Scott's Representation Theorem and the Univalent Karoubi Envelope

Arnoud van der Leer
Delft University of Technology, The Netherlands

Kobe Wullaert
Delft University of Technology, The Netherlands

Benedikt Ahrens
Delft University of Technology, The Netherlands

Abstract

Lambek and Scott constructed a correspondence between simply-typed lambda calculi and Cartesian closed categories. Scott's Representation Theorem is a cousin to this result for *untyped* lambda calculi. It states that every untyped lambda calculus arises from a reflexive object in some category.

We present a formalization of Scott's Representation Theorem in univalent foundations, in the (Rocq-)UniMath library. Specifically, we implement two proofs of that theorem, one by Scott and one by Hyland. We also explain the role of the Karoubi envelope – a categorical construction – in the proofs and the impact the chosen foundation has on this construction. Finally, we report on some automation we have implemented for the reduction of λ-terms.

2012 ACM Subject Classification Theory of computation → Denotational semantics; Theory of computation → Logic and verification

Keywords and phrases Lambda calculi, algebraic theories, categorical semantics, Karoubi envelope, formalization, Rocq-UniMath, univalent foundations

Digital Object Identifier 10.4230/LIPIcs.ITP.2025.33

Related Version *Preprint*: `https://arxiv.org/abs/2506.22196`

Supplementary Material *Software (Source Code)*: `https://github.com/UniMath/UniMath/tree/5941f44e3c13d2ac385f5a8b9b486c0088dd3f46`
archived at `swh:1:dir:93afb3f22a54064c0baa8493bd326f385fc96ebe`

Acknowledgements We thank Mohamed Barakat, Tom de Jong, and Niels van der Weide, as well as the anonymous referees for valuable feedback on versions of this paper.We gratefully acknowledge the work by the Rocq development team in providing the Rocq proof assistant and surrounding infrastructure, as well as their support in keeping UniMath compatible with Rocq.

1 Introduction

We present a formalization of two proofs of Dana Scott's Representation Theorem (SRT) and a detailed comparison between them, all incorporated into the UniMath library [42] of univalent mathematics, based on the proof assistant Rocq (formerly named Coq) [35].

SRT provides denotational semantics for the untyped lambda calculus. It is a cousin to the arguably better-known result by Lambek and P.J. Scott [25], which provides denotational semantics for the *simply-typed* lambda calculus (Section 1.1). Specifically, SRT states that lambda calculi can be modelled by certain objects in certain categories, and that every lambda calculus arises as such an object (Section 1.2).

In this work, based on [38], we formalize the original proof of SRT by Dana Scott [32] and a more modern proof by Martin Hyland [23]. We formalize them in univalent foundations, which are more granular than traditional set-theoretic foundations. This shows strongly in our analysis of these proofs: univalent foundations provide exactly the right language to explain the fundamental difference between the two constructions.

In the remainder of the introduction, we review Lambek and Scott's correspondence between Cartesian closed categories and simply-typed lambda calculi in Section 1.1. Afterwards, in Section 1.2, we give an introduction to SRT. In Section 1.3, we provide a brief overview of univalent foundations and category theory therein, to the extent used in this paper. In Section 1.4, we give an overview of the paper and pointers to the formalization.

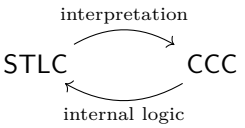
1.1 The Correspondence Between Categories and Typed λ-calculi

We sketch the correspondence between simply-typed lambda calculi (STLCs) and Cartesian closed categories. A STLC is a simply-typed language with at least abstraction and application. The type system of the language should be closed under function types $A \to B$. Lambek and Scott [25] describe the construction of denotational models for such calculi.

▶ **Definition 1.** *A **Cartesian closed category**, abbreviated **CCC**, is a category* $\mathbf{C}$*, equipped with a terminal object* 1*, binary products* $A \times B$*, and exponentials* B^A*.*

Now, a Cartesian closed category $\mathbf{C}$ gives rise to a STLC, its "internal language": roughly, take the objects of the category to be types, and morphisms $f : A \to B$ of the category to be terms of type B in context A. Conversely, any simply-typed lambda calculus gives rise to a Cartesian closed category, as follows. A type A of the calculus gives an object $[\![A]\!]$ in the category. A term $t : A$ of type A in context $x_1 : A_1, \ldots, x_n : A_n$ gives a morphism $[\![t]\!] : [\![A_1]\!] \times \ldots \times [\![A_n]\!] \to [\![A]\!]$.

Internal logic and interpretation, together, provide a *correspondence between simply-typed lambda calculi and Cartesian closed categories*:



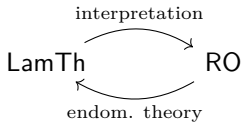
(1)

1.2 Scott's Representation Theorem

In 1969, Scott gave a famous series of lectures [33] where he initially claimed that untyped λ-calculi did not have (non-trivial) categorical models. Intuitively, for an object D of some category to be a model of the untyped lambda calculus, one would need functions $D \leftrightarrows D^D$ modelling abstraction and application, such that the composition $D^D \to D \to D^D$ is the identity – to model β-equality. (For η-equality, the other composite must be the identity on D.) In particular, this would mean that the function space D^D can be embedded into D itself. In the category of sets, this is only possible for $D = 1$. However, as Scott's lectures progressed, he realized that his initial claim was incorrect. In the end, Scott constructed a model $\mathcal{D}_\infty$ in the category of directed complete partial orders (dcpos). Later, models such as *Scott's graph model* and *Böhm's tree model* were constructed; see, e.g., [11, Part V].

To arrive at a result similar to the correspondence of Diagram (1), SRT abstracts away from particular calculi and models. We use Hyland's [23] axiomatization of lambda calculi as λ-theories, which are particular *algebraic theories* from universal algebra. A λ-theory is an algebraic theory equipped with operations modelling abstraction and application, satisfying β- (and potentially η-) equality. On the semantic side, a model of a lambda calculus is defined to be a *reflexive object* D in a Cartesian closed category; that is, a diagram $D \leftrightarrows D^D$ that exhibits D^D as a retract of D (see Definition 16).

For every reflexive object, one can construct its so-called *endomorphism theory*. Then SRT can be stated as follows: every untyped λ-calculus arises as the endomorphism theory of a reflexive object. To summarize, SRT provides the following diagram, where LamTh consists of λ-theories and RO of pairs of a Cartesian closed category and a reflexive object therein:



(2)

Unlike in the case of *simply-typed* lambda calculi, this diagram does not represent an equivalence, but only a *section-retraction pair*. That is, if we start with a lambda calculus, interpret it and then take the endomorphism theory of the interpretation, we end up with "the same" lambda calculus. However, starting with a reflexive object, taking its endomorphism theory and then the interpretation does not necessarily yield the original reflexive object; see also Remark 19.

Scott and Hyland gave different constructions of the interpretation of a lambda calculus and the proof that this forms a section. In this paper, we formalize both, and compare them, in the light of univalent foundations.

1.3 A Quick Tour of Univalent Foundations and Univalent Categories

In this section, we give a brief introduction to univalent foundations and univalent category theory. Other short introductions can be found in [18, 9]. A comprehensive introduction is given in the HoTT book [37]; in particular, [37, Chap. 9] discusses univalent categories.

Univalent foundations (UF) are foundations of mathematics satisfying the univalence principle. This principle says, roughly, that two equivalent mathematical objects satisfy all the same properties. Technically, UF are an extension of Martin-Löf type theory (MLTT) [27] by the univalence axiom and, optionally, (certain) higher inductive types. We assume the reader is familiar with the basics of MLTT as discussed, for instance, in [37, Chap. 1]. To fix notation: we write $g \circ f$ for the composition of functions $f : X \to Y$ and $g : Y \to Z$.

Given a type X and elements $x, y : X$, we denote by $x = y$ the identity type between x and y. The type $x = y$ can contain multiple distinct elements, that is, different identity terms from x to y. For any $x : X$, we have the identity term $\mathsf{refl}(x) : x = x$. In particular, given two types S and T (elements of some universe type), we have the type $S = T$. There is also a seemingly weaker notion of sameness, called *equivalence*, written $S \simeq T$; its elements are functions $S \to T$ with a two-sided inverse (see [37, Chap. 4] for details). We have a function idtoequiv : $(S = T) \to (S \simeq T)$, sending $\mathsf{refl}(S)$ to the identity equivalence on S. The **univalence axiom** says that this map is an equivalence, for any S and T. In particular, every structure on types transports along an equivalence of types.

In univalent foundations, types are *stratified* according to how "complex" their identity types are: Let T be a type. It is **contractible** if there is an equivalence $e : T \simeq 1$ to the unit type 1. It is a **proposition** if any two of its elements are identical: $\prod_{s,t:T} s = t$. It is a **set** if all of its identity types $s = t$, for $s, t : T$, are propositions (identity proofs in T are unique). It is a **groupoid** if all of its identity types $s = t$, for $s, t : T$, are sets. This stratification of types can be continued recursively, but the aforementioned definitions suffice for the purpose of this paper.

Given a type A, we form a new type $\|A\|$, the **propositional truncation of** A; $\|A\|$ is a proposition, and inhabited if and only if A is inhabited. When postulating that something "merely exists in T with property P" (such as in Definition 44), we formalize it as $\|\sum_{x:T} P(x)\|$.

There are two notions of **category** ( ) in univalent foundations. Both kinds consist, in particular, of a type O of objects and a dependent type $M : O \to O \to \mathbf{Type}$ of morphisms. Given a category $\mathbf{C}$, we write $\mathbf{C}(X, Y)$ or $X \to Y$ for the type of morphisms between objects X and Y. A category also has suitably typed composition and identity operations, and proofs of the associativity and unitality of composition. To ensure that unitality and associativity are propositions, we require that all the $\mathbf{C}(X, Y)$ are sets. Just like with functions, we write $g \circ f$ for the composition of morphisms $f : \mathbf{C}(X, Y)$ and $g : \mathbf{C}(Y, Z)$. For details, we refer to [5, Def. 3.1] – there, categories are called **pre**categories.

A (pre)category as described above does not correspond to a category in Voevodsky's model of univalent foundations in simplicial sets [24]; it has data that is not usually present in a category, namely a potentially non-trivial identity type on the type O of objects. There are two ways to "kill" that data. Firstly, we could assert that the type O of objects should also form a set in the above sense; this leads to the notion of **setcategory**. Secondly, we could assert that the identities $a = b$ of $a, b : O$ correspond exactly to **isomorphisms** $a \cong b$ of the category; this leads to the notion of **univalent category**. More precisely, in a univalent category $\mathbf{C}$ we demand that for every $a, b : \mathbf{C}$, the map $\mathsf{idtoiso}_{a,b} : (a = b) \to (a \cong b)$, mapping $\mathsf{refl}(a)$ to the identity isomorphism on a, is an equivalence, in analogy to the univalence axiom. One can show that the type of objects of a univalent category forms a groupoid.

There are thus two theories of categories in univalent foundations. Setcategory theory behaves much like traditional category theory; however, many naturally occurring categories are not setcategories. On the other hand, there are many univalent categories. Importantly, the category **Set** of sets (of a fixed universe) in the sense described above, is univalent; its objects are pairs of a type X (of a fixed universe) together with a proof that X is a set, and morphisms are type-theoretic functions. Roughly, univalence of this category is a consequence of the univalence axiom (for that universe). Furthermore, for two categories $\mathbf{C}$ and $\mathbf{D}$, the functor category $[\mathbf{C}, \mathbf{D}]$ is univalent if $\mathbf{D}$ is univalent; hence, in particular, presheaf categories $P\mathbf{A} = [\mathbf{A}^{\mathsf{op}}, \mathbf{Set}]$ are univalent. Still, not every category is univalent, e.g., the category $\bullet \leftrightarrows \bullet$ with two distinct but isomorphic objects, or the set-Karoubi envelope of Section 3.1, see Example 42. However, every category has a univalent replacement, which we call its **Rezk completion**:

▶ **Theorem 2** ([5, Thm. 8.5]). *For every category* $\mathbf{C}$*, there exists a univalent category* $\mathbf{D}$ *with a fully faithful and essentially surjective functor* $\iota : \mathbf{C} \hookrightarrow \mathbf{D}$.

In summary, univalent foundations give us two flavors of category theory: category theory up to isomorphism (setcategories) and category theory up to adjoint equivalence (univalent categories). We demonstrate the difference between Scott's and Hyland's proof of SRT by showing that Scott's construction happens in the realm of setcategories, whereas Hyland's construction happens in the realm of univalent categories. Furthermore, Hyland's construction yields the Rezk completion of Scott's construction.

1.4 Synopsis and Guide to the Formalization

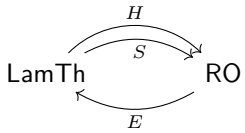
In Section 2 we present our formalization of SRT as shown in Equation (2). More precisely, we formalize two different constructions of the function labelled "interpretation", one by Scott, and one by Hyland, and we prove that both are sections to the function associating to a reflexive object its endomorphism theory. In Section 3 we zoom in on the different interpretation functions. Both use a variant of the *Karoubi envelope* of a category. We provide a novel analysis and comparison of the two variants, through the lens of univalent foundations, and show how the two variants relate Scott's and Hylands constructions. In Section 4 we describe a tactic for the manipulation of lambda terms. In Section 5 we discuss related work.

Most of the definitions and results presented in this paper are formalized and computer-checked in UniMath [42], a library of univalent mathematics based on Rocq. Our code has been integrated into the UniMath library, and comprises more than 11000 lines of code, split up more or less evenly between definitions and proofs. We can identify the following main components: the definitions of the basic categories with their object and morphism types (3000 lines); the proof of Scott's version of SRT, which uses a lot of reasoning about λ-terms (3000 lines); the definition of the Karoubi envelope (2000 lines); the constructions of various algebraic theories, like the endomorphism theory, and the proof of the relation between the representation theorems (both 1000 lines).

Throughout the paper, definitions and results are decorated with links, for example, , to an HTML version of UniMath. That HTML version is derived from commit 5941f44 of UniMath. Proof-checking and creation of the HTML documentation can be reproduced locally by following the UniMath compilation instructions. Note that for technical reasons, the formalization does not always closely follow the material in this paper, with the main difference described in Remark 21. Another point of complication is the necessary switching between two different ways to encode n-tuples $x : X^n$, either as nested 2-tuples $((\dots((x_1, x_2), x_3), \dots), x_n)$ or as functions $x : \{1, \dots, n\} \to X$.

2 What we Formalized: Scott's Representation Theorem

In this section, we present our formalization of SRT. In Section 2.1, we define the type LamTh of λ-theories, the type RO of reflexive objects, and the endomorphism theory generated by a reflexive object, in the form of a function $E : \mathsf{RO} \to \mathsf{LamTh}$ (E for "endomorphism theory"). In Sections 2.2 and 2.3, we present Scott's and Hyland's constructions of a section to $E : \mathsf{RO} \to \mathsf{LamTh}$, that is, of functions $S, H : \mathsf{LamTh} \to \mathsf{RO}$ (where S and H stand for "Scott" and "Hyland", respectively) such that $E \circ S = \mathsf{id}_{\mathsf{LamTh}} = E \circ H$.



(3)

▶ Remark 3. It might seem surprising that we can show $E \circ S = \mathsf{id}_{\mathsf{LamTh}}$ instead of just $E(S(T)) \cong T$ for $T : \mathsf{LamTh}$ and a suitable notion of isomorphism of λ-theories. We can show the seemingly stronger identity thanks to the univalence axiom. Specifically, the univalence axiom entails that isomorphisms $T \cong T'$ of λ-theories coincide with identities $T = T'$ – expressed below by the statement that the category of λ-theories is univalent (Proposition 13). That is, our statement of SRT does not make reference to (iso)morphisms of λ-theories; the construction, however, does: we construct an isomorphism first, and turn it into an identity using univalence afterwards.

2.1 Untyped λ-Calculi and Their Denotational Semantics

We give a definition of untyped λ-*calculi* (Definition 8), an extension of the concept of an *algebraic theory* from universal algebra (Definition 4). We also define reflexive objects in Cartesian closed categories, the intended denotational semantics for such calculi.

▶ **Definition 4** ( ). *We define an* ***algebraic theory*** T *to be a sequence of sets* T_n *indexed over* $\mathbb{N}$ *with for all* $1 \leq i \leq n$ *elements ("variables" or "projections")* $x_{n,i} : T_n$ *(we often leave* n *implicit and write* x_i*), together with a substitution operation* $\_ \bullet \_ : T_m \times T_n^m \to T_n$ *for all* m *and* n*, such that for all* $1 \leq j \leq l$*,* $f : T_l$*,* $g : T_m^l$ *and* $h : T_n^m$*,*

$$x_j \bullet g = g_j, \quad f \bullet (x_{l,i})_i = f \quad \text{and} \quad (f \bullet g) \bullet h = f \bullet (g_i \bullet h)_i.$$

Algebraic theories are also known as *(abstract) clones*; they are similar to *operads* – see, e.g., [22, 36] for details.

▶ **Definition 5** (🍤). *A **morphism** f **between algebraic theories** T and T' is a sequence of functions $f_n : T_n \to T'_n$ (we usually leave the n implicit and just write f if the context is clear) such that for all $1 \leq j \leq n$, $s : T_m$ and $t : T_n^m$, $f_n(x_j) = x_j$, and $f_n(s \bullet t) = f_m(s) \bullet (f_n(t_i))_i$.*

Algebraic theories and their morphisms form a category **AlgTh** (🍤). Furthermore, an isomorphism $S \cong T$ of algebraic theories consists of pointwise bijections $f_n : S_n \cong T_n$ that respect the variables and substitution. By univalence, each f_n corresponds to an identity $p_n : S_n = T_n$ also respecting the algebraic structure. Again by univalence, the p_n's combine into an identity of algebraic theories $S = T$. Hence:

▶ **Proposition 6** (🍤). **AlgTh** *is univalent.*

▶ **Example 7** (🍤🍤). We have a *forgetful* functor from **AlgTh** to **Set** sending an algebraic theory T to the set T_0. Conversely, for every set A one can construct an algebraic theory $T(A)$ with $T(A)_n = \{1, \ldots, n\} \sqcup A$. The variables are given by $x_{n,i} = i : T(A)_n$, for $0 < i \leq n$. The substitution $f \bullet g$ is defined as follows: if $f : A$, then $f \bullet g = f$; if $1 \leq f \leq n$, then $f \bullet g = g_f$. Then, $A \mapsto T(A)$ is part of a functor $T : \mathbf{Set} \to \mathbf{AlgTh}$. Furthermore, T is a left adjoint to the forgetful functor.

Let $\iota_{m,n} : T_m \to T_{m+n}$ be the "weakening" function $f \mapsto f \bullet (x_{m+n,1}, \ldots, x_{m+n,m})$. Note that $\iota_{m,n}(f) \bullet g = f \bullet (g_i)_{i \leq m}$ and $\iota_{m,n}(f \bullet g) = f \bullet (\iota_{m,n}(g_i))_i$.

▶ **Definition 8** (🍤). *A **λ-theory** is an algebraic theory L, together with sequences of functions $\lambda_n : L_{n+1} \to L_n$ and $\rho_n : L_n \to L_{n+1}$, such that for all $f : L_{m+1}$, $g : L_m$ and $h : L_n^m$,*

$$\lambda_m(f) \bullet h = \lambda_n(f \bullet ((\iota_{n,1}(h_i))_i + (x_{n+1}))) \quad \text{and} \quad \rho_n(g \bullet h) = \rho_m(g) \bullet ((\iota_{n,1}(h_i))_i + (x_{n+1})).$$

Furthermore, we assume that L satisfies β-equality, that is $\rho_n \circ \lambda_n = \mathsf{id}_{L_n}$ for all n.

We also formalized the notion of λ-theory without β-equality, and of λ-theory with both β- and η-equality (meaning $\lambda_n \circ \rho_n = \mathsf{id}_{L_n}$), but we only formalized the main results for the λ-theories of Definition 8.

▶ **Example 9** (🍤). An important example of a λ-theory is provided by the λ-calculus without constants Λ. We present it here as a higher inductive type, to have it satisfy β-equality. For $i : \{1, \ldots, n\}$, $s, t : \Lambda_n$, $u : \Lambda_{n+1}$ and $v_1, \ldots, v_n : \Lambda_m$, it has constructors

$$\mathtt{Var}_n(i) : \Lambda_n, \quad \mathtt{App}_n(s,t) : \Lambda_n, \quad \mathtt{Abs}_n(u) : \Lambda_n \quad \text{and} \quad \mathtt{Subst}_{n,m}(s, (v_1, \ldots, v_n)) : \Lambda_m,$$

together with identities about the interaction between `Subst` with every constructor, as well as the identity $\mathtt{App}(\mathtt{Abs}(f), g) = \mathtt{Subst}(f, (\mathtt{Var}(1), \ldots, \mathtt{Var}(n), g))$ for β-equality. The functions $\rho_n : \Lambda_n \to \Lambda_{n+1}$ are given by $\rho(f) = \mathtt{App}(\mathtt{Subst}(f, (x_1, \ldots, x_n)), x_{n+1})$. ⌟

▶ Remark 10 (🍤). Because the UniMath community does not permit the usage of inductive types in the core library, we defined Λ *synthetically*, instead of constructing it: our formalization takes a hypothesis Λ of a type describing the λ-calculus without constants as a sequence of sets $(\Lambda_n)_n$ with functions `Var`, `App`, `Abs` and `Subst` that satisfy the correct identities. The type of Λ also contains an *induction principle*, which allows one to construct terms of type $\prod_n \prod_{f:\Lambda_n} T_n(f)$ for a family of sets T. The non-dependent version of this induction principle takes a set X and functions $var_n : \{1, ..., n\} \to X$, $app_n : X \times X \to X$, $abs_n : X \to X$ and $subst_n : X \times X^n \to X$, satisfying the same identities as the constructors of Λ, and gives functions $f_n : \Lambda_n \to X$ such that $f(\mathtt{Var}(i)) = var(i)$, $f(\mathtt{App}(s,t)) = app(f(s), f(t))$ etc.

▶ **Definition 11** (🥧). *A **morphism** f **between** λ**-theories** L and L' is an algebraic theory morphism f such that $f_n \circ \lambda_n = \lambda_n \circ f_{n+1}$ and $\rho_n \circ f_n = f_{n+1} \circ \rho_n$.*

λ-theories and their morphisms form a category **LamTh** (🥧) with LamTh as the underlying type of objects.

▶ **Example 12** (🥧). For any λ-theory L, there is a unique morphism $i : \Lambda \to L$, defined using structural induction on the terms of Λ: it sends variables to variables, substitution to substitution, etc. This makes Λ into the initial object of **LamTh**.

We can prove the following analogously to Proposition 6:

▶ **Proposition 13** (🥧). **LamTh** *is univalent.*

Proposition 13 expresses that identities $T = T'$ of λ-theories are equivalent to isomorphisms $T \cong T'$; this enables us to express SRT purely on the level of types, without needing to mention (iso)morphisms of λ-theories, as discussed in Remark 3.

Let L be a λ-theory. Hyland shows that $\rho_n(f)$ corresponds to fx_{n+1}, which is f applied to x_{n+1} [23, Section 3.1]. Now consider the element $\rho(x_{1,1}) : L_2$.

▶ **Definition 14** (🥧). *Using the substitution, we have binary operations on the L_n, sending $(f, g) : L_n \times L_n$ to $\rho(x_{1,1}) \bullet (f, g) : L_n$.*

▶ Remark 15. This means that L has the structure of a λ-calculus with β-equality, with variables x_i, application $fg = \rho(x_{1,1}) \bullet (f, g)$ and abstraction $\lambda x_{n+1}, f = \lambda(f)$.

Now we are ready to talk about denotational semantics for the untyped λ-calculus, for which we will need the following definition.

▶ **Definition 16** (🥧). *If $\mathbf{C}$ is a category with binary products, a **reflexive object** in $\mathbf{C}$ is an object X such that the exponential X^X exists, and with a retraction from X onto X^X: morphisms $f : X \to X^X$ and $g : X^X \to X$ such that $f \circ g = \mathsf{id}_{X^X}$.*

▶ **Definition 17** (🥧). *We define a function $E : \mathsf{RO} \to \mathsf{LamTh}$ as follows. Let X be a reflexive object in a category $\mathbf{C}$, with the retraction given by $\mathrm{abs} : X^X \to X$ and $\mathrm{app} : X \to X^X$.*

*The **endomorphism theory** $E(X)$ of X is the algebraic theory given by $E(X)_n = \mathbf{C}(X^n, X)$ with projections as variables $x_{n,i} : X^n \to X$ and a substitution that sends $f : X^m \to X$ and $g_1, \ldots, g_m : X^n \to X$ to $f \circ \langle g_i \rangle_i : X^n \to X$.*

If $\varphi_Y : \mathbf{C}(X \times Y, X) \xrightarrow{\sim} \mathbf{C}(Y, X^X)$ is the bijection given by the exponential X^X, we can give $E(X)$ a λ-theory structure by setting, for $f : E(X)_{n+1}$ and $g : E(X)_n$,

$$\lambda(f) = \mathrm{abs} \circ \varphi_{X^n}(f) \quad \text{and} \quad \rho(g) = \varphi^{-1}_{X^n}(\mathrm{app} \circ g).$$

β-equality for $E(X)$ follows immediately from the fact that $\mathrm{app} : X \to X^X$ is a retraction.

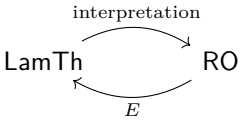
The proofs that $E(X)$ is an algebraic theory follow mainly from properties of the product and naturality of φ_Y. Note that $E(X)$ would satisfy η-equality if app were a section.

▶ **Example 18.** As a trivial example, take $X = \{\star\}$ in the category of sets. The set of functions $X^X = X \to X$ is isomorphic to X, and so we get an endomorphism theory $E(X)$ with $E(X)_n = \{\star\}$.

Because of cardinality reasons, Example 18 is the only reflexive object in the category of sets. Hence, for nontrivial models of the untyped λ-calculi, we have to look in categories different from **Set**. Scott's model D_∞ lives in the category of dcpos whose morphisms are (continuous) section-retraction pairs [31].

SRT states that we can represent every untyped λ-calculus as an endomorphism theory of some reflexive object:

▶ **Theorem.** *The function E of Definition 17 has a right inverse:*



(4)

In the following two subsections, we will discuss two proofs of this theorem: the original proof by Dana Scott (Theorem 24) and a more abstract proof by Martin Hyland (Theorem 36).

▶ Remark 19 (🍊). The constructions by Scott and Hyland yield different reflexive objects. In particular, Hyland's category has an empty object with no morphisms into it, whereas Scott's category always has at least one morphism between any two objects. By the representation theorems, the function E sends both of these to the same λ-theory, so E is not injective. Therefore, contrary to the situation for STLCs in Section 1.1, E is not an equivalence.

▶ Remark 20. By definition, the λ-theories we consider satisfy β-equality. A variant of SRT also holds for λ-theories satisfying both β and η-equality.

▶ Remark 21. We use the technology of *displayed categories* [6] for implementing many of the concepts in this section. A displayed category $\mathbf{D}$ over a category $\mathbf{C}$ gives the objects and morphisms of $\mathbf{C}$ "additional structure". Consider how a λ-theory is an algebraic theory, together with operations ρ and λ, and a λ-theory morphism is an algebraic theory morphism, that preserves ρ and λ. This is formalized by defining **LamTh** to be displayed over **AlgTh**. In turn, **AlgTh** is displayed over $\mathbf{Set}^{\mathbb{N}}$. The goal of displayed categories is to build categories of complicated objects and morphisms in "layers", incrementally adding more structure. These layers can be built and reasoned about modularly. The proof of univalence of those categories, Propositions 6 and 13, is then especially simple: it relies on univalence of **Set** and univalence of the layers leading up to **LamTh**. In the formalization, after the categories have been defined, we define the object types in terms of their categories, to minimize the amount of duplicate code. For example, we define a λ-theory to be an object of **LamTh**. A drawback to this approach is that it somewhat obfuscates the precise definition of the objects and morphisms of such a category: the structure of a λ-theory is hidden away in a stack of four displayed categories, although the constructor and accessors give some hints about what a λ-theory consists of.

2.2 Scott's Construction

In this subsection, we will give Scott's original proof [32] of SRT.

It is a fairly syntactical proof, making heavy use of the operations of the λ-calculus for the various constructions. We first define the category $\mathbf{R}$ (Definition 22) and show that it has products and exponential objects. Then we construct an object in this category and show that it is a reflexive object. Lastly, we construct the endomorphism theory and show that it is isomorphic to the λ-theory that we started with (Theorem 24).

Let L be a λ-theory. First of all, for $a_1, a_2 : L_0$, we define

$$a_1 \circ a_2 = \lambda x_1, a_1(a_2 x_1); \qquad \pi_i = \lambda x_1, x_1(\lambda x_2 x_3, x_{i+1});$$
$$(a_1, a_2) = \lambda x_1, x_1 a_1 a_2; \qquad \langle a_1, a_2 \rangle = \lambda x_1, (a_1 x_1, a_2 x_1).$$

Although, actually, since every one of these starts with a λ-abstraction, we need to lift the constants a_i to $\iota_{0,1}(a_i) : L_1$ to make the definitions above typecheck. Note that $\pi_i(a_1, a_2) = a_i$ and $\pi_i \circ \langle a_1, a_2 \rangle = \lambda x_1, a_i x_1$.

We define $(a_i)_i = (a_1, \dots, a_n) = ((\dots(((\lambda x_1, x_1), a_1), a_2), \dots), a_n)$ (analogous for $\langle a_i \rangle_i$) and correspondingly $\pi_{n,i} = \pi_2 \circ {\pi_1}^{n-i}$.

Scott introduces the following category, referred to as the **category of retracts**:

▶ **Definition 22** (). *Let* $\mathbf{R}$ *be the category with as objects the* $A : L_0$ *such that* $A \circ A = A$ *and as morphisms* $f : A \to B$ *the* $f : L_0$ *such that* $B \circ f = f = f \circ A$*. The composition is given by the composition of* λ*-terms and the identity on* $A : \mathbf{R}$ *is given by* A *itself.*

$\mathbf{R}$ has a terminal object, given by $I = \lambda x_1 x_2, x_2 : \mathbf{R}$ (). It is terminal because for $f : A \to I$, we have $f = I \circ f = I$.

$\mathbf{R}$ also has binary products $A_1 \times A_2 = \langle p_1, p_2 \rangle$ for $p_i = A_i \circ \pi_i$ the projections, with product morphisms $\langle f, g \rangle$ ().

For $B, C : \mathbf{R}$, we have an exponential object $C^B = \lambda x_1, C \circ x_1 \circ B$, with a bijection $\psi : \mathbf{R}(A \times B, C) \xrightarrow{\sim} \mathbf{R}(A, C^B)$, given by $\psi(f) = \lambda x_1 x_2, f(x_1, x_2)$ and $\psi^{-1}(f) = \lambda x_1, g(\pi_1 x_1)(\pi_2 x_1)$ ().

Now we can define our reflexive object:

▶ **Definition 23** (). *We define* $U : \mathbf{R}$*, given by the identity* $\lambda x_1, x_1$*. For all* $A : \mathbf{R}$*, we can also consider* A *as both a morphism* $r_A : \mathbf{R}(U, A)$ *and a morphism* $s_A : \mathbf{R}(A, U)$ *with* $r_A \circ s_A = \mathsf{id}_A$*, exhibiting* A *as a retract of* U*. In particular,* U *is reflexive.*

This allows us to give Scott's proof of the representation theorem:

▶ **Theorem 24** (). *The function* E *of Definition 17 has a right inverse* S*, given by* $S(L) = (\mathbf{R}, U)$*.*

Proof. Let L be a λ-theory. As mentioned in Definition 23, U is a reflexive object, so the endomorphism theory $E(U)$ has a λ-theory structure.

We have bijections $\psi_n : E(U)_n \xrightarrow{\sim} L_n$, given by $\psi_n(f) = \iota_{0,n}(f)(x_i)_i$, and $\psi_n^{-1}(g) = \lambda x_1, g \bullet (\pi_{n,i} x_1)_i$. These are bijections because $E(U)_n = \{f : L_0 \mid f \circ U^n = f\}$, with $U^n = \langle \pi_{n,i} \rangle_i$. Explicitly, $E(U)$ has variables $x_{n,i} = \pi_{n,i}$, substitution $f \bullet g = f \circ \langle g_i \rangle_i$, abstraction $\lambda(f) = \lambda x_1 x_2, \iota_{0,2}(f)(x_1, x_2)$ and application $\rho(g) = \lambda x_1, \iota_{0,1}(g)(\pi_1 x_1)(\pi_2 x_1)$ for $f : E(U)_{n+1}$ and $g : E(U)_n$. Using this, it is pretty straightforward to check that the ψ_n respect the λ-theory structure, so ψ is an isomorphism of λ-theories. Then, univalence of **LamTh** gives an identity $E(S(L)) = L$. ◀

2.3 Hyland's Construction

While Scott proves the representation theorem in a very syntactical way, constructing the category $\mathbf{R}$ for his representation theorem using the λ-calculus operations, Hyland uses more machinery from category theory, and works with a different category: the category of (T-)presheaves $\mathbf{Pshf}_T$ of a λ-theory T.

▶ Remark 25. The T-presheaves can be described as the presheaves over the category $\mathbf{L}$, referred to as the *Lawvere theory* associated to T (Definition 33). Instead of working with the general construction, we spell out what this means, and refer to presheaves as the unfolded version hereof. In Lemma 34, we show that these notions coincide.

In this section, we define the category of presheaves (Definition 26), and construct a reflexive object herein Corollary 32. Furthermore, we show that the endomorphism theory of the reflexive object is isomorphic to the λ-theory that we started with (Theorem 36).

▶ **Definition 26** (). *A **presheaf** P for an algebraic theory T is a sequence of sets P_n indexed over $\mathbb{N}$, together with an action*

$$\_ \bullet \_ : P_m \times T_n^m \to P_n \qquad \textit{(note that we use the same notation as in Definition 4)}$$

for all m, n, such that for all $t : P_l$, $f : T_m^l$ and $g : T_n^m$,

$$t \bullet (x_{l,i})_i = t, \quad \textit{and} \quad (t \bullet f) \bullet g = t \bullet (f_i \bullet g)_i.$$

A prime example of a T-presheaf is T itself:

▶ **Definition 27** (). *For an algebraic theory T, its **theory presheaf** (also denoted T) is the T-presheaf given by the sets of T. Its T-action is the substitution operation of T.*

▶ **Definition 28** (). *For an algebraic theory T, a **morphism f between T-presheaves** P and Q is a sequence of functions $f_n : P_n \to Q_n$ such that for all $t : P_m$ and $f : T_n^m$, $f_n(t \bullet f) = f_m(t) \bullet f$.*

T-presheaves and their morphisms form the category of T-presheaves $\mathbf{Pshf}_T$ (). Because presheaf (iso)morphisms preserve $\bullet$, we have, analogous to Proposition 6:

▶ **Lemma 29** (). *$\mathbf{Pshf}_L$ is univalent.*

▶ **Definition 30** (). *Given a T-presheaf Q, we can construct a presheaf $A(Q)$ with $A(Q)_n = Q_{n+1}$ and, for $q : A(Q)_m$ and $f : T_n^m$, whose action is given by*

$$q \bullet_{A(Q)} f = q \bullet_Q (\iota_{n,1}(f_1), \dots, \iota_{n,1}(f_m), x_{n+1}).$$

This is reminiscent of the λ-theory axioms.

▶ **Lemma 31** (). *The category of T-presheaves has binary products: for presheaves P and Q, we have $(P \times Q)_n = P_n \times Q_n$ and $(p, q) \bullet t = (p \bullet t, q \bullet t)$. Furthermore, for all T-presheaves Q, $A(Q)$ is the exponential object Q^T.*

▶ **Corollary 32** (). *The theory presheaf associated to a λ-theory is reflexive.*

Proof. Given a λ-theory L, we have sequences of functions $\lambda_n : A(L)_n \to L_n$ and $\rho_n : L_n \to A(L)_n$. These commute with the L-actions, so they constitute presheaf morphisms. Furthermore, by β-equality, we have $\rho \circ \lambda = \mathsf{id}_{A(L)}$. ◀

For ease of understanding, and to make Hyland's version of SRT particularly smooth, we defined presheaves in a very algebraic way. However, in category theory, a presheaf is commonly defined to be a contravariant, set-valued functor $\mathbf{C}^{\mathsf{op}} \to \mathbf{Set}$ on some category $\mathbf{C}$. Lemma 34 justifies our definition by showing that T-presheaves are equivalent to presheaves on some category:

▶ **Definition 33** (). *Let T be an algebraic theory. We construct the category $\mathbf{L}$ with objects, morphisms, identity and composition for $f : \mathbf{L}(l, m)$, $g : \mathbf{L}(m, n)$ given by*

$$\mathbf{L}_0 = \{0, 1, 2, \dots\}, \quad \mathbf{L}(m, n) = T_m^n, \quad \mathsf{id}_n = (x_i)_i \quad \textit{and} \quad g \circ f = (g_i \bullet f)_i.$$

*We call $\mathbf{L}$ the **Lawvere theory** associated to T.*

Note that in $\mathbf{L}$, the object n behaves as the n-fold product 1^n with product projections $\pi_{n,i} = x_{n,i} : \mathbf{L}(n, 1)$.

▶ **Lemma 34** (🔫). *Let T be an algebraic theory, and $\mathbf{L}$ be its associated Lawvere theory as defined in Definition 33. The category $\mathbf{Pshf}_T$ of T-presheaves is equivalent to the category $P\mathbf{L}$ of presheaves (contravariant functors) on $\mathbf{L}$.*

Proof. A T-presheaf P corresponds to a $\mathbf{L}$-presheaf Q, where the sets P_n correspond to the action of Q on objects $Q(n)$, and the P-action $P_m \times T_n^m \to P_n$ corresponds to the action of Q on morphisms $\mathbf{L}(n,m) \times Q(m) \to Q(n)$. ◀

In the proof of the following lemma, we will make use of the so-called **Yoneda embedding** $よ : \mathbf{C} \hookrightarrow P\mathbf{C}$ of a category $\mathbf{C}$ into its presheaf category. It sends an object $X : \mathbf{C}$ to the functor $Y \mapsto \mathbf{C}(Y, X)$. Note that it is fully faithful: it gives bijections on the morphisms $\mathbf{C}(X,Y) \cong P\mathbf{C}(よ(X), よ(Y))$. A well-known lemma in category theory is the **Yoneda lemma**, which shows that for $X : \mathbf{C}$ and $Y : P\mathbf{C}$, there is an equivalence $P\mathbf{C}(よ(X), Y) \simeq Y(X)$, which is natural in X and Y.

▶ **Lemma 35** (🔫). *For $Q : \mathbf{Pshf}_T$, we have bijections $\mathbf{Pshf}_T(T^n, Q) \cong Q_n$.*

Proof. Lemma 34 shows that T-presheaves are equivalent to presheaves on the Lawvere theory $\mathbf{L}$ associated to T. Under this equivalence, $よ(n)$ corresponds to the power $T^n = (T_m^n)_m$ of the theory presheaf, for all $n : \mathbf{L}$. Then the Yoneda lemma gives a bijection $\mathbf{Pshf}_T(T^n, Q) \cong Q_n$. Explicitly, it sends $f : \mathbf{Pshf}_T(T^n, Q)$ to $f_n(x_1, \ldots, x_n)$ and $q : Q_n$ to $(t_i)_i \mapsto q \bullet t$. ◀

Now we can give Hyland's proof of the representation theorem:

▶ **Theorem 36** (🔫). *The function E of Definition 17 has a right inverse H, given by $H(L) = (\mathbf{Pshf}_L, L)$, where the last $L : \mathbf{Pshf}_L$ is the theory presheaf of Definition 27.*

Proof. Let L be a λ-theory. Recall from Corollary 32 that the theory presheaf L is a reflexive object with $(L^L)_n = L_{n+1}$, so $E(L)$ has a λ-theory structure.

Lemma 35 gives a sequence of bijections $\varphi_n : \mathbf{Pshf}_L(L^n, L) \cong L_n$ for all n, sending $f : \mathbf{Pshf}_L(L^n, L)$ to $f(x_1, \ldots, x_n)$, and conversely sending $s : L_n$ to $((t_1, \ldots, t_n) \mapsto s \bullet (t_1, \ldots, t_n))$. It considers λ-terms in n variables as n-ary functions on the λ-calculus. Therefore, φ preserves the x_i, $\bullet$, ρ and λ, which makes it into an isomorphism of λ-theories. Then, univalence of **LamTh** gives an identity $E(S(L)) = L$. ◀

So Hyland shows that the representation theorem follows largely from the fact that the functions from L_n to itself can be represented by L_{n+1}, together with the Yoneda lemma for the Lawvere theory associated to L (Lemma 35).

Observe that the proof of Theorem 36 does not require β- or η-equality, but if L has β- or η-equality, then we can immediately see (even without the isomorphism from the theorem) that the endomorphism theory also has this property.

3 The Karoubi Envelope

In this section, we relate the construction of Hyland to that of Scott. The relation is made precise by realizing that their categories are both strongly related to the *Karoubi envelope*.

Below, we characterize the Karoubi envelope via its defining universal property (Definition 37). In Sections 3.1 and 3.2, we provide two implementations hereof. Classically, these two constructions are equivalent. In univalent foundations however, we see that the former provides a setcategory, whereas the latter provides a univalent category, and that one is the Rezk completion of the other (Theorem 47).

Before stating the defining property of the Karoubi envelope, we first need the following definitions. Let $\mathbf{C}$ be a category and $X, Y : \mathbf{C}$ objects. We will denote the type of section-retraction pairs of Y onto X with

$$X \lhd Y = \sum_{r:\mathbf{C}(Y,X)} \sum_{s:\mathbf{C}(X,Y)} r \circ s = \mathsf{id}_X.$$

Now, note that for $(r, s) : X \lhd Y$, $s \circ r : \mathbf{C}(Y, Y)$ is an idempotent morphism, since $s \circ r \circ s \circ r = s \circ r$. We say that some idempotent morphism $f : \mathbf{C}(X, X)$ **splits** if we can find some $Y : \mathbf{C}$ and some $(r, s) : X \lhd Y$ such that $f = s \circ r$. If f does not split, a natural question to ask is whether we can find an embedding $\iota : \mathbf{C} \hookrightarrow \mathbf{D}$ into some category $\mathbf{D}$ such that the idempotent $\iota(f) : \mathbf{D}(\iota(X), \iota(X))$ does split. This is one way to arrive at the *Karoubi envelope*. Its classical universal property is as follows:

▶ **Definition 37** (🚒). *The **Karoubi envelope** of $\mathbf{C}$ is a category $\mathbf{D}$ such that*

1. *Every idempotent in $\mathbf{D}$ splits;*
2. *There is a fully faithful functor $\iota : \mathbf{C} \to \mathbf{D}$;*
3. *For every object $Y : \mathbf{D}$, there merely exists an object $X : \mathbf{C}$ and a retraction $Y \lhd \iota(X)$.*

Classically, this determines the Karoubi envelope up to equivalence of categories. However, as explained in Section 1.3, in univalent foundations, there are two notions of category: *setcategories* and *univalent categories*. We hence add another condition:

4. The setcategory Karoubi envelope (i.e. the Karoubi envelope in setcategory theory) should be a setcategory; the univalent Karoubi envelope should be a univalent category.

In Section 3.1 (resp. 3.2), we construct the setcategory (resp. univalent) Karoubi envelope.

3.1 The Setcategory Construction

In this section, we construct a category $\mathbf{R}^S(\mathbf{C})$ from a category $\mathbf{C}$, and show that it is the setcategory Karoubi envelope if $\mathbf{C}$ is a setcategory. The construction is the same as the one formalized in the 1lab [26] (see also Section 5).

▶ **Definition 38** (🚒). *We define the category $\mathbf{R}^S(\mathbf{C})$. The objects of $\mathbf{R}^S(\mathbf{C})$ are tuples (X, f) with $X : \mathbf{C}$ and $f : \mathbf{C}(X, X)$ such that $f \circ f = f$. The morphisms between (X_1, f_1) and (X_2, f_2) are morphisms $g : \mathbf{C}(X_1, X_2)$ such that $f_2 \circ g \circ f_1 = g$. This can be summarized in the following diagram:*

$$f_1 \circlearrowright X_1 \xrightarrow{g} X_2 \circlearrowright f_2 \tag{5}$$

The identity morphism on (X, f) is given by f and $\mathbf{R}^S(\mathbf{C})$ inherits composition from $\mathbf{C}$.

▶ **Theorem 39** (🚒). *If $\mathbf{C}$ is a setcategory, then $\mathbf{R}^S(C)$ is a setcategory Karoubi envelope.*

Proof.

1. Let $X = (Y, a) : \mathbf{R}^S(\mathbf{C})$ and $f : \mathbf{R}^S(\mathbf{C})(X, X)$ idempotent. Then f splits via $(f, f) : (Y, f) \lhd X$ since $f \circ f = f = \mathsf{id}_{(Y,f)}$.
2. Let $\iota : \mathbf{C} \to \mathbf{R}^S(\mathbf{C})$ be the functor sending $X : \mathbf{C}$ to (X, id_X) and $f : \mathbf{C}(X, Y)$ to f. It is fully faithful:

$$\mathbf{R}^S(\mathbf{C})((X, \mathsf{id}_X), (Y, \mathsf{id}_Y)) = \{f : \mathbf{C}(X, Y) \mid \mathsf{id}_Y \circ f \circ \mathsf{id}_X = f\} \simeq \mathbf{C}(X, Y).$$

3. Let $X = (Y, a) : \mathbf{R}^S(\mathbf{C})$ with $Y : \mathbf{C}$ and $a : Y \to Y$ idempotent. Then, $(a, a) : X \triangleleft \iota(Y)$, since $a \circ a = a = \mathsf{id}_{(Y,a)}$.
4. If $\mathbf{C}$ is a setcategory, the objects in $\mathbf{R}^S(\mathbf{C})$ form a set, since both the type of objects of $\mathbf{C}$ and $\mathbf{C}(X, X)$ are sets, for any $X : \mathbf{C}$. ◀

It is a general fact that completions induce monads:

▶ Remark 40 (🔫). The map $\mathbf{C} \mapsto \mathbf{R}^S(\mathbf{C})$ constitutes a monad on the category of setcategories. The unit is given by the embedding functor $\mathbf{C} \hookrightarrow \mathbf{R}^S(\mathbf{C})$. The multiplication $\mathbf{R}^S(\mathbf{R}^S(\mathbf{C})) \to \mathbf{R}^S(\mathbf{C})$ sends $((X, a), b)$ to (X, b), for $X : \mathbf{C}$, $a : \mathbf{C}(X, X)$ and $b : \mathbf{R}^S(C)((X, a), (X, a))$.

▶ **Proposition 41** (🔫). *If $\mathbf{R}^S(\mathbf{C})$ is univalent, $\mathbf{C}$ is univalent as well.*

Proof. The fully faithful embedding $\iota : \mathbf{C} \hookrightarrow \mathbf{R}^S(\mathbf{C})$ induces equivalences $(X \cong Y) \simeq (\iota(X) \cong \iota(Y))$. We also have an equivalence $(X = Y) \simeq (\iota(X) = \iota(Y))$, because any identity $X = Y$ also preserves the identity morphism on X. Therefore, if $\mathbf{R}^S(\mathbf{C})$ is univalent, we have a chain of equivalences $(X = Y) \simeq (\iota(X) = \iota(Y)) \simeq (\iota(X) \cong \iota(Y)) \simeq (X \cong Y)$, which shows that $\mathbf{C}$ is univalent as well. ◀

To show that the converse of Proposition 41 does not hold, the next example considers the 1-object category $\mathbf{C}_M$ associated to a monoid M. That is, $\mathbf{C}_M$ has one object called, say, $\star$, morphisms $\mathbf{C}_M(\star, \star) = M$, and the composition is given by the monoid multiplication.

▶ **Example 42.** Consider the commutative monoid M consisting of the three matrices $a = \left(\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right)$, $b = \left(\begin{smallmatrix} 1 & 0 \\ 0 & 0 \end{smallmatrix}\right)$ and $c = \left(\begin{smallmatrix} -1 & 0 \\ 0 & 0 \end{smallmatrix}\right)$ under matrix multiplication. Then, $\mathbf{C}_M$ is univalent since a is the only isomorphism. However, $\mathbf{R}^S(\mathbf{C}_M)$ is *not* univalent; indeed, we have $((\star, b) \cong (\star, b)) = \{b, c\} = 2 \neq 1 = ((\star, b) = (\star, b))$.

▶ **Example 43** (🔫). For L a λ-theory, the category $\mathbf{R}$ (Definition 22) is equal to $\mathbf{R}^S(\mathbf{C}_{L_0})$, where L_0 is the monoid $(\{f : L_0 \mid \lambda(\rho(f)) = f\}, \circ, \lambda(x_1))$, consisting of terms with η-equality.

3.2 The Univalent Construction

In this section, we construct a category $\mathbf{R}^U(\mathbf{C})$ from a category $\mathbf{C}$, and show that it is the univalent Karoubi envelope of $\mathbf{C}$, regardless of whether $\mathbf{C}$ is univalent or not. We use the fully faithful Yoneda embedding $よ : \mathbf{C} \hookrightarrow P\mathbf{C}$.

▶ **Definition 44** (🔫). *We define the category $\mathbf{R}^U(\mathbf{C})$ as the full subcategory of $P\mathbf{C}$ consisting of objects $F : P\mathbf{C}$ such that there merely exist an object $X : \mathbf{C}$ and a retraction-section pair $(r, s) : F \triangleleft よ(X)$, summarized in the following diagram:*

$$よ(X_1) \underset{s_1}{\overset{r_1}{\rightleftarrows}} F_1 \xrightarrow{g} F_2 \underset{r_2}{\overset{s_2}{\rightleftarrows}} よ(X_2) \tag{6}$$

▶ **Theorem 45** (🔫). *$\mathbf{R}^U(C)$ is a univalent Karoubi envelope of C.*

Proof.

1. Let $f : \mathbf{R}^U(\mathbf{C})((Y, h_Y), (Y, h_Y))$ be idempotent. Consider the equalizer (E, s) of f and id_Y in $P\mathbf{C}$. Note that $f \circ f = \mathsf{id}_Y \circ f$, so by the universal property of the equalizer, there exists $s : P\mathbf{C}(Y, E)$ such that $f = s \circ r$.

$$\begin{array}{ccccc} E & \xrightarrow{s} & Y & \underset{\mathsf{id}_Y}{\overset{f}{\rightrightarrows}} & Y \\ {\scriptstyle r}\uparrow & \nearrow_{f} & & & \\ Y & & & & \end{array} \tag{7}$$

Note that $s \circ (r \circ s) = f \circ s = s \circ \mathsf{id}_E$, so again by the universal property of the equalizer, $r \circ s = \mathsf{id}_E$, exhibiting E as a retract of Y in $P\mathbf{C}$. Since we can compose retracts, and since (Y, h_Y) is in $\mathbf{R}^U(\mathbf{C})$, we get some h_E such that (E, h_E) is in $\mathbf{R}^U(\mathbf{C})$. Lastly, since the morphisms of $\mathbf{R}^U(\mathbf{C})$ are borrowed directly from $P\mathbf{C}$, we have $f = s \circ r$ and f splits.

2. The Yoneda embedding provides a functor $\iota : \mathbf{C} \to \mathbf{R}^U(\mathbf{C})$, given by

$$\iota(X) = (\text{ﻉ}(X), \|(X, \mathsf{id}_X, \mathsf{id}_X)\|) \quad \text{and} \quad \iota(f) = \text{ﻉ}(f).$$

Since the Yoneda embedding is fully faithful, ι is fully faithful as well.

3. For every $(Y, h_Y) : \mathbf{R}^U(\mathbf{C})$, h_Y witnesses that there merely exist an object $X : \mathbf{C}$ and a retraction $(r, s) : Y \triangleleft \text{ﻉ}(X)$ in $P\mathbf{C}$, and this gives a retraction $(r, s) : (Y, h_Y) \triangleleft \iota(X)$.
4. $\mathbf{R}^U(\mathbf{C})$ is univalent as a full subcategory of a univalent presheaf category. ◀

▶ Remark 46 (🥧). Instead of Item 1 above, one can also show that idempotents split in $\mathbf{R}^U(\mathbf{C})$ by showing that idempotents split in **Set**, and that, under certain conditions (including univalence of the target category), taking a functor category and a full subcategory preserves the splitting of idempotents.

▶ **Theorem 47** (🥧). *$\mathbf{R}^U(\mathbf{C})$ is the Rezk completion of $\mathbf{R}^S(\mathbf{C})$.*

Proof. $\mathbf{R}^U(\mathbf{C})$ is univalent, and there is a fully faithful and essentially surjective functor $\mathbf{R}^S(\mathbf{C}) \to \mathbf{R}^U(\mathbf{C})$, sending $(X, a) : \mathbf{R}^S(\mathbf{C})$ to the equalizer of $\text{ﻉ}(\mathsf{id}_X)$ and $\text{ﻉ}(a)$. ◀

3.3 The Relation Between Scott's and Hyland's Constructions

In this section, we will use the setcategory and univalent Karoubi envelopes to "embed" Scott's reflexive object into Hyland's reflexive object, and show that the correctness of Hyland's construction can be viewed as a consequence of the correctness of Scott's construction.

Recall that Scott and Hyland construct different right inverses to the function $E : \mathsf{RO} \to \mathsf{LamTh}$. Scott constructed the right inverse as $S(L) = (\mathbf{R}, U)$, whereas Hyland constructed $H(L) = (\mathbf{Pshf}_L, L)$. In this section, we show that $\mathbf{R}$ can be embedded into $\mathbf{Pshf}_L$, and that U and L coincide under this embedding.

Fix a λ-theory L. Recall that $\mathbf{R}$ is equal to $\mathbf{R}^S(\mathbf{C}_{L_0})$, where L_0 is the monoid $(L_0, \circ, \lambda(x_1))$; see Example 43. We construct the embedding as a composition of the following functors:

$$\begin{array}{ccccccc} \mathbf{R} & \xrightarrow{=} & \mathbf{R}^S(\mathbf{C}_{L_0}) & \xrightarrow{\quad\sim\quad} & \mathbf{R}^S(\mathbf{C}_{L_1}) & & \\ & & & & \downarrow & & \\ \mathbf{Pshf}_L & \xrightarrow{\sim} & P\mathbf{L} & \xrightarrow{\sim} & P\mathbf{C}_{L_1} & \hookrightarrow & \mathbf{R}^U(\mathbf{C}_{L_1}) \end{array} \tag{8}$$

where L_1 is the monoid $(L_1, \bullet, x_1)$. The following propositions construct two of the functors:

▶ **Proposition 48** (🥧). *The categories $\mathbf{R}^S(\mathbf{C}_{L_0})$ and $\mathbf{R}^S(\mathbf{C}_{L_1})$ are equivalent.*

Proof. The functions $\rho : L_0 \to L_1$ and $\lambda : L_1 \to L_0$ give an isomorphism of monoids $L_0 \cong L_1$, because elements of L_0 (resp. L_1) satisfy η-equality (resp. β-equality). The functoriality of $\mathbf{C}_-$ and $\mathbf{R}^S(-)$ turns this into an equivalence of categories $\mathbf{R}^S(\mathbf{C}_{L_0}) \simeq \mathbf{R}^S(\mathbf{C}_{L_1})$. ◀

▶ **Proposition 49** (🥧). *Let $\mathbf{D}$ be a category with all colimits. Precomposition with the embedding $\mathbf{C}_{L_1} \hookrightarrow \mathbf{L}$ gives an equivalence between functor categories $[\mathbf{L}, \mathbf{D}]$ and $[\mathbf{C}_{L_1}, \mathbf{D}]$.*

Taking $D = \mathbf{Set}^{\mathsf{op}}$ gives an equivalence $P\mathbf{L} \simeq P(\mathbf{C}_{L_1})$, because taking opposite categories preserves equivalences. Together with Proposition 48, Theorem 47, the fact that $\mathbf{R}^U(\mathbf{C}_{L_1})$ is a full subcategory of $P(\mathbf{C}_{L_1})$ and Lemma 34, this gives an embedding of $\mathbf{R}$ into $\mathbf{Pshf}_L$:

▶ **Theorem 50** (). *The composite of the functors is fully faithful. Furthermore, they send the reflexive object* $U : \mathbf{R}$ *to the reflexive object* $L : \mathbf{Pshf}_L$.

The embedding of Theorem 50 witnesses the passage from setcategories (the top row) to univalent categories (the lower row), and the vertical arrow in the diagram is precisely given by the Rezk completion. The theorem allows us to view $\mathbf{R}$ as a full subcategory of $\mathbf{Pshf}_L$, where $U : \mathbf{R}$ becomes $L : \mathbf{Pshf}_L$, which relates Hyland's proof to that of Scott:

Scott's and Hyland's constructions send a λ-theory L to the reflexive objects $S(L) = (\mathbf{R}, U)$ and $H(L) = (\mathbf{Pshf}_L, L)$ respectively. The endomorphism theory of a reflexive object $(\mathbf{C}, X)$ is the λ-theory $E(L)$ with $E(L)_n = \mathbf{C}(X^n, X)$. The result of this construction does not change when we pass to a larger category, as long as the powers of X, the exponential X^X and the morphisms between them do not change. Therefore, one expects an isomorphism $E(H(L)) \cong E(S(L))$, which can be composed with the isomorphism $E(S(L)) \cong L$ from Scott's version of SRT to get $E(H(L)) \cong L$ for Hyland's version of SRT: if Scott's construction gives a section to the retraction E, then Hyland's construction gives a section as well, even though the constructions themselves are quite different.

4 A Tactic for Propagating Substitutions

Now, as mentioned in Remark 15, any λ-theory allows the operations `var`, `app`, `abs` and `subst` with the same interaction as for the pure λ-calculus. Using these, we can define combinators like $a \circ b$ or $\langle a, b \rangle$ for $a, b : L_0$, used in the original proof of SRT. However the equalities about the interaction between `subst` and the other operations are usually not definitional. Consider the following term (using concatenation for application): $\lambda x_5, x_5(x_1x_2(x_4x_3)) \bullet (x_1, x_2, x_3, x_1) : L_3$. It is not hard to see that this results in $\lambda x_4, x_4(x_1x_2(x_1x_3))$. However, it takes several steps to rewrite this: moving the substitution past the λ-abstraction, then into four applications, and lastly using five variable substitutions, resulting in a total of ten rewrites for a seemingly trivial term. In other proofs, 40 or more of these rewrites need to be done, and since there are many such proofs, this quickly becomes tedious. Therefore, we developed a tactic to perform these rewrites. A first version of this tactic was a variation of

```
Ltac reduce_lambda := (rewrite subst_var + ... + rewrite beta_equality).
```

attempting to rewrite with some of the identities. The statement `repeat reduce_lambda` saved a lot of manual work, but it sometimes took a couple of seconds.

Therefore, we developed a new version of the tactic, called `propagate_subst` (), written in `Ltac2` [30]. It recursively traverses the λ-term in the left-hand side of the goal, checking whether the term matches a form that can be rewritten into something else. It performs the possible rewrites, and also prints these rewrite statements which can replace it. For a small example, if the goal is about the interaction between the object $U : \mathbf{R}$ and substitution, `U_term • f = U_term`, a call to `unfold U_term; repeat reduce_lambda` takes ca. 100 ms, but `propagate_subst ()` solves the goal in about 35 ms and prints

```
refine '(subst_abs _ _ _ @ _).
refine '(maponpaths (λ x, (abs x)) (var_subst _ _ _) @ _).
refine '(maponpaths (λ x, (abs x)) (extend_tuple_inr _ _ _) @ _).
```

Replacing the call to `propagate_subst` by these tactics results in the same rewrites, but these only take about five milliseconds. Many equations in `AlgebraicTheories.Combinators` were proved using the tactic. For example, the first 9 lines of `subst_compose` take about 40 ms, whereas `propagate_subst ()` and `reduce_lambda` take about 330 ms and a second respectively.

Apart from the performance gain at the expense of having longer proofs, another reason for replacing calls to `propagate_subst` by the generated statements is the fact that the UniMath community aims to avoid extensive automation in the final proofs, to increase maintainability.

On top of the speedup, this tactic is modular: some of its parts are also tactics themselves, and can be reused for other tactics as well. For example, the `traverse` tactic, which traverses a λ-term in the goal and executes something for every subterm, is also used in another tactic we programmed that is called `generate_refine`, which takes a pattern, and for every subterm that matches it, prints a statement such as

```
refine '(maponpaths (λ x, ... x ...) _ @ _).
```

which can be used to quickly generate statements that very precisely rewrite one subterm.

The `propagate_subst` tactic is also extensible: the patterns for both the subterm traversal and the rewrites are kept in a list, which can be extended when new combinators are defined. For example, at the point where the tactics are defined, the traversal only works for the constructors `var`, `app`, `abs` and `subst`, and the rewrites only work for the interactions between those. Using this, composition $a \circ b$ is defined, and so a pattern to branch into a and b is added to the traversals, and a rewrite with $(a \circ b) \bullet t = (a \bullet t) \circ (b \bullet t)$ is added. Progressing through the file, the same is done for combinators like the pair (a, b), the projection π_1 (including a rewrite $\pi_1\langle a, b\rangle = a$), `curry` and `n_tuple` (consisting of nested pairs).

We generalized `propagate_subst` to a tactic for simplification with arbitrary sets of identities (), and applied this to formulas in a hyperdoctrine (), a formalization related to [41].

5 Related Work

We first discuss related work in the area of **formalization of denotational semantics** of lambda calculi, with particular focus on fixpoints.

Perhaps most relevant to our work, Benton, Kennedy, and Varming [13] formalize, in Rocq, a constructive notion of ω-cpos and the construction, via limits, of solutions to recursive domain equations in the category of ω-cpos. This involves constructing fixpoints on the level of types. Similarly, Dockins [17] formalizes effective domain theory in Rocq. Those works are thus complementary to ours: the authors work exclusively in the category of certain domains (e.g., ω-cpos), whereas we study general reflexive objects in suitable categories: their specific ω-cpos could be fed into our *endomorphism theory* function to obtain specific lambda calculi.

De Jong [15] models PCF in domains, formalizing the Scott model in particular. In [16], De Jong and Escardó construct Scott's D_∞ in univalent foundations, in Agda.

Møgelberg [28] develops a language with fixed point combinators as a metalanguage to reason about domains. In another line of work, *guarded* type theory (see, e.g., [14, 29]) is being developed as a meta-language for reasoning about recursive domain equations.

Less closely related to our work on the technical level are other efforts formalizing the syntax and semantics of programming languages specifically in UniMath. In particular, Ahrens, Lumsdaine, and Voevodsky [7] compare, in UniMath, different notions of model for dependent type theories. Van der Weide [39] constructs an equivalence of bicategories between univalent locally cartesian closed categories and univalent categories with suitable type formers, and various extensions of that equivalence to additional structure.

Finally, Altenkirch, Kaposi, Šinkarovs, and Végh [10] construct, in Cubical Agda, an equivalence between the simply-typed lambda calculus and combinatory logic. Both theories are given as Generalized Algebraic Theories.

We mention some work on formalization of **initial algebra semantics** for languages including the untyped lambda calculus. Hirschowitz and Maggesi [20, Thm. 3] formalize the untyped lambda calculus with β- and η-equality. Ahrens [1] proves an initial semantics result

including, as a guiding instance, the lambda calculus with β- and η-*reduction* (as opposed to β- and η-*equality*), formalized in Rocq. Ahrens, Hirschowitz, Lafont, and Maggesi [4, 3] prove initial semantics for abstract syntax with equations, and formalize their results in UniMath.

Next, we discuss related work in the area of **formalization of (univalent) category theory**. The formalization of univalent category theory started with [5], and was continued in many different works, e.g., [6, 8, 2, 40], by various authors. We are basing our formalization on the library of univalent category theory built to accompany these formalizations.

There are several other libraries on univalent category theory. The HoTT library [12] focuses on "wild" (higher) categories, that is, categories that omit truncation levels for types of cells. For our purpose, it is important to work with "true" categories, that is, things that correspond to categories in Voevodsky's simplicial set model of univalent foundations [24]. There does not seem to be a formalization of the Karoubi envelope in the HoTT library. The 1lab [26] is another library of univalent category theory. It features a formalization of the Karoubi envelope of a *pre*category at `https://1lab.dev/Cat.Instances.Karoubi.html`. Here, a "precategory" is essentially a category without the assumption that it is either a setcategory or a univalent category. The construction implemented there is the one described in Section 3.1; as discussed in Example 42, it does not necessarily yield a univalent category.

There are many other libraries of formalized category theory; for space reasons, we only point to papers listing and comparing some of them. Gross, Chlipala, and Spivak [19] provide a comparison of different libraries that, at publication time, was quite comprehensive. Hu and Carette [21] provide another comparison of different libraries.

Finally, Shulman [34] studies the question whether idempotent functions between types in MLTT split, with a formalization in Rocq. Semantically, that work is concerned with morphisms between ∞-groupoids, not with morphisms in a 1-category as in our setting.

6 Conclusion

We have presented a formalization of SRT in univalent foundations. We see two benefits to working in univalent foundations:

1. We can express on the level of *types* a meaningful relation between λ-theories and reflexive objects, by stating that the functions of Diagram (2) form a section-retraction pair. Proving this is possible because, starting with a given λ-theory, going to reflexive objects and back to λ-theories yields an isomorphic – and hence identical – λ-theory. This uses crucially that the category of λ-theories is univalent, that is, that isomorphism of λ-theories is the same as their identities – a consequence of the univalence axiom.
2. The differences between Scott's and Hyland's constructions are clarified in univalent foundations: Scott's is adequate for setcategories, Hyland's for univalent categories.

We have furthermore programmed a tactic to manipulate lambda terms, combining the convenience of automation with the good performance of precise rewriting.

There are some questions we have left open. In particular, one could ask how Diagram (2) lifts to the level of *categories*. In that case, the left-hand side clearly forms a univalent 1-category (Proposition 13). The status of the right-hand side is less clear: it could form a (univalent) 1-category whose objects are certain setcategories with an object therein, or a bicategory whose objects are certain univalent categories.

References

1 Benedikt Ahrens. Modules over relative monads for syntax and semantics. *Math. Struct. Comput. Sci.*, 26(1):3–37, 2016. doi:10.1017/S0960129514000103.

2 Benedikt Ahrens, Dan Frumin, Marco Maggesi, Niccolò Veltri, and Niels van der Weide. Bicategories in univalent foundations. *Math. Struct. Comput. Sci.*, 31(10):1232–1269, 2021. doi:10.1017/S0960129522000032.

3 Benedikt Ahrens, André Hirschowitz, Ambroise Lafont, and Marco Maggesi. High-level signatures and initial semantics. In Dan R. Ghica and Achim Jung, editors, *27th EACSL Annual Conference on Computer Science Logic, CSL 2018, September 4-7, 2018, Birmingham, UK*, volume 119 of *LIPIcs*, pages 4:1–4:22. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.CSL.2018.4.

4 Benedikt Ahrens, André Hirschowitz, Ambroise Lafont, and Marco Maggesi. Modular specification of monads through higher-order presentations. In Herman Geuvers, editor, *4th International Conference on Formal Structures for Computation and Deduction, FSCD 2019, June 24-30, 2019, Dortmund, Germany*, volume 131 of *LIPIcs*, pages 6:1–6:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.FSCD.2019.6.

5 Benedikt Ahrens, Krzysztof Kapulkin, and Michael Shulman. Univalent categories and the Rezk completion. *Mathematical Structures in Computer Science*, 25(5):1010–1039, January 2015. doi:10.1017/s0960129514000486.

6 Benedikt Ahrens and Peter LeFanu Lumsdaine. Displayed categories. *Log. Methods Comput. Sci.*, 15(1), 2019. doi:10.23638/LMCS-15(1:20)2019.

7 Benedikt Ahrens, Peter LeFanu Lumsdaine, and Vladimir Voevodsky. Categorical structures for type theory in univalent foundations. *Log. Methods Comput. Sci.*, 14(3), 2018. doi:10.23638/LMCS-14(3:18)2018.

8 Benedikt Ahrens, Ralph Matthes, Niels van der Weide, and Kobe Wullaert. Displayed monoidal categories for the semantics of linear logic. In Amin Timany, Dmitriy Traytel, Brigitte Pientka, and Sandrine Blazy, editors, *Proceedings of the 13th ACM SIGPLAN International Conference on Certified Programs and Proofs, CPP 2024, London, UK, January 15-16, 2024*, pages 260–273. ACM, 2024. doi:10.1145/3636501.3636956.

9 Benedikt Ahrens and Paige Randall North. Univalent foundations and the equivalence principle. In Stefania Centrone, Deborah Kant, and Deniz Sarikaya, editors, *Reflections on the Foundations of Mathematics: Univalent Foundations, Set Theory and General Thoughts*, pages 137–150. Springer International Publishing, Cham, 2019. doi:10.1007/978-3-030-15655-8_6.

10 Thorsten Altenkirch, Ambrus Kaposi, Artjoms Šinkarovs, and Tamás Végh. Combinatory Logic and Lambda Calculus Are Equal, Algebraically. In Marco Gaboardi and Femke van Raamsdonk, editors, *8th International Conference on Formal Structures for Computation and Deduction (FSCD 2023)*, volume 260 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 24:1–24:19, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.FSCD.2023.24.

11 Hendrik Pieter Barendregt. *The lambda calculus - its syntax and semantics*, volume 103 of *Studies in logic and the foundations of mathematics*. North-Holland, 1985.

12 Andrej Bauer, Jason Gross, Peter LeFanu Lumsdaine, Michael Shulman, Matthieu Sozeau, and Bas Spitters. The HoTT library: a formalization of homotopy type theory in Coq. In Yves Bertot and Viktor Vafeiadis, editors, *Proceedings of the 6th ACM SIGPLAN Conference on Certified Programs and Proofs, CPP 2017, Paris, France, January 16-17, 2017*, pages 164–172. ACM, 2017. doi:10.1145/3018610.3018615.

13 Nick Benton, Andrew Kennedy, and Carsten Varming. Some Domain Theory and Denotational Semantics in Coq. In Stefan Berghofer, Tobias Nipkow, Christian Urban, and Makarius Wenzel, editors, *Theorem Proving in Higher Order Logics*, pages 115–130, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg. doi:10.1007/978-3-642-03359-9_10.

14 Lars Birkedal, Rasmus Ejlers Møgelberg, Jan Schwinghammer, and Kristian Støvring. First steps in synthetic guarded domain theory: step-indexing in the topos of trees. *Log. Methods Comput. Sci.*, 8(4), 2012. doi:10.2168/LMCS-8(4:1)2012.

15 Tom de Jong. The Scott model of PCF in univalent type theory. *Math. Struct. Comput. Sci.*, 31(10):1270–1300, 2021. doi:10.1017/S0960129521000153.

16 Tom de Jong and Martín Hötzel Escardó. Domain theory in constructive and predicative univalent foundations. In Christel Baier and Jean Goubault-Larrecq, editors, *29th EACSL Annual Conference on Computer Science Logic, CSL 2021, January 25-28, 2021, Ljubljana, Slovenia (Virtual Conference)*, volume 183 of *LIPIcs*, pages 28:1–28:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICS.CSL.2021.28.

17 Robert Dockins. Formalized, effective domain theory in coq. In Gerwin Klein and Ruben Gamboa, editors, *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*, volume 8558 of *Lecture Notes in Computer Science*, pages 209–225. Springer, 2014. doi:10.1007/978-3-319-08970-6_14.

18 Daniel R. Grayson. An introduction to univalent foundations for mathematicians. *Bulletin of the American Mathematical Society*, 55(4):427–450, 2018. doi:10.1090/bull/1616.

19 Jason Gross, Adam Chlipala, and David I. Spivak. Experience Implementing a Performant Category-Theory Library in Coq. In Gerwin Klein and Ruben Gamboa, editors, *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*, volume 8558 of *Lecture Notes in Computer Science*, pages 275–291. Springer, 2014. doi:10.1007/978-3-319-08970-6_18.

20 André Hirschowitz and Marco Maggesi. Modules over monads and initial semantics. *Information and Computation*, 208(5):545–564, 2010. Special Issue: 14th Workshop on Logic, Language, Information and Computation (WoLLIC 2007). doi:10.1016/j.ic.2009.07.003.

21 Jason Z. S. Hu and Jacques Carette. Formalizing category theory in Agda. In Catalin Hritcu and Andrei Popescu, editors, *CPP '21: 10th ACM SIGPLAN International Conference on Certified Programs and Proofs, Virtual Event, Denmark, January 17-19, 2021*, pages 327–342. ACM, 2021. doi:10.1145/3437992.3439922.

22 J. Martin E. Hyland. Towards a notion of lambda monoid. In John Power and Cai Wingfield, editors, *Proceedings of the Workshop on Algebra, Coalgebra and Topology, WACT 2013, Bath, UK, March 1, 2013*, volume 303 of *Electronic Notes in Theoretical Computer Science*, pages 59–77. Elsevier, 2013. doi:10.1016/J.ENTCS.2014.02.004.

23 J. Martin E. Hyland. Classical lambda calculus in modern dress. *Mathematical Structures in Computer Science*, 27(5):762–781, 2017. doi:10.1017/S0960129515000377.

24 Krzysztof Kapulkin and Peter LeFanu Lumsdaine. The simplicial model of Univalent Foundations (after Voevodsky). *Journal of the European Mathematical Society*, 23(6):2071–2126, 2021. doi:10.4171/jems/1050.

25 J. Lambek and P. J. Scott. *Introduction to higher order categorical logic*, volume 7 of *Cambridge Studies in Advanced Mathematics*. Cambridge University Press, Cambridge, 1986.

26 Amélia Liao, Jonathan Coates, Reed Mullanix, et al. 1lab. URL: https://1lab.dev/.

27 Per Martin-Löf. *An intuitionistic theory of types*. Bibliopolis, 1984.

28 Rasmus Ejlers Møgelberg. Interpreting polymorphic FPC into domain theoretic models of parametric polymorphism. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *Automata, Languages and Programming, 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part II*, volume 4052 of *Lecture Notes in Computer Science*, pages 372–383. Springer, 2006. doi:10.1007/11787006_32.

29 Rasmus Ejlers Møgelberg and Marco Paviotti. Denotational semantics of recursive types in synthetic guarded domain theory. *Math. Struct. Comput. Sci.*, 29(3):465–510, 2019. doi:10.1017/S0960129518000087.

30 Pierre-Marie Pédrot. Ltac2: Tactical warfare, 2019. Talk at CoqPL 2019. URL: https://popl19.sigplan.org/details/CoqPL-2019/8/Ltac2-Tactical-Warfare.

31 Dana S. Scott. Continuous lattices. In F. W. Lawvere, editor, *Toposes, Algebraic Geometry and Logic*, pages 97–136, Berlin, Heidelberg, 1972. Springer Berlin Heidelberg. doi:10.1007/BFb0073967.

32 Dana S. Scott. Relating theories of the λ-calculus. In J. P. Seldin and J. R. Hindley, editors, *To H. B. Curry: Essays on Combinatory Logic and Formalism*, pages 403–450. 1980.

33 Dana S. Scott. A type-theoretical alternative to ISWIM, CUCH, OWHY. *Theoretical Computer Science*, 121(1):411–440, 1993. doi:10.1016/0304-3975(93)90095-B.

34 Michael Shulman. Idempotents in intensional type theory. *Log. Methods Comput. Sci.*, 12(3), 2016. doi:10.2168/LMCS-12(3:9)2016.

35 The Coq Development Team. The Coq reference manual – release 8.19.0. https://coq.inria.fr/doc/V8.19.0/refman, 2024.

36 S. N. Tronin. Abstract clones and operads. *Siberian Mathematical Journal*, 43:746–755, 2002. doi:10.1023/A:1016340806503.

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

38 Arnoud van der Leer. Universal algebra, univalent foundations and the untyped λ-calculus. Master's thesis, Delft University of Technology, December 2024. URL: https://repository.tudelft.nl/record/uuid:e6582866-9c0d-4a13-8eda-42c25e0deba4.

39 Niels van der Weide. The internal languages of univalent categories. *CoRR*, abs/2411.06636, 2024. doi:10.48550/arXiv.2411.06636.

40 Niels van der Weide, Nima Rasekh, Benedikt Ahrens, and Paige Randall North. Univalent double categories. In Amin Timany, Dmitriy Traytel, Brigitte Pientka, and Sandrine Blazy, editors, *Proceedings of the 13th ACM SIGPLAN International Conference on Certified Programs and Proofs, CPP 2024, London, UK, January 15-16, 2024*, pages 246–259. ACM, 2024. doi:10.1145/3636501.3636955.

41 Niels van der Weide and Kobe Wullaert. Rezk completions for (elementary) topoi. https://msp.cis.strath.ac.uk/types2025/abstracts/TYPES2025_paper71.pdf, 2025. Abstract presented at TYPES 2025.

42 Vladimir Voevodsky, Benedikt Ahrens, Daniel R. Grayson, et al. Unimath — a computer-checked library of univalent mathematics. available at http://unimath.org. doi:10.5281/zenodo.13828995.