- [CD14] Pierre Clairambault and Peter Dybjer. “The biequivalence of locally cartesian closed categories and Martin-Löf type theories”. In: *Mathematical Structures in Computer Science* 24.6 (2014). DOI: [10.1017/S0960129513000881](https://doi.org/10.1017/S0960129513000881) (cit. on pp. 5, 8).
- [CGH14] Pierre-Louis Curien, Richard Garner, and Martin Hofmann. “Revisiting the categorical interpretation of dependent type theory”. In: *Theoretical Computer Science* 546 (2014). Models of Interaction: Essays in Honour of Glynn Winskel, pp. 99–119. ISSN: 0304-3975. DOI: [10.1016/j.tcs.2014.03.003](https://doi.org/10.1016/j.tcs.2014.03.003) (cit. on p. 5).
- [CHS19] Thierry Coquand, Simon Huber, and Christian Sattler. “Homotopy canonicity for cubical type theory”. In: *4th International Conference on Formal Structures for Computation and Deduction (FSCD 2019)*. Ed. by Herman Geuvers. Vol. 131. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2019. ISBN: 978-3-95977-107-8 (cit. on pp. 4, 5).
- [CJ95] Aurelio Carboni and Peter Johnstone. “Connected limits, familial representability and Artin glueing”. In: *Mathematical Structures in Computer Science* 5.4 (1995), pp. 441–459. DOI: [10.1017/S096012950001183](https://doi.org/10.1017/S096012950001183) (cit. on p. 11).
- [Clo+18] Ranald Clouston, Bassel Manna, Rasmus Ejlers Møgelberg, Andrew M. Pitts, and Bas Spitters. *Modal Dependent Type Theory and Dependent Right Adjoints*. 2018. arXiv: [1804.05236](https://arxiv.org/abs/1804.05236) (cit. on p. 8).
- [Coq16] The Coq Development Team. *The Coq Proof Assistant Reference Manual*. 2016 (cit. on p. 6).
- [Coq19] Thierry Coquand. “Canonicity and normalization for dependent type theory”. In: *Theoretical Computer Science* 777 (2019). In memory of Maurice Nivat, a founding father of Theoretical Computer Science - Part I, pp. 184–191. ISSN: 0304-3975. DOI: [10.1016/j.tcs.2019.01.015](https://doi.org/10.1016/j.tcs.2019.01.015). arXiv: [1810.09367](https://arxiv.org/abs/1810.09367) (cit. on pp. 2, 4, 5).
- [Dyb96] Peter Dybjer. “Internal type theory”. In: *Types for Proofs and Programs: International Workshop, TYPES '95 Torino, Italy, June 5–8, 1995 Selected Papers*. Ed. by Stefano Berardi and Mario Coppo. Berlin, Heidelberg: Springer Berlin Heidelberg, 1996, pp. 120–134. ISBN: 978-3-540-70722-6 (cit. on p. 7).
- [Fio02] Marcelo Fiore. “Semantic Analysis of Normalisation by Evaluation for Typed Lambda Calculus”. In: *Proceedings of the 4th ACM SIGPLAN International Conference on Principles and Practice of Declarative Programming*. PPDP '02.

- Pittsburgh, PA, USA: ACM, 2002, pp. 26–37. ISBN: 1-58113-528-9. DOI: [10.1145/571157.571161](https://doi.org/10.1145/571157.571161) (cit. on p. 4).
- [Fre78] Peter Freyd. “On proving that $\mathbf{1}$ is an indecomposable projective in various free categories”. Unpublished manuscript. 1978 (cit. on p. 1).
- [FS99] Marcelo Fiore and Alex Simpson. “Lambda Definability with Sums via Grothendieck Logical Relations”. In: *Typed Lambda Calculi and Applications*. Ed. by Jean-Yves Girard. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 147–161. ISBN: 978-3-540-48959-7 (cit. on p. 2).
- [Gir71] Jean-Yves Girard. “Une extension de l’interprétation de Gödel à l’analyse, et son application à l’élimination de coupures dans l’analyse et la théorie des types”. In: *Proceedings of the Second Scandinavian Logic Symposium*. 1971 (cit. on p. 1).
- [Gir72] Jean-Yves Girard. “Interprétation fonctionnelle et élimination des coupures de l’arithmétique d’ordre supérieur”. PhD thesis. Université Paris VII, 1972 (cit. on p. 1).
- [Gro60] Alexander Grothendieck. “Éléments de géométrie algébrique : I. Le langage des schémas”. fr. In: *Publications Mathématiques de l’IHÉS* 4 (1960), pp. 5–228. URL: http://www.numdam.org/item/PMIHES_1960__4__5_0 (cit. on p. 3).
- [GSB19] Daniel Gratzer, Jonathan Sterling, and Lars Birkedal. “Implementing a Modal Dependent Type Theory”. In: *Proceedings of the ACM on Programming Languages* 3.ICFP (July 2019), 107:1–107:29. ISSN: 2475-1421. DOI: [10.1145/3341711](https://doi.org/10.1145/3341711) (cit. on p. 2).
- [Han+13] Peter Hancock, Conor McBride, Neil Ghani, Lorenzo Malatesta, and Thorsten Altenkirch. “Small Induction Recursion”. In: *Typed Lambda Calculi and Applications: 11th International Conference, TLCA 2013, Eindhoven, The Netherlands, June 26-28, 2013. Proceedings*. Ed. by Masahito Hasegawa. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 156–172. ISBN: 978-3-642-38946-7 (cit. on p. 17).
- [Har16] Robert Harper. *Practical Foundations for Programming Languages*. Second. New York, NY, USA: Cambridge University Press, 2016 (cit. on p. 21).
- [Har92] Robert Harper. “Constructing Type Systems over an Operational Semantics”. In: *Journal of Symbolic Computation* 14.1 (July 1992), pp. 71–84. ISSN: 0747-7171 (cit. on p. 2).
- [HHP93] Robert Harper, Furio Honsell, and Gordon Plotkin. “A Framework for Defining Logics”. In: *J. ACM* 40.1 (Jan. 1993), pp. 143–184. ISSN: 0004-5411. DOI: [10.1145/138027.138060](https://doi.org/10.1145/138027.138060) (cit. on p. 6).
- [Hof95] Martin Hofmann. “On the interpretation of type theory in locally cartesian closed categories”. In: *Computer Science Logic*. Ed. by Leszek Pacholski and Jerzy Tiuryn. Berlin, Heidelberg: Springer Berlin Heidelberg, 1995, pp. 427–441. ISBN: 978-3-540-49404-1 (cit. on p. 5).
- [Jac99] Bart Jacobs. *Categorical Logic and Type Theory*. Studies in Logic and the Foundations of Mathematics 141. Amsterdam: North Holland, 1999 (cit. on p. 7).
- [Joy17] Andre Joyal. *Notes on Clans and Tribes*. 2017. arXiv: [1710.10238](https://arxiv.org/abs/1710.10238) (cit. on p. 5).
- [JT93] Achim Jung and Jerzy Tiuryn. “A new characterization of lambda definability”. In: *Typed Lambda Calculi and Applications*. Ed. by Marc Bezem and Jan Friso Groote. Berlin, Heidelberg: Springer Berlin Heidelberg, 1993, pp. 245–257. ISBN: 978-3-540-47586-6 (cit. on p. 2).

- [KHS19] Ambrus Kaposi, Simon Huber, and Christian Sattler. “Gluing for type theory”. In: *4th International Conference on Formal Structures for Computation and Deduction (FSCD 2019)*. Ed. by Herman Geuvers. Vol. 131. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2019. ISBN: 978-3-95977-107-8 (cit. on pp. 4, 5, 8, 11).
- [KKA19] Ambrus Kaposi, András Kovács, and Thorsten Altenkirch. “Constructing Quotient Inductive-inductive Types”. In: *Proc. ACM Program. Lang.* 3:POPL (Jan. 2019), 2:1–2:24. ISSN: 2475-1421. DOI: [10.1145/3290315](https://doi.org/10.1145/3290315) (cit. on p. 4).
- [Koc06] Anders Kock. *Synthetic Differential Geometry*. 2nd ed. London Mathematical Society Lecture Note Series. Cambridge University Press, 2006. DOI: [10.1017/CB09780511550812](https://doi.org/10.1017/CB09780511550812) (cit. on p. 3).
- [KPB15] Neelakantan R. Krishnaswami, Pierre Pradic, and Nick Benton. “Integrating Linear and Dependent Types”. In: *POPL ’15: Proceedings of the 42nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*. Mumbai, India: ACM, 2015, pp. 17–30. ISBN: 978-1-4503-3300-9 (cit. on p. 2).
- [KS19] Chris Kapulkin and Christian Sattler. *Homotopy canonicity of homotopy type theory*. Slides from a talk given at the International Conference on Homotopy Type Theory (HoTT 2019). Aug. 2019. URL: <https://hott.github.io/HoTT-2019/conf-slides/Sattler.pdf> (cit. on p. 6).
- [Law63] F. William Lawvere. “Functorial Semantics of Algebraic Theories”. PhD thesis. Columbia University, 1963 (cit. on p. 6).
- [Mac98] Saunders Mac Lane. *Categories for the Working Mathematician*. 2nd. Springer-Verlag New York, 1998 (cit. on p. 26).
- [Mar75] Per Martin-Löf. “An Intuitionistic Theory of Types: Predicative Part”. In: *Logic Colloquium ’73*. Ed. by H. E. Rose and J. C. Shepherdson. Vol. 80. Studies in Logic and the Foundations of Mathematics. Elsevier, 1975, pp. 73–118. DOI: [10.1016/S0049-237X\(08\)71945-1](https://doi.org/10.1016/S0049-237X(08)71945-1) (cit. on p. 1).
- [Mou+15] Leonardo de Moura, Soonho Kong, Jeremy Avigad, Floris van Doorn, and Jakob von Raumer. “The Lean Theorem Prover (System Description)”. In: *Automated Deduction - CADE-25: 25th International Conference on Automated Deduction, Berlin, Germany, August 1-7, 2015, Proceedings*. Ed. by Amy P. Felty and Aart Middeldorp. Cham: Springer International Publishing, 2015, pp. 378–388. ISBN: 978-3-319-21401-6 (cit. on p. 6).
- [MP00] Ieke Moerdijk and Erik Palmgren. “Wellfounded trees in categories”. In: *Annals of Pure and Applied Logic* 104.1 (2000), pp. 189–218. ISSN: 0168-0072 (cit. on p. 17).
- [MS93] John C. Mitchell and Andre Scedrov. “Notes on scoping and relators”. In: *Computer Science Logic*. Ed. by E. Börger, G. Jäger, H. Kleine Büning, S. Martini, and M. M. Richter. Berlin, Heidelberg: Springer Berlin Heidelberg, 1993, pp. 352–378. ISBN: 978-3-540-47890-4 (cit. on p. 2).
- [New18] Clive Newstead. “Algebraic Models of Dependent Type Theory”. PhD thesis. Carnegie Mellon University, 2018 (cit. on pp. 8, 9).
- [nLa16] nLab. *smooth locus*. 2016. URL: <https://ncatlab.org/nlab/show/smooth+locus> (cit. on p. 3).
- [nLa17] nLab. *free topos*. 2017. URL: <https://ncatlab.org/nlab/show/free+topos> (cit. on p. 2).

- [nLa18a] nLab. *Freyd cover*. 2018. URL: <https://ncatlab.org/nlab/show/Freyd+cover> (cit. on p. 2).
- [nLa18b] nLab. *functorial geometry*. 2018. URL: <https://ncatlab.org/nlab/show/functorial+geometry> (cit. on p. 3).
- [nLa19a] nLab. *flat functor*. 2019. URL: <https://ncatlab.org/nlab/show/flat+functor> (cit. on p. 11).
- [nLa19b] nLab. *tensor product of functors*. 2019. URL: <https://ncatlab.org/nlab/show/tensor+product+of+functors> (cit. on p. 26).
- [Nor07] Ulf Norell. “Towards a practical programming language based on dependent type theory”. PhD thesis. SE-412 96 Göteborg, Sweden: Department of Computer Science and Engineering, Chalmers University of Technology, Sept. 2007 (cit. on p. 6).
- [Nor09] Ulf Norell. “Dependently Typed Programming in Agda”. In: *Proceedings of the 4th International Workshop on Types in Language Design and Implementation*. TLDI ’09. Savannah, GA, USA: ACM, 2009, pp. 1–2. ISBN: 978-1-60558-420-1 (cit. on p. 6).
- [NPS90] Bengt Nordström, Kent Peterson, and Jan M. Smith. *Programming in Martin-Löf’s Type Theory*. Vol. 7. International Series of Monographs on Computer Science. NY: Oxford University Press, 1990 (cit. on p. 6).
- [Red18] The RedPRL Development Team. *redtt*. 2018. URL: <http://www.github.com/RedPRL/redtt> (cit. on p. 6).
- [SAG19] Jonathan Sterling, Carlo Angiuli, and Daniel Gratzer. “Cubical Syntax for Reflection-Free Extensional Equality”. In: *4th International Conference on Formal Structures for Computation and Deduction (FSCD 2019)*. Ed. by Herman Geuvers. Vol. 131. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2019, 31:1–31:25. ISBN: 978-3-95977-107-8. DOI: [10.4230/LIPIcs.FSCD.2019.31](https://doi.org/10.4230/LIPIcs.FSCD.2019.31). arXiv: [1904.08562](https://arxiv.org/abs/1904.08562) (cit. on pp. 4, 5, 8, 17).
- [Sch87] Peter Schroeder-Heister. “Structural Frameworks with Higher-level Rules: Philosophical Investigations on the Foundations of Formal Reasoning”. Habilitation. Fachgruppe Philosophie, Universität Konstanz, 1987 (cit. on p. 7).
- [See84] R. A. G. Seely. “Locally cartesian closed categories and type theory”. In: *Mathematical Proceedings of the Cambridge Philosophical Society* 95.1 (1984), pp. 33–48. DOI: [10.1017/S0305004100061284](https://doi.org/10.1017/S0305004100061284) (cit. on p. 5).
- [SH18] Jonathan Sterling and Robert Harper. “Guarded Computational Type Theory”. In: *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*. Oxford, United Kingdom: ACM, 2018. ISBN: 978-1-4503-5583-4. arXiv: [1804.09098](https://arxiv.org/abs/1804.09098) (cit. on p. 2).
- [Shu15] Michael Shulman. “Univalence for inverse diagrams and homotopy canonicity”. In: *Mathematical Structures in Computer Science* 25.5 (2015), pp. 1203–1277. DOI: [10.1017/S0960129514000565](https://doi.org/10.1017/S0960129514000565) (cit. on p. 6).
- [Ste18] Jonathan Sterling. *Algebraic Type Theory and Universe Hierarchies*. Dec. 2018. arXiv: [1902.08848](https://arxiv.org/abs/1902.08848) [[math.LO](https://arxiv.org/abs/1902.08848)] (cit. on p. 8).
- [Str05] Thomas Streicher. “Universes in toposes”. In: *From Sets and Types to Topology and Analysis: Towards practical foundations for constructive mathematics*. Ed. by Laura Crosilla and Peter Schuster. Vol. 48. Oxford Logical Guides. Oxford: Oxford University Press, 2005, pp. 78–90. ISBN: 978-0-19-856651-9. DOI: [10.1093/acprof:oso/9780198566519.001.0001](https://doi.org/10.1093/acprof:oso/9780198566519.001.0001) (cit. on pp. 7, 13, 16).

- [Tai67] W. W. Tait. “Intensional Interpretations of Functionals of Finite Type I”. In: *The Journal of Symbolic Logic* 32.2 (1967), pp. 198–212. ISSN: 00224812. URL: <http://www.jstor.org/stable/2271658> (cit. on p. 1).
- [Tay99] Paul Taylor. *Practical Foundations of Mathematics*. Cambridge studies in advanced mathematics. Cambridge, New York (N. Y.), Melbourne: Cambridge University Press, 1999. ISBN: 0-521-63107-6 (cit. on p. 12).
- [Uem17] Taichi Uemura. “Fibred Fibration Categories”. In: *Proceedings of the 32nd Annual ACM/IEEE Symposium on Logic in Computer Science*. Reykjavik, Iceland: IEEE Press, 2017, 24:1–24:12. ISBN: 978-1-5090-3018-7. URL: <http://dl.acm.org/citation.cfm?id=3329995.3330019> (cit. on p. 6).
- [Uem19] Taichi Uemura. *A General Framework for the Semantics of Type Theory*. 2019. arXiv: 1904.04097 (cit. on pp. 5–9, 13).
- [VMA19] Andrea Vezzosi, Anders Mörtberg, and Andreas Abel. “Cubical Agda: A Dependently Typed Programming Language with Univalence and Higher Inductive Types”. In: *Proceedings of the 24th ACM SIGPLAN International Conference on Functional Programming*. ICFP ’19. Boston, Massachusetts, USA: ACM, 2019. DOI: [10.1145/3341691](https://doi.org/10.1145/3341691) (cit. on p. 6).
- [WB18] Paweł Wieczorek and Dariusz Biernacki. “A Coq Formalization of Normalization by Evaluation for Martin-Löf Type Theory”. In: *Proceedings of the 7th ACM SIGPLAN International Conference on Certified Programs and Proofs*. CPP 2018. Los Angeles, CA, USA: ACM, 2018, pp. 266–279. ISBN: 978-1-4503-5586-5. DOI: [10.1145/3167091](https://doi.org/10.1145/3167091) (cit. on p. 2).
- [Wra74] Gavin Wraith. “Artin glueing”. In: *Journal of Pure and Applied Algebra* 4.3 (1974), pp. 345–348. ISSN: 0022-4049. DOI: [https://doi.org/10.1016/0022-4049\(74\)90014-0](https://doi.org/10.1016/0022-4049(74)90014-0) (cit. on pp. 1, 11).