# Primitive Recursive Dependent Type Theory

Johannes Schipp von Branitz
Ulrik Buchholtz

December 26, 2023

**Abstract**

...

## Contents

# 1 Introduction

In this section we discuss how Martin-Löf Type Theory can be restricted such that all definable terms $f : \mathrm{N} \to \mathrm{N}$ define a primitive recursive function.

We prove that the straightforward way to achieve this is restricting the induction principle of the natural numbers to all types which do not contain dependent product types or other nontrivial inductive types. The proof proceeds by gluing the set-model to a certain presheaf category of primitive recursive functions.

The resulting system has the downside that while it is complete with respect to primitive recursive functions, not every primitive recursive function can be defined in a straightforward way. We present examples and potential solutions.

Martin Hofmann defined calculi which are sound and complete with respect to polytime functions (**??**). We expect that these calculi admit conservative extensions to a dependent type-theoretical setting. Such an extension would provide a more convenient formal language for defining and reasoning about polytime functions.

We find that it is a non-trivial exercise to add a comonadic modality do dependent type theory, needed to define safe recursion, to MLTT. Indeed, the modality $\Box$ used in [Hof97; Hof98] is not a general type former; types $\Box A$ are only allowed in the domain of function types and $\Box A \to B$ is considered a subtype of $A \to B$. This restriction is placed in order to avoid using let $\cdots$ in-style introduction and elimination rules.

We decide to first consider a simpler case, namely generalising primitive recursive functions to a dependent type theory.

**Definition 1.1.** The *basic primitive recursive functions* are constant functions, the successor function and projections out of finite products of $\mathbb{N}$. A *primitive recursive function* is obtained by finite applications of function composition and the primitive recursion operator

$$(l : \mathbb{N}) \to (\mathbb{N}^l \to \mathbb{N}) \to (\mathbb{N}^{l+2} \to \mathbb{N}) \to (\mathbb{N}^{l+1} \to \mathbb{N})$$
$$\mathrm{primrec}^l_{g,h}(0, x) = g(x)$$
$$\mathrm{primrec}^l_{g,h}(n+1, x) = h(n, \mathrm{primrec}^l_{g,h}(n, x), x)$$

Many functions are primitive recursive. Some examples include addition, exponentiation and the greatest common divisor.

One of the simplest and earliest-discovered examples of a total computable function which is not primitive recursive is the Ackermann function (c.f. [Ack28]). A modern two-argument variant A is given by

$$\mathrm{A}(0, n) := n + 1$$
$$\mathrm{A}(m+1, 0) := \mathrm{A}(m, 1)$$
$$\mathrm{A}(m+1, n+1) := \mathrm{A}(m, \mathrm{A}(m+1, n))$$

for non-negative integers $m$ and $n$.

This definition can be understood in ordinary dependent type theory as the definition of a term $\mathrm{A} : \mathrm{N} \to \mathrm{N} \to \mathrm{N}$ by pattern matching. One can show that $A$ grows faster than any primitive recursive function and is therefore not primitive recursive. The explosion in growth is caused by the definition

$$\mathrm{A}(m+1, n+1) := \mathrm{A}(m, \mathrm{A}(m+1, n)).$$

What would a dependent type theory which is sound and complete w.r.t. primitive recursion look like? In the formal definition of the Ackermann function one has to use the induction principle of N for the constant type family

$$(n : \mathrm{N}) \to (\mathrm{N} \to \mathrm{N}).$$

One one hand, induction on natural numbers and products suffice to represent all primitive recursive functions.

**Theorem 1.2** (Theorem 4.6 in [HS20])**.** The primitive recursive functions are exactly the representable functions in the free cartesian category with parametrized NNO.

On the other hand, if the category has exponentials, more than just the primitive recursive functions are representable.

**Theorem 1.3** (Theorem 4.8 in [HS20])**.** The provably total functions of Peano Arithmetic are exactly the representable functions in the free cartesian closed category with weak NNO.

In other words, adding Π-types to the recursion principle of the NNO in a category gets us beyond primitive recursion.

Therefore, one might hope that if elimination into $\mathrm{N} \to \mathrm{N}$ is not allowed, all definable functions remain primitive recursive. We partially prove that this is in fact the case.

## 2 Abstract Syntax of the Basic Theory

We would like to restrict the induction principle of an inductive natural numbers type without entirely removing dependent products from the theory. To do so, we introduce a universe containing the empty type, the unit type, and which is closed under dependent sum, and identity types, but not dependent products. We denote the theory $\mathrm{T}_{\mathrm{pr}}$ (Primitive Recursive Dependent Type Theory). We give a formal account of that theory by specifying its abstract syntax using the logical framework.

$$
\begin{aligned}
&\text{record } \mathrm{T_{pr}} : \mathrm{SIG} \text{ where}\\
&\quad \mathrm{tp}_\alpha : \square\\
&\quad \mathrm{tm}_\alpha : \mathrm{tp}_\alpha \to \square\\
&\quad \langle\uparrow^\beta_\alpha\rangle : \mathrm{tp}_\alpha \to \mathrm{tp}_\beta\\
&\qquad \_ : \{A\}\langle\uparrow^\alpha_\alpha\rangle A =_{\mathrm{tp}_\alpha} A\\
&\qquad \_ : \{A\}\langle\uparrow^\gamma_\beta\rangle\langle\uparrow^\beta_\alpha\rangle A =_{\mathrm{tp}_\gamma} \langle\uparrow^\gamma_\alpha\rangle A\\
&\qquad \_ : \{A\}\mathrm{tm}_\alpha(A) =_\square \mathrm{tm}_\beta(\langle\uparrow^\beta_\alpha\rangle A)\\
&\qquad \_ : \{A,B\}\langle\uparrow^\beta_\alpha\rangle A =_{\mathrm{tp}_\beta} \langle\uparrow^\beta_\alpha\rangle B \to A =_{\mathrm{tp}_\alpha} B\\
&\quad \Sigma_\alpha : (A : \mathrm{tp}_\alpha) \times (B : \mathrm{tm}_\alpha(A) \to \mathrm{tp}_\alpha) \to \mathrm{tp}_\alpha\\
&\quad \mathrm{pair}_\alpha : (A : \mathrm{tp}_\alpha) \times (B : \mathrm{tm}_\alpha(A) \to \mathrm{tp}_\alpha) \to ((x : \mathrm{tm}_\alpha) \times \mathrm{tm}_\alpha(B(x))) \cong_{\mathrm{tp}_\alpha} \mathrm{tm}_\alpha(\Sigma_\alpha)\\
&\qquad \_ : \{A,B\}\langle\uparrow^\beta_\alpha\rangle\Sigma_\alpha(A,B) =_{\mathrm{tp}_\beta} \Sigma_\beta(\langle\uparrow^\beta_\alpha\rangle A, \langle\uparrow^\beta_\alpha\rangle \circ B)\\
&\quad \mathrm{eq}_\alpha : (A : \mathrm{tp}_\alpha) \to \mathrm{tp}_\alpha\\
&\quad \mathrm{refl}_\alpha : (A : \mathrm{tp}_\alpha) \to (x, y : \mathrm{tm}_\alpha(A)) \to (x =_{\mathrm{tm}_\alpha(A)} y) \cong \mathrm{tm}_\alpha(\mathrm{eq}_\alpha(A))\\
&\qquad \_ : (A : \mathrm{tp}_\alpha) \to \langle\uparrow^\beta_\alpha\rangle\mathrm{eq}_\alpha(A) =_{\mathrm{tp}_\beta} \mathrm{eq}_\beta(\langle\uparrow^\beta_\alpha\rangle A)\\
&\quad \Pi : (A : \mathrm{tp}_1) \times (B : \mathrm{tm}_1(A) \to \mathrm{tp}_1) \to \mathrm{tp}_1\\
&\quad \lambda : (A : \mathrm{tp}_1) \times (B : \mathrm{tm}_1(A) \to \mathrm{tp}_1) \to (x : \mathrm{tm}_1(A) \to \mathrm{tm}_1(B(x))) \cong \mathrm{tm}_1(\Pi(A,B))\\
&\quad \mathrm{N}_0 : \mathrm{tp}_0\\
&\quad \mathrm{zero}_0 : \mathrm{tm}_0(\mathrm{N}_0)\\
&\quad \mathrm{succ}_0 : \mathrm{tm}_0(\mathrm{N}_0) \to \mathrm{tm}_0(\mathrm{N}_0)\\
&\quad \mathrm{primind} : (C : \mathrm{tm}_0(\mathrm{N}_0) \to \mathrm{tp}_0)\\
&\qquad\quad \to (c_Z : \mathrm{tm}_0(C(\mathrm{zero}_0)))\\
&\qquad\quad \to (c_S : (n : \mathrm{tm}_0(\mathrm{N}_0)) \to (\mathrm{tm}(C(n))) \to C(\mathrm{succ}_0(n)))\\
&\qquad\quad \to (f : (x : \mathrm{tm}_0(\mathrm{N}_0)) \to \mathrm{tm}_0(C(x)))\\
&\qquad\qquad \times (f(\mathrm{zero}_0) =_{\mathrm{tm}_0(C(\mathrm{zero}_0))} c_Z)\\
&\qquad\qquad \times ((n : \mathrm{tm}_0(\mathrm{N}_0)) \to (f(\mathrm{succ}_0 n) =_{\mathrm{tm}_0(C(\mathrm{succ}_0 n))} c_S(fn)))\\
&\quad \mathrm{U}_0 : \mathrm{tp}_1\\
&\qquad \_ : \mathrm{tm}_1(\mathrm{U}_0) =_\square \mathrm{tp}_0
\end{aligned}
$$

Here $\alpha, \beta$ and $\gamma$ range over the two-element linear order $\{0 < 1\}$ and $\alpha \leq \beta \leq \gamma$. Curly braces such as $\{A\}$ denote implicitly universally quantified arguments. Note that we add dependent sums at both type levels together with coherence conditions for $\uparrow$. On the other hand, it suffices to add natural numbers at the first level. We abuse notation in that $\uparrow := \uparrow^1_0$. For other $\alpha$ and $\beta$, $\uparrow^\beta_\alpha$ is the identity function anyway. We may further write $\mathrm{N}_1 :=\uparrow \mathrm{N}_0$ and similarly for the other constructors. We may also write $\mathrm{N} := \mathrm{N}_0$.

**Theorem 2.1.** The theory $\mathrm{T_{pr}}$ is complete with respect to primitive recursion

in the sense that any primitive recursive function can be defined in it.

*Proof.* The basic primitive recursive functions are clearly definable using primind. Function composition is given by substitution. Note that the complications of substitution are absorbed by the logical framework. The term primind also covers the recursor primrec$^l$. The extra variables are absorbed by the context. ■

# 3 Semantics in Sets

In this section we define two interpretations $[\![-]\!]_{\mathrm{Set}}$ and $[\![-]\!]_{\mathrm{PrR}}$ in Set and in a certain presheaf category PrR of primitive recursive functions.

The theory $\mathrm{T}_{\mathrm{pr}}$ is a direct restriction of MLTT. As such, any model for MLTT is a model for $\mathrm{T}_{\mathrm{pr}}$. This includes any presheaf topos on a small category.

# 4 The Structure of Presheaves on a Category of Primitive Recursive Functions

### The Category $R$ of Primitive Recursive Functions

**Definition 4.1.** We define the category R of arities and primitive recursive functions which has objects the natural numbers and morphisms $\mathrm{R}(m, m')$ are given by the primitive recursive functions of type $\mathbb{N}^m \to \mathbb{N}^{m'}$.

We sometimes write

$$\mathbf{1} := 0$$

for the arity 0, because it denotes the terminal object, and

$$\mathbf{N}^m := m$$

for the arity $m$.

**Lemma 4.2.** The category R has finite products. The product of $m$ and $n$ is given by $m + n$.

**Lemma 4.3.** The category R does not have all finite limits.

*Proof.* There is no $k : \mathbb{N}$ and matching morphisms such that the square

$$\begin{array}{ccc} \mathbb{N}^k & \longrightarrow & \mathbb{N} \\ \downarrow & & \downarrow{\scriptstyle \lambda\_.0} \\ \mathbb{N} & \xrightarrow{\lambda\_.1} & \mathbb{N} \end{array}$$

commutes, so the cospan has no pullback. ■

**Lemma 4.4.** The category R is equivalent to the category R′ with just two objects 0 and 1, and morphisms $\mathrm{R}'(m, n) := \mathrm{R}(m, n)$.

*Proof.* Choose your favourite primitive recursive bijection[1] $\mathbb{N} \to \mathbb{N} \times \mathbb{N}$. ■

[1] Inverses of primitive recursive functions are primitive recursive. citation

**Presheaves on** $\mathrm{R}$

**Lemma 4.5.** For $n, m \in \mathbb{N}$ we have that

$$\mathrm{y}_n \times \mathrm{y}_m \cong \mathrm{y}_{n+m}.$$

*Proof.* The natural isomorphism can be constructed directly; at each component $x$ it combines $n$ primitive recursive functions with domain $\mathbb{N}^x$ and $m$ such functions to $n+m$ such functions. Alternatively, this lemma also follows from Lemma 4.2 and the fact that the Yoneda embedding preserves finite limits. ■

**Notation for Universes in** $\mathrm{PrR}$

We assume a hierarchy of two strong cumulative universes $\mathcal{U}_0 < \mathcal{U}_1$ in PrR and work in its internal language.

**Definition 4.6.** We have an object

$$\begin{aligned}\mathcal{U}_0^{\mathrm{pr}} := (X : \mathcal{U}_0) \times ((g : X) \\ \to (h : \mathrm{y}_1 \to X \to X) \\ \to ((f : \mathrm{y}_1 \to X) \\ \times (\_ : f(\mathrm{zero}) = g) \\ \times (\_ : (n : \mathrm{y}_1) \to f(\mathrm{succ} n) = h(n, fn))))\end{aligned}$$

of $\mathcal{U}_0$-small types which $\mathrm{N}_0$ can eliminate into. A variant of this captures $\mathrm{y}_1$ as a parametrised natural numbers algebra by introducing an arbitrary context $\Gamma$.

$$\begin{aligned}\mathcal{U}_0^{\mathrm{cpr}} := (X : \mathcal{U}_0) \times ((\Gamma : \mathcal{U}_0) \\ \to (g : \Gamma \to X) \\ \to (h : \Gamma \to \mathrm{y}_1 \to X \to X) \\ \to ((f : \Gamma \to \mathrm{y}_1 \to X) \\ \times (\_ : (\gamma : \Gamma) \to f^\gamma(\mathrm{zero}) = g^\gamma) \\ \times (\_ : (\gamma : \Gamma) \to (n : \mathrm{y}_1) \to f^\gamma(\mathrm{succ} n) = h^\gamma(n, f^\gamma(n)))))\end{aligned}$$

We also have the object

$$\hat{\mathcal{U}}_0^{\mathrm{pr}} := (X : \mathcal{U}_0) \times (r : \mathrm{y}_1 \to (X + \mathbf{1})) \times (s : (X + \mathbf{1}) \to \mathrm{y}_1) \times (r \circ s = \mathrm{id}_{X+\mathbf{1}})$$

of types $X$ such that the coproduct $X + \mathbf{1}$ is a retract of $\mathrm{y}_1$. The section $s$ allows us to encode elements of $X$ as elements of $\mathrm{y}_1$.

## Decidable Subsets of Natural Numbers

This section contains useful lemmas for the next section.

**Definition 4.7.** A subset $S \subset \mathbb{N}^n$ is called *decidable* if there is a primitive recursive function $\chi_S : \mathbb{N}^n \to \mathbb{N}$ such that $\chi_S(x) = 1 \Leftrightarrow x \in S$ and $\chi_S(x) = 0 \Leftrightarrow x \notin S$.

**Lemma 4.8.** For primitive recursive $f : \mathbb{N}^n \to \mathbb{N}$ and $g : \mathbb{N}^m \to \mathbb{N}$, the set

$$\{f = g\} := \{(x, y) \in \mathbb{N}^n \times \mathbb{N}^m \mid fx = gy\} \subset \mathbb{N}^{n+m}$$

is decidable.

*Proof.* We can decide $\{f = g\}$ with the p.r. function

$$\chi_{\{f=g\}}(x, y) = 1 - \mid \max(fx, gy) - \min(fx, gy) \mid,$$

where $\mid 0 \mid = 0$ and $\mid n + 1 \mid = 1$. ■

### The Finite Cover Sheaf Topos on R′, respectively

Let us recall the definition of a basis for a Grothendieck topology on a category which is not required to have pullbacks.

**Definition 4.9.** A *basis* on a category $\mathbf{C}$ is given by a function $K : \mathrm{Ob}\mathbf{C} \to \mathrm{ObSet}$ where $K(C)$ is a family of morphisms with codomain $C$ such that all of the following three conditions are satisfied.

1. If $f : C' \to C$ is an isomorphism, then $\{f\} \in K(C)$.
2. If $\{f_i\} \in K(C)$, then for any $g : D \to C$ there exists a basic cover $\{h_j\} \in K(D)$ such that for each $j$, the morphism $g \circ h_j$ factors through some $f_i$.
3. If $\{f_i : C_i \to C\} \in K(C)$ and for each $i$ we have $\{g_{ij} : D_{ij} \to C_i\} \in K(C)$, then the family of composites $\{f_i \circ g_{ij} \mid i, j\}$ is in $K(C)$.

**Definition 4.10.** Let $J_{\mathrm{fin}}$ be the Grothendieck topology on R′ (resp. R) generated by the basis $K$ consisting of finite jointly surjective families. This means that a family of morphisms $\{f_i : \mathbf{N}^{n_i} \to \mathbb{N}\}$ with $n_i \in \{0, 1\}$ (resp. $n_i\mathbb{N}$) is basic iff it is finite and the induced map out of the coproduct in Set is surjective.

**Lemma 4.11.** $K$ is in fact a basis on $R$ (resp. $R'$).

*Proof.* Conditions 1 and 3 are clear. There are various ways for constructing a cover $\{h_j\}$ as required in 2, make explicit. ■

We can give an even simpler (more refined) basis $K'$ for $J_{\mathrm{fin}}$. mb remove this

**Lemma 4.12.** $J_{\mathrm{fin}}$ is generated by the basis $K'$ consisting of finite, jointly surjective, disjoint families. This means that in addition to the requirements for $K$, we also require that for any two elements $f_i : \mathbb{N}^{n_i} \to \mathbb{N}$ and $f_j : \mathbb{N}^{n_j} \to \mathbb{N}$ of each finite jointly surjective family of morphisms the pullback of the cospan

$$f_i : \mathbb{N}^{n_i} \to \mathbb{N} \leftarrow \mathbb{N}^{n_j} : f_j$$

in Set is empty.

*Proof.* 1 and 3 are clear again. Regarding 2, for now we only consider the case for $f_1, ..., f_n \in K'(1)$ where the domain of each $f_i$ is $\mathbb{N}^1$, and for each $g : \mathrm{R}(1,1)$ we get a partition

$$\mathbb{N}^2 = \{g = f_1\} + \cdots + \{g = f_n\}.$$

We may omit empty cofactors in this composition. Then the lifts

$$\xi_{\{g=f_i\}}\mathbb{N}^2 \to \mathbb{N}$$

precomposed with our fixed choice of isomorphism $\mathbb{N} \to \mathbb{N}^2$ satisfy the requirements. ■

**Definition 4.13.** We denote the sheaf topos as follows:

$$\mathcal{R} := \mathrm{Sh}(\mathrm{R}, J_{\mathrm{fin}})$$

and

$$\mathcal{R}' := \mathrm{Sh}(\mathrm{R}', J_{\mathrm{fin}}).$$

We assume strong cumulative universes $\mathcal{V}_0 < \mathcal{V}_1$ in $\mathcal{R}$ and denote the universes $\mathcal{U}_0^{\mathrm{pr}}, \hat{\mathcal{U}}_0^{\mathrm{pr}}, \mathcal{U}_0^{\mathrm{cpr}} : \mathrm{PrR}$ in the context of $\mathcal{R}$ by $\mathcal{V}_0^{\mathrm{pr}}, \hat{\mathcal{V}}_0^{\mathrm{pr}}, \mathcal{V}_0^{\mathrm{cpr}}$, respectively.

The next lemma justifies the previous definition, because it implies that $\mathcal{R}(\mathbf{a}\mathrm{y}_1, \mathbf{a}\mathrm{y}_1) \cong \mathrm{R}(1,1)$.

**Lemma 4.14.** $J_{\mathrm{fin}}$ is subcanonical, i.e. all representables are sheaves.

### $\mathrm{y}_1$-elimination into $\mathrm{y}_1$

**Lemma 4.15.** The representable sheaf $\mathrm{y}_1$ is in $\mathcal{U}_0^{\mathrm{pr}}$ and $\mathcal{V}_0^{\mathrm{pr}}$.

*Proof.* Note that for any $F : \mathrm{PrR}$ and $l : \mathbb{N}$,

$$(\mathrm{y}_1 \Rightarrow F)_l \cong \mathrm{PrR}(\mathrm{y}_1 \times \mathrm{y}_l, F) \cong F_{l+1}.$$

We define the presheaf $F^+$ by $F_l^+ := F_{l+1}$. It follows that

$$\mathrm{PrR}(\mathrm{y}_0, \mathrm{y}_1 \to (\mathrm{y}_1 \to \mathrm{y}_1 \to \mathrm{y}_1) \to \mathrm{y}_1 \to \mathrm{y}_1) \cong \mathrm{PrR}(\mathrm{y}_1 \times \mathrm{y}_1^{++} \times \mathrm{y}_1, \mathrm{y}_1).$$

We can define the natural transformation in $\mathrm{PrR}(\mathrm{y}_1 \times \mathrm{y}_1^{++} \times \mathrm{y}_1, \mathrm{y}_1)$ which at component $l$ is defined as

$$\begin{aligned}(\mathbb{N}^l \to \mathbb{N}) \times (\mathbb{N}^{l+2} \to \mathbb{N}) \times (\mathbb{N}^l \to \mathbb{N}) &\to (\mathbb{N}^l \to \mathbb{N}) \\ (g, h, n) &\mapsto \lambda x.\mathrm{primrec}_{g,h}^l(n(x), x)\end{aligned}$$

This shows that $\mathrm{y}_1 : \mathcal{U}_0$ satisfies the restrictions of $\mathcal{U}_0^{\mathrm{pr}}$. tbd: computation rules The result for $\mathcal{V}_0^{\mathrm{pr}}$ follows because $\mathrm{y}_1$ is already a sheaf. ■

**Finite Types in the Universes**

**Lemma 4.16.** $\mathbf{0} : \mathcal{U}_0^{\mathrm{pr}}$ and $\mathbf{0} : \mathcal{V}_0^{\mathrm{pr}}$.

*Proof.* There is exactly one map $r_\emptyset : \mathrm{y}_1 \to (\emptyset + \mathbf{1})$ given by $n \mapsto \mathrm{inr}(\star)$. Any element of $\mathbb{N}$ yields a section via the Yoneda embedding. ■

**Lemma 4.17.** $\mathbf{2}$ is not an element of $\hat{\mathcal{U}}_0^{\mathrm{pr}}$.

*Proof.* Assume that $\mathbf{2} = \mathrm{y}_0 + \mathrm{y}_0$ is a retract of $\mathrm{y}_1$. Then $\mathbf{2}$ is tiny, meaning that $X \mapsto X^{\mathbf{2}}$ is a left adjoint. In particular, it preserves all small colimits. However, there are natural isomorphisms

$$(X \mapsto X \times X) \cong (X \mapsto X^{\mathbf{1}} \times X^{\mathbf{1}}) \cong (X \mapsto X^{\mathbf{2}}).$$

The leftmost functor does not preserve all colimits, a contradiction. ■

**Lemma 4.18.** The presheaf $\mathbf{2}$ is separated.

*Proof.* No object of R is covered by the empty family. ■

**Definition 4.19.** Let $\mathbf{2}^+$ denote the sheafification of $\mathbf{2}$.

**Remark 4.20.** Let us unravel the action of $2^+$ on objects and morphisms. An element of $2^+(\mathbb{N}^n)$ is an equivalence class of matching families

$$\{x_f \in \mathbf{2} \mid f : \mathbb{N}^k \to \mathbb{N}^n \in P\}$$

for some cover $P \in J_{\mathrm{fin}}(\mathbb{N}^n)$ satisfying

$$x_{f \circ g} = x_f$$

for all $g : \mathbb{N}^l \to \mathbb{N}^k$. Here two such matching families $\{x_f \mid f \in P\}$ and $\{y_g \mid g \in Q\}$ are equivalent when there is a common refinement $T \subset P \cap Q$ with $T \in J_{\mathrm{fin}}(\mathbb{N}^n)$ such that $x_f = y_f$ for all $f \in T$. Since $J_{\mathrm{fin}}$ is generated by the basis $K'$ (c.f. Lemma 4.12), each matching family is completely determined by finitely many

$$\{x_{g_j}\}_{j=1}^m$$

where $\{g_j\}$ is a finite, jointly surjective, *disjoint* family of morphisms with codomain $\mathbb{N}^n$. In fact, we can collect all of the $g_j$ where $x_{g_j} = 0$, and similarly for $x_{g_j} = 1$ to obtain an equivalent matching family $\{0, 1\}$ on two morphisms (the extension $\xi_\star$ of the coproduct of the respective $g_j$). (If not all sections are 0, or 1. Otherwise, we get a singleton matching family $\{0\}$ or $\{1\}$.)

The action of $2^+$ on a morphism $\varphi : \mathbb{N}^k \to \mathbb{N}^n$ restricts a matching family $\{x_f\}$ to its pullback along $\varphi$. This restriction does not change the value of any individual $x_f$.

**Lemma 4.21.** $\mathbf{2}^+$ is a retract of $\mathrm{y}_1$ in $\mathcal{R}'$. In other words, $\mathbf{1} : \hat{\mathcal{V}}_0^{\mathrm{pr}}$.

*Proof.* By the Yoneda lemma since $\mathrm{y}_1$ is a sheaf, we have that

$$\mathcal{R}(\mathrm{y}_1, \mathbf{2}^+) \cong \mathrm{PrR}(\mathrm{y}_1, \mathbf{2}^+) \cong \mathbf{2}^+(1).$$

Under above isomorphism, an element $u \in \mathbf{2}^+(1)$ is sent to the natural transformation, here denoted $r_u$, which at component $m \in \{0, 1\}$ is given by

$$\begin{aligned} r_u(m) : (\mathbb{N}^m \to \mathbb{N}) &\to \mathbf{2}^+(m) \\ \varphi &\mapsto \mathbf{2}^+(\varphi)(u). \end{aligned}$$

Composition of natural transformations is defined componentwise. Hence, we need to find an equivalence class of matching families $u$, and a natural transformation $s$ with components

$$s_m : 2^+(m) \to (\mathbb{N}^m \to \mathbb{N})$$

such that for all $x \in 2^+(m)$,

$$2^+(s_m(x))(u) = x, \tag{1}$$

natural in $m$. Because sheafification is/determines a reflective subcategory of PrR, $s$ is completely determined by a map $\tilde{s} : \mathbf{2} \to \mathrm{y}_1$, and $\tilde{s} = s \circ \eta_{\mathbf{2}}$. By the universal property of $\mathbf{2} = \mathbf{1} +_{\mathrm{PrR}} \mathbf{1}$, $s$ is determined by two global sections of $\mathrm{y}_1$. Let us choose the constant functions $c_0$ and $c_1$.

For our retraction we use the cover of $\mathbb{N}$ generated by $\_ \cdot 2, \_ \cdot 2 + 1 : \mathbb{N} \to \mathbb{N}$ and the matching family $u$ with value a $\in \mathbf{2}$ on $\_ \cdot 2$ and b on the other map.

The equation Equation (1) can be reduced to the case where $x$ is a global section again. If we denote the two global sections of $\mathbf{2}$ by $\mathbf{a}$ and $\mathbf{b}$, we need to show

$$\mathbf{2}^+(c_0)(u) = \mathbf{a}$$

and

$$\mathbf{2}^+(c_1)(u) = \mathbf{b}$$

with $\mathbf{a}$ denoting the constant global sections.

It is easy to see that the pullback $c_n^*(S)$ of any $J_{\mathrm{fin}}$-sieve $S$ on $\mathbf{N}$ along a constant function $c_n : \mathbb{N} \to \mathbb{N}$ gives the maximal sieve on $\mathbb{N}$. Since the restriction maps $\mathbf{2}^+(f)$ are locally (on each component of (the equivalence class of)) the matching family by identity functions, it follows that indeed

$$\mathbf{2}^+(c_0)(u) = \mathbf{a}$$

and

$$\mathbf{2}^+(c_1)(u) = \mathbf{b}.$$

■

**Lemma 4.22.** All finite types $\mathbf{N}$ are in $\hat{\mathcal{V}}_0^{\mathrm{pr}}$.

*Proof.* One can use a similar strategy as in Lemma 4.21 using similar disjoint covers or multiple applications of the equaliser isomorphism. ■

### Relationships Between the Universes

**Lemma 4.23.** There is a map $\phi : \mathcal{U}_0^{\mathrm{cpr}} \to \mathcal{U}_0^{\mathrm{pr}}$ forgetting the context $\Gamma$ and preserving the base type $X$. There is a map $\psi : \mathcal{U}_0^{\mathrm{pr}} \to \mathcal{U}_0^{\mathrm{cpr}}$.

$$\phi \circ \psi = \mathrm{id}_{\mathcal{U}_0^{\mathrm{pr}}}.$$

Similarly for $\mathcal{V}_0^{\mathrm{cpr}} \to \mathcal{V}_0^{\mathrm{pr}}$.

*Proof.* Correctness of $\phi$ is trivial. Regarding $\psi$: If we assume an elimination principle into $X$, and have $g : \Gamma \to X$ and $h : \Gamma \to \mathrm{y}_1 \to X \to X$ then we can absorb $\Gamma$ into the context; for every $\gamma : \Gamma$ we get a map $f^\gamma : N \to X$, i.e. an appropriate term of type $\Gamma \to \mathrm{y}_1 \to X$. The identity is trivial as well. ■

**Remark 4.24.** Above we should also have $\psi \circ \phi = \mathrm{id}_{\mathcal{V}_0^{\mathrm{cpr}}}$. Either way, the map $\psi$ is enough for our purposes.

### $\mathrm{y}_1$-elimination into 2

**Lemma 4.25.** $\mathbf{2} : \mathcal{V}_0^{\mathrm{cpr}}$.

*Proof.* By Lemma 4.23 it suffices to show that $\mathbf{2} : \mathcal{V}_0^{\mathrm{pr}}$. Let $(\mathbf{2}, r, s, p)$ be a retraction as constructed in Lemma 4.21. Furthermore, assume

$$g : \mathbf{2},$$

and

$$h : \mathrm{y}_1 \to \mathbf{2} \to \mathbf{2}.$$

We define

$$\tilde{g} := s(g) : \mathrm{y}_1$$

and

$$\begin{aligned} \tilde{h} &: \mathrm{y}_1 \to \mathrm{y}_1 \to \mathrm{y}_1 \\ \tilde{h}(n, x) &:= s(h(n, rx)). \end{aligned}$$

By $\mathrm{y}_1$-induction (c.f. Lemma 4.15) we get

$$\begin{aligned} \tilde{f} &: \mathrm{y}_1 \to \mathrm{y}_1 \\ \tilde{f}(0) &= s(g) \\ \tilde{f}(\mathrm{succ}(n)) &= s(h(n, r(\tilde{f}(x)))). \end{aligned}$$

We define the desired $f$ by

$$\begin{aligned} f &: \mathrm{y}_1 \to \mathrm{y}_1 \\ f(n) &:= r(\tilde{f}(n)). \end{aligned}$$

Then

$$f(0) = rsg = g$$

and

$$f(\mathrm{succ}n) = rsh(n, r\tilde{f}n) = h(n, fn)$$

for all $n : \mathrm{y}_1$. ■

**Remark 4.26.** The construction of Lemma 4.25 works for any type $X$ with retraction $\mathrm{y}_1 \to X$. The difficult part is proving that a retraction $\mathrm{y}_1 \to X + \mathbf{1}$ gives $X : \mathcal{V}_0^{\mathrm{pr}}$, and contextual elimination into $\mathbf{2}$ suffices to prove that, see below.

### Decidable Equality for $\mathrm{y}_1$

**Lemma 4.27.** The predecessor function $\mathrm{P} := \mathrm{y}_{\mathrm{pred}}$ is a section of the successor function viewed as

$$\mathrm{succ} : \mathrm{y}_1 \to (n : \mathrm{y}_1) \times (n \neq 0).$$

This is true in PrR and $\mathcal{R}$.

*Proof.* We prove the result for PrR first. We cannot yet use induction on $\mathrm{y}_1$. Let $n : Z \to \mathrm{y}_1$ such that $n \neq 0$. We show that $\mathrm{succ}(\mathrm{P}(n)) = n$ so that function extensionality yields the result. We may assume that $Z = \mathrm{y}_x$ is representable (cf. [LM94, III.6 and VI.7]). Then the result follows immediately from the Yoneda lemma.

Because the $J_{\mathrm{fin}}$ is subcanonical, application of sheafification to above identity implies the result for $\mathcal{R}$. ■

**Lemma 4.28.** For any presheaf $X : \mathrm{PrR}$, the square

$$\begin{array}{ccc} X & \xrightarrow{!} & \mathbf{1} \\ \downarrow{\scriptstyle \mathrm{inl}} & & \downarrow{\scriptstyle \mathrm{inl}} \\ X + \mathbf{1} & \xrightarrow{\langle !_X, !_{\mathbf{1}} \rangle} & \mathbf{1} + \mathbf{1} \end{array}$$

is a pullback. Clearly this is also true for $X : \mathcal{R}$.

*Proof.* The map

$$X \to (x : X + \mathbf{1}) \times (\langle !_X, !_{\mathbf{1}} \rangle = \mathrm{inl}(\star))$$

preserves $x$ and uses that $\mathbf{1}$ is contractible. Its inverse sends $(\mathrm{inl}(x), p)$ to $x$, and $(\mathrm{inr}(\star), q)$ to an element obtained by ex falso, since $\mathrm{inl}(\star) \neq \mathrm{inr}(\star)$. ■

**Lemma 4.29.** $\mathrm{y}_1$ has decidable equality in $\mathcal{R}$.

*Proof.* By y applied to the primitive recursive subtraction function, it suffices to decide $(0 = n)$ for all $n : \mathrm{y}_1$. We make use of the fact that $\mathbf{2}$ has decidable equality. We inductively define (cf. Lemma 4.25)

$$\begin{aligned} \pi : \mathrm{y}_1 &\to \mathbf{2} \\ 0 &\mapsto 0 \\ \mathrm{succ}n &\mapsto 1 \end{aligned}$$

and

$$\begin{aligned} \varphi : \mathrm{y}_1 \to \mathbf{2} \to \mathrm{y}_1 \\ \varphi_n(0) &:= 0 \\ \varphi_n(1) &:= n. \end{aligned}$$

Let $n : \mathrm{y}_1$. By decidability of equality in $\mathbf{2}$ we have a term

$$p : (\pi(n) = 0) + (\pi(n) \neq 0).$$

Since

$$\varphi_n(\pi(0)) = 0$$

and

$$\varphi_n(\pi(n)) = n$$

we also have a function

$$f : (\pi(n) = 0) + (\pi(n) \neq 0) \to (n = 0) + (n \neq 0)$$

given by $\mathrm{ap}_{\varphi_n}$ in the first case and precomposition with $\mathrm{ap}_{\varphi_n}$ in the second. The term $f(p)$ is of the desired type. ∎

### $\mathrm{y}_1$-elimination into retracts

In this subsubsection we work only in $\mathcal{R}$.

**Lemma 4.30.** There is a map

$$(n : \mathrm{y}_1) \to (n = 0) + ((m : \mathrm{y}_1) \times (n = \mathrm{succ}(m))).$$

*Proof.* Let $n : \mathrm{y}_1$. We use that $n = 0$ is decidable. If $n = 0$, the construction is clear. Otherwise, let $\mathrm{P} := \mathrm{y}_{\mathrm{pred}}$ be the predecessor function and set $m := \mathrm{P}(n)$. Then

$$\mathrm{succ}(\mathrm{P}(n)) = n$$

by Lemma 4.27. ∎

**Theorem 4.31.** There is a map

$$\Phi : \hat{\mathcal{V}}_0^{\mathrm{pr}} \to \mathcal{V}_0^{\mathrm{pr}}$$

sending the base type $X$ to the base type $X$.

*Proof.* Let $(X, r, s, p) : \hat{\mathcal{U}}_0^{\mathrm{pr}}$, $g : X$ and $h : \mathrm{y}_1 \to X \to X$. We inductively (Lemma 4.15) define a helper function $\tilde{f} : \mathrm{y}_1 \to \mathrm{y}_1$ with initial value

$$\tilde{g} := s(g)$$

and step function

$$\tilde{h} : \mathrm{y}_1 \to \mathrm{y}_1 \to \mathrm{y}_1$$
$$\tilde{h}(n, x) := \begin{cases} s(h(n, r(x))), (r(x) : X) \\ s(g), (r(x) = \star). \end{cases}$$

Then, we define

$$f : \mathrm{y}_1 \to X$$
$$f(n) := \begin{cases} r(\tilde{f}(n)), (r(\tilde{f}(n)) : X) \\ g, (r(\tilde{f}(n)) = \star). \end{cases}$$

We verify the computation rules of this $f$. We clearly have that $f(0) = r(s(g)) = g$. To check that

$$f(\mathrm{succ}(n)) = h(n, f(n))$$

we verify that

$$f(S(n)) = r(\tilde{f}(\mathrm{succ}(n))) = r(s(h(n, r(\tilde{f}(n))))) = h(n, f(n)),$$

if $r(\tilde{f}(\mathrm{succ}(n))) : X$. Otherwise, this equality does not necessarily hold, so we need to show that $(n : \mathrm{y}_1) \to (x : X) \times r(\tilde{f}(\mathrm{succ}(n))) = x$. This $x$ is of course given by $h(n, r(\tilde{f}(n)))$, but we don't have a suitable induction principle to show this. Instead, we prove that there is a lift

$$\begin{array}{ccc} & & X \\ & \overset{r \circ \tilde{f}}{\nearrow} & \big\downarrow \mathrm{inl} \\ \mathrm{y}_1 & \longrightarrow & X + \mathbf{1}. \end{array}$$

By Lemma 4.28 it suffices to show that for any $n : Z \to \mathrm{y}_1$ there is a lift

$$\begin{array}{ccc} & & \mathbf{1} \\ & \overset{}{\nearrow} & \big\downarrow \mathrm{inl} \\ Z & \xrightarrow[\langle !_X, !_{\mathbf{1}} \rangle \circ r \circ \tilde{f} \circ n]{} & \mathbf{1} + \mathbf{1}. \end{array}$$

Switching back to type theoretic language, we have to show that

$$\langle !_X, !_{\mathbf{1}} \rangle (r(\tilde{f}(n))) = \mathrm{inl}(\star). \tag{2}$$

We make a case distinction on $n = 0$. If $n = 0$, then

$$\langle !_X, !_{\mathbf{1}} \rangle (r(\tilde{f}(0))) = \langle !_X, !_{\mathbf{1}} \rangle (r(s(g))) = \langle !_X, !_{\mathbf{1}} \rangle (g) = \mathrm{inl}(\star).$$

Otherwise, by Lemma 4.30, we know that $n = \mathrm{succ}(\mathrm{P}(n))$, and we compute

$$\begin{aligned}
\langle !_X, !_{\mathbf{1}}\rangle(r(\tilde{f}(\mathrm{succ}(\mathrm{P}(n))))) &= \langle !_X, !_{\mathbf{1}}\rangle(r(\tilde{h}(n, \tilde{f}(\mathrm{succ}(\mathrm{P}(n))))))\\
&= \begin{cases} \langle !_X, !_{\mathbf{1}}\rangle(r(s(h(n, r(\tilde{f}(\mathrm{succ}(\mathrm{P}(n)))))))), r(\tilde{f}(\mathrm{succ}(\mathrm{P}(n))))) : X \\ \langle !_X, !_{\mathbf{1}}\rangle(r(s(g))), \text{otherwise} \end{cases}\\
&= \begin{cases} \langle !_X, !_{\mathbf{1}}\rangle(h(n, r(\tilde{f}(\mathrm{succ}(\mathrm{P}(n)))))), r(\tilde{f}(\mathrm{succ}(\mathrm{P}(n))))) : X \\ \langle !_X, !_{\mathbf{1}}\rangle(g), \text{otherwise} \end{cases}\\
&= \mathrm{inl}(\star),
\end{aligned}$$

since the codomain of $h$ is $X$.

An alternative proof would show Equation (2) by induction, but it is only the current lemma itself that allows use of the necessary induction rule. ■

This gives us our full elimination principle.

### Closure under Σ- and idenitity types

**Lemma 4.32.** The universes $\hat{\mathcal{U}}_0^{\mathrm{pr}}$ and $\hat{\mathcal{V}}_0^{\mathrm{pr}}$ are closed under Σ-types.

*Proof.* Assume we have $\bar{A} = (A, r, s, p) : \hat{\mathcal{U}}_0^{\mathrm{pr}}$, and $\bar{B}(a) = (B(a), r_a, s_a, p_a) : \hat{\mathcal{U}}_0^{\mathrm{pr}}$ given $a : A$. Then

$$\begin{aligned}
\mathrm{y}_1 \times \mathrm{y}_1 &\to (a : A) \times B(a)\\
(m, n) &\mapsto (r(m), r_{r(m)}(n))
\end{aligned}$$

has right inverse the map

$$\begin{aligned}
(a : A) \times B(a) &\to \mathrm{y}_1 \times \mathrm{y}_1\\
(a, b) &\mapsto (s(a), s_a(b)).
\end{aligned}$$

This follows from the computation

$$(r(s(a)), r_{r(s(a))}(s_a(b))) = (a, r_a(s_a(b))) = (a, b).$$

Next, note that $\mathrm{y}_1 \times \mathrm{y}_1 \cong \mathrm{y}_2 \cong \mathrm{y}_1$. Isomorphisms are retractions and retractions are stable under composition, so in total we have a composite retraction $\mathrm{y}_1 \to \mathrm{y}_1 \times \mathrm{y}_1 \to (a : A) \times B(a)$. ■

**Lemma 4.33.** $\hat{\mathcal{V}}_0^{\mathrm{pr}}$ is closed under identity types of terms.

*Proof.* Assume we have $\bar{X} = (X, r, s, p) : \hat{\mathcal{U}}_0^{\mathrm{pr}}$ and $x, y : X$. We shall define a retraction $r' : \mathrm{y}_1 \to (x = y) + \mathbf{1}$ with section $s'$. Note that $r$ and $s$ induce retraction-section pairs on identity types by application. In particular, $(x = y) + \mathbf{1}$ is a retract of $(s(x) = s(y)) + 1$ by $(\mathrm{ap}_r, \star)$. We proceed by case analysis on $s(x) = s(y)$ (Lemma 4.29). If $s(x) = s(y)$, then $(s(x) = s(y)) + 1 \simeq \mathbf{2}$. Otherwise, $(s(x) = s(y)) + 1 \simeq \mathbf{1}$. and $\mathrm{y}_1 \to \mathbf{1}$, giving a retraction $\mathrm{y}_1 \to (s(x) = s(y)) + \mathbf{1}$. Composition with $(\mathrm{ap}_r, \star)$ yields a retraction $r'$ of desired type. ■

**Remark 4.34.** Lemma 4.33 is not true for $\hat{\mathcal{U}}_0^{\mathrm{pr}}$ because $\mathbf{2}$ is not a retract of $\mathrm{y}_1$.

**Lemma 4.35.** If $\mathcal{U}_0^{\mathrm{cpr}}$ is closed under coproducts, then it is also closed under products.

*Proof Sketch.* Let $A, B$ be two types satisfying the contextual elimination principle of $\mathrm{y}_1$. Assume we have an initial value $(a_0, b_0) : A \times B$ and a step function $h : \mathrm{y}_1 \to A \times B \to A \times B$. Parametrisation by an arbitrary $\gamma : \Gamma$ is left implicit. By closure under coproducts, we may define a sequence in $A + B$ starting as follows.

$$
\begin{aligned}
&a_0,\\
&b_0,\\
&h_1(0,(a_0,b_0)),\\
&h_2(0,(a_0,b_0)),\\
&h_1(1,(h_1(0,(a_0,b_0)),h_2(0,(a_0,b_0)))),\\
&h_2(1,(h_1(0,(a_0,b_0)),h_2(0,(a_0,b_0)))),\ldots.
\end{aligned}
$$

Choosing the two subsequences at odd and even indices yields a sequence in $A \times B$. ■

### $\mathrm{y}_1$ is not the natural numbers object of PrR

**Definition 4.36.** A function $f : \mathbb{N}^k \to \mathbb{N}$ is *representable* in a cartesian closed category $C$ with weak natural numbers object $N$ if there is an arrow $\tilde{f} : N^k \to N$ such that, for every $k$-tuple $(n_1, \ldots, n_k)$ of natural numbers, the following square commutes.

$$
\begin{array}{ccc}
\mathbb{N}^k & \xrightarrow{\quad f \quad} & \mathbb{N} \\
\Big\downarrow{\scriptstyle \mathrm{num}^k} & & \Big\downarrow{\scriptstyle \mathrm{num}} \\
\Gamma(N^k) & \xrightarrow{\Gamma(\tilde{f})} & \Gamma(N)
\end{array}
$$

Here $\mathrm{num}(n)$ denotes the $n$-th numeral in $N$.

**Theorem 4.37.** The representable presheaf $\mathrm{y}_1$ is not a natural numbers object in PrR nor in $\mathcal{R}$.

*First Proof.* Assume that $\mathrm{y}_1$ is a natural numbers object. Then the Ackerman function $A : \mathbb{N}^2 \to \mathbb{N}$ is representable, i.e. there exists $\tilde{A} : \mathrm{PrR}(\mathrm{y}_2, \mathrm{y}_1)$ such that $\Gamma(\tilde{A})$ is equal to $A$ on all numerals. This means that the Ackerman function is primitive recursive, a contradiction. ■

*Second Proof.* We know that the constant presheaf $\Delta\mathbb{N}$ is a natural numbers object in PrR. We show that there is no natural transformation $\alpha : \mathrm{y}_1 \to \Delta\mathbb{N}$ such that the following diagram commutes.

$$\begin{array}{ccccc} \mathrm{y}_0 & \xrightarrow{\text{zero}} & \mathrm{y}_1 & \xrightarrow{\text{succ}} & \mathrm{y}_1 \\ \| & & \downarrow{\scriptstyle\alpha} & & \downarrow{\scriptstyle\alpha} \\ \mathrm{y}_0 & \xrightarrow{0} & \Delta\mathbb{N} & \xrightarrow{+1} & \Delta\mathbb{N} \end{array}$$

By the Yoneda lemma, there is exactly one such natural transformation $\alpha_n : \mathrm{y}_1 \to \Delta\mathbb{N}$ for each $n : \mathbb{N}$, and it acts as

$$\alpha_n(k)(f) = \Delta\mathbb{N}(f)(n) = \mathrm{id}_{\mathbb{N}}(n) = n,$$

where $f : \mathbb{N}^k \to \mathbb{N}$ is primitive recursive. Chasing $k : \mathrm{y}_1$ along both legs of the right square above, we get

$$\alpha_n(k) + 1 = n + 1 \neq n = \alpha_n(\mathrm{succ}(k)). \qquad \blacksquare$$

**Conjectures**

What else is $\hat{\mathcal{V}}_0^{\mathrm{pr}}$ closed under?
Probably yes: finitary inductive types
Probably no: coproducts, lists, set quotients

# 5 Semantics in a Topos of Primitive Recursive Functions

Semantics are now in $\mathcal{R}$..., and $N_1$ should be interpreted as NNO. Does this give coherence issues? We define an interpretation $[\![-]\!]_{\mathcal{R}}$ of $\mathrm{T}_{\mathrm{pr}}$ in $\mathcal{R}$. Just like any Grothendieck topos, $\mathcal{R}$ has a natural numbers object $\mathbb{N}$. However, we show that $\mathrm{N}_0$ can be soundly interpreted as $\mathrm{y}_1$. Then the Yoneda lemma implies that natural transformations $\mathcal{R}(\mathrm{y}_1, \mathrm{y}_1)$ are primitive recursive functions $\mathbb{N} \to \mathbb{N}$. The interpretation of the other type-theoretic constructors and rules is standard.

To interpret the general elimination principle of N, ...

# 6 Soundness

In this section we use a synthetic gluing argument to show that the set-theoretical interpretation of functions actually coincides with the primitive recursive one.

**Lemma 6.1.** The interpretations $[\![-]\!]_{\mathrm{Set}}$ and $[\![-]\!]_{\mathcal{R}}$ are flat functors.

They are even locally Cartesian closed, because they define models of a theory expressed as an LCCC. Hence, they extend to left exact functors $\widehat{[\![-]\!]_{\mathcal{R}}}$ and $\widehat{[\![-]\!]_{\mathrm{Set}}}$ along their Yoneda embedding. It follows that we have a left exact extension

$$\rho := \Gamma(\widehat{[\![-]\!]_{\mathcal{R}}}) \times \widehat{[\![-]\!]_{\mathrm{Set}}} : \mathrm{Pr}\,\mathrm{T}_{\mathrm{pr}} \to \mathrm{Set}$$

and the comma category $\mathrm{ShG} := \mathrm{Set} \downarrow \rho$ is a logos.

**Theorem 6.2.** All $\mathrm{T}_{\mathrm{pr}}$-definable functions between the natural numbers are primitive recursive. To be precise, for any closed term

$$\cdot \vdash f :\uparrow \mathrm{N}_0 \to \uparrow \mathrm{N}_0$$

in $\mathrm{T}_{\mathrm{pr}}$ the function $[\![f]\!]_{\mathrm{Set}}$ is primitive recursive.

*Proof sketch.* In the internal language of ShG we distinguish the 'diagonal' computability predicate $\mathrm{N}^* : \mathbb{N} \to \Gamma(\mathrm{y}_1) \times \mathbb{N}$. This object is strictly aligned over $\mathrm{N}_0$. Choosing this interpretation in the glue topos requires the terms succ and zero to fill the diagram

$$\begin{array}{ccccc} 1 & \xrightarrow{0} & \mathbb{N} & \xrightarrow{+1} & \mathbb{N} \\ {\scriptstyle !}\downarrow & & \downarrow{\scriptstyle \mathrm{N}^*} & & \downarrow{\scriptstyle \mathrm{N}^*} \\ 1 & \xrightarrow{\rho\mathrm{zero}} & \Gamma(\mathrm{y}_1) \times \mathbb{N} & \xrightarrow{\rho\mathrm{succ}} & \Gamma(\mathrm{y}_1) \times \mathbb{N} \end{array}$$

All the other types and connectives can be lifted in the standard way. The assumed closed term may be interpreted into the computability algebra as a global element

$$f^* : 1_{\mathrm{Sh}G} \to \{\mathrm{tm}^*_V(\uparrow N \to \uparrow N)^*) \mid \phi \hookrightarrow f\}.$$

Unfolding the definitions, we *should* get a commutative square

$$\begin{array}{ccc} \mathbb{N} & \xrightarrow{\widehat{[\![f]\!]_{\mathrm{Set}}}} & \mathbb{N} \\ \downarrow & & \downarrow \\ \Gamma(\mathrm{y}_1) \times \mathbb{N} & \xrightarrow{\rho(\hat{f})} & \Gamma(\mathrm{y}_1) \times \mathbb{N}. \end{array}$$

This implies commutativity of the square

$$\begin{array}{ccc} \mathbb{N} & \xrightarrow{[\![f]\!]_{\mathrm{Set}}} & \mathbb{N} \\ \downarrow & & \downarrow \\ \Gamma(\mathrm{y}_1) & \xrightarrow{\Gamma([\![f]\!]_{\mathrm{PrR}})} & \Gamma(\mathrm{y}_1). \end{array}$$

Here $\Gamma([\![f]\!]_{\mathrm{PrR}})$ is primitive recursive. In other words, for every $n : \mathbb{N}$, $[\![f]\!]_{\mathrm{Set}}(n) = \mathrm{app}_f(n)$, which is the desired result. ■

We remark that if a calculus formalises polytime functions rather than primitive recursive ones such as in [Hof97], it is necessary to interpret the syntax in Set, because that is where the model of computation lives. On the other hand, being primitive recursive is actually a property of the algorithm defined by the syntax rather than its corresponding set-theoretic interpretation. If one can express such properties in a suitable way it might be possible to dispense of the interpretation in Set altogether. One idea of approaching this is to define another theory $\mathrm{T}^{\mathrm{simple}}_{\mathrm{pr}}$ whose terms are by definition exactly primitive recursive. Since both calculi admit canonical forms additional gluing constructions could be employed to show that these normal forms coincide.

# 7 Canonicity

In this section we use synthetic Tait computability to prove canonicity for $\mathrm{T}_{\mathrm{pr}}$. We remark that this result does not immediately follow from [Ste21, § 4.5.3], because the type theory presented there does not contain a natural numbers object. Other than that, the strategy of the proof is exactly analogous.

**Theorem 7.1.** Let $m : 1 \to \mathrm{tm}(\mathrm{N}) : \mathrm{T}_{\mathrm{pr}}$ be a closed term of natural numbers type; then there exists an $n : \mathbb{N}$ such that $m = \mathrm{succ}^n(\mathrm{zero})$, meant as a statement about global elements in Set.

*Proof.* Form the glue topos G along the left exact global sections functor $\Gamma : \mathrm{Pr}\,\mathrm{T}_{\mathrm{pr}} \to \mathrm{Set}$. Using the natural numbers object $\mathbb{N}$ of the logos ShG, we lift $\mathrm{N}, \mathrm{zero}$ and succ. The other types and connectives are lifted in the same way as shown in the reference.

$$
\begin{aligned}
\mathrm{N}^*_V &: \{\mathrm{tp}^*_V \mid \phi \hookrightarrow \mathrm{N}\} \\
\mathrm{N}^*_V &:= (\mathrm{N}, \Sigma_{\mathrm{syn}:\mathrm{tmN}} \bullet_\phi (\Sigma_{n:\mathbb{N}} (\mathrm{syn} = \mathrm{succ}^n(\mathrm{zero})))) \\
\mathrm{zero}^*_V &: \{\mathrm{tm}^*_V(\mathrm{N}^*_V) \mid \phi \hookrightarrow (\mathrm{zero}, \star)\} \\
\mathrm{zero}^*_V &:= (\mathrm{zero}, \eta_{\setminus\phi}(0, \star)) \\
\mathrm{succ}^*_V &: \{\mathrm{tm}^*_V(\mathrm{N}^*_V) \to \mathrm{tm}^*_V(\mathrm{N}^*_V) \mid \phi \hookrightarrow (\mathrm{succ}, \lambda\_.\star)\} \\
\mathrm{succ}^*_V(\mathrm{syn}, \mathrm{sem}) &: \{\Sigma_{\mathrm{syn}:\mathrm{tmN}} \bullet_\phi (\Sigma_{n:\mathbb{N}} (\mathrm{syn} = \mathrm{succ}^n(\mathrm{zero}))) \mid \phi \hookrightarrow (\mathrm{succ}(\mathrm{syn}), \star)\} \\
\mathrm{succ}^*_V(\mathrm{syn}, \mathrm{sem}) &:= \mathrm{try}\ \mathrm{sem}\ [\alpha \mid \phi \hookrightarrow (\mathrm{succ}(\mathrm{syn}), \star)] \\
&\quad \text{where} \\
&\qquad \alpha : \Sigma_{n:\mathbb{N}} (\mathrm{syn} = \mathrm{succ}^n(\mathrm{zero})) \\
&\qquad\quad \to \{\Sigma_{\mathrm{syn}':\mathrm{tm}(\mathrm{N})} \bullet_\phi \Sigma_{n:\mathbb{N}} (\mathrm{syn}' = \mathrm{succ}^{n+1}(\mathrm{zero})) \mid \phi \hookrightarrow (\mathrm{succ}(\mathrm{syn}), \star)\} \\
&\qquad \alpha(n, p) \ = (\mathrm{succ}(\mathrm{syn}), \eta_{\setminus\phi}(n+1, \mathrm{app}_{\mathrm{succ}}(p)))
\end{aligned}
$$

We remark that the partial elements $(\mathrm{succ}(\mathrm{syn}), \star)$ appearing in the extent types and try statements have an implicit $\lambda z : \phi$ in front. The definition of $\alpha$ is valid, because

$$z : \phi \vdash \eta_{\setminus\phi}(n+1, \mathrm{app}_{\mathrm{succ}}(p)) = \star : \bullet_\phi(\ldots).$$

A term $m : 1_{\mathrm{Pr}\,\mathrm{T}_{\mathrm{pr}}} \to \mathrm{tm}(\mathrm{N})$ lifts to $m^* : \{1_{\mathrm{ShG}} \to \mathrm{tm}^*_V(\mathrm{N}^*_V) \mid \phi \hookrightarrow m\}$. Unfolding the definition, we have a global element of $\bullet_\phi(\Sigma_{n:\mathbb{N}}(m = \mathrm{succ}^n(\mathrm{zero})))$. This computes as follows.

$$
\begin{aligned}
&\mathrm{Hom}_{\mathrm{ShG}}(1_{\mathrm{ShG}}, \bullet_\phi(\Sigma_{n:\mathbb{N}}(m = \mathrm{succ}^n(\mathrm{zero})))) && \\
=&\mathrm{Hom}_{\mathrm{ShG}}(1_{\mathrm{ShG}}, i_* i^*(\Sigma_{n:\mathbb{N}}(m = \mathrm{succ}^n(\mathrm{zero})))) && \text{(by definition)} \\
\cong&\mathrm{Hom}_{\mathrm{Set}}(1_{\mathrm{Set}}, i^*(\Sigma_{n:\mathbb{N}}(m = \mathrm{succ}^n(\mathrm{zero})))) && (i_* \text{ lex}) \\
\cong&\mathrm{Hom}_{\mathrm{Set}}(1_{\mathrm{Set}}, \Sigma_{n:\mathbb{N}} i^*(m = \mathrm{succ}^n(\mathrm{zero}))) && (i^* \text{ cocont.}) \\
\cong&\mathrm{Hom}_{\mathrm{Set}}(1_{\mathrm{Set}}, \Sigma_{n:\mathbb{N}}(i^*(m) = i^*(\mathrm{succ}^n(\mathrm{zero})))) && (i^* \text{ lex}) \\
=&\mathrm{Hom}_{\mathrm{Set}}(1_{\mathrm{Set}}, \Sigma_{n:\mathbb{N}}(m = \mathrm{succ}^n(\mathrm{zero})))
\end{aligned}
$$

The last step is true, because $m$ above is actually $j_*(\mathrm{y}_m))$ and so $i^*(j_*(\mathrm{y}_m)) = \Gamma(\mathrm{y}_m) = m$. ■

# 8 Ramifications

In this section we discuss limitations and potential extensions of $\mathrm{T}_{\mathrm{pr}}$. We also review the published literature on type theory and primitive recursion.

## 8.1 Data Types

One way of increasing the ease of programming in $\mathrm{T}_{\mathrm{pr}}$ would be to add additional inductive types such as lists to the universe $\mathcal{U}_0$. The soundness proof in Section 6 is modular in that merely the data of the new type needs to be lifted to the glue topos in a suitable way. Of course, one also has to adapt the interpretations in a suitable way, mainly by proving that $\mathcal{U}_0^{\mathrm{pr}}$ admits the new types.

We do not take this path, since $\mathrm{T}_{\mathrm{pr}}$ is currently mainly of theoretical interest and convenience of programming is not of high priority.

## 8.2 Modalities

It might be possible to overcome the inconvenience of not being able to define functions out of the natural numbers by adding a comonadic modality $\Box$ to the theory. The new type $\Box\mathrm{N}$ will have a general induction principle and it should be possible to define terms of type $\Box\mathrm{N} \times \mathrm{N} \to \mathrm{N}$ by simultaneous induction. While in general, there should be no way of obtaining a term of type $\mathrm{N} \to \mathrm{N}$ from a term of type $\Box\mathrm{N} \to \mathrm{N}$, it might be feasible to add to the calculus a rule which produces a function of type $\mathrm{N} \times \mathrm{N} \to \mathrm{N}$ given $f : \Box\mathrm{N} \times \mathrm{N} \to \mathrm{N}$, $g : \mathrm{N} \times \Box\mathrm{N} \to \mathrm{N}$ and an extensionality proof $(m, n : \mathrm{N}) \to f(\eta m, n) = g(m, \eta n)$. Here $\eta$ denotes the counit of $\Box$.

In this section we describe how such a modified calculus $\mathrm{T}_{\mathrm{pr}}^{\Box}$ could be defined and its soundness proved formally.

On the syntax side, an obvious candidate is MTT instantiated with mode theory the walking adjunction or the walking comonad.

We give an example of potential semantics.

The interpretation in Set can be directly extended from the interpretation of $\mathrm{T}_{\mathrm{pr}}$ by evaluating $\Box$ as the trivial, identity comonad on any type.

Let $Q$ denote the category of arities and functions between powers of natural numbers. We define a new category $Q \rtimes R$. Its objects are pairs of natural numbers and a morphism $(\overrightarrow{u}, \overrightarrow{v}) : (Q \rtimes R)((m, n), (m', n'))$ consists of $m'$ set-theoretic functions $u_i \mathbb{N}^m \to \mathbb{N}$ and $n'$ set-theoretic functions $v_j : \mathbb{N}^{m+n} \to \mathbb{N}$ such that for each $x : \mathbb{N}^m$ the slice $v_j(x, -)$ is primitive recursive.

There is an inclusion functor

$$(-, 0) : Q \to (Q \rtimes R)$$

which is right adjoint to the projection

$$\pi : (Q \rtimes R) \to Q$$

which forgets the $\overrightarrow{v}$-part. The inclusion also has a right adjoint

$$\begin{aligned} + : (Q \rtimes R) &\to Q \\ (m, n) &\mapsto m + n \\ (\overrightarrow{u}, \overrightarrow{v}) &\mapsto (\overrightarrow{uv}, \cdot) \end{aligned}$$

which only forgets that the $\overrightarrow{v}$-functions are primitive recursive, but not the functions themselves.

Passing to presheaf categories, we have a string of adjunctions

$$\pi^* \dashv \pi_* \cong (-, 0)^* \dashv (-, 0)_* \cong +^*.$$

Here $-^*$ denotes the induced functor by precomposition. Since $(-, 0)^*$ and $+^*$ are right adjoints to precomposition functors, and since adjoints are unique, they are given by the right Kan extensions $\pi_*$ and $(-, 0)_*$, respectively. [Gra+21, Lemma 8.2] implies that $(-, 0)^*$ and $+^*$ can be turned into dependent right adjoints. It follows from [Gra+21, Theorem 7.1] that they model an instance of MTT .

Similarly, the round trip $\Box := +^* \circ (-, 0)^*$ is right adjoint to the round trip composition functor $\Box' := \pi^* \circ (-, 0)^*$.

We would like to interpret $\mathrm{N} : \mathrm{T}^{\Box}_{\mathrm{pr}}$ as $\mathrm{y}_{(0,1)}$ because the endomorphisms of that object are exactly primitive recursive functions. However, $\Box \mathrm{y}_{(0,1)}$ does not have the same nice properties that $\Box' \mathrm{y}_{(0,1)}$ has when it comes to simplifications of exponentials $\Box \mathrm{y}_{(0,1)} \Rightarrow F$.

**Question 8.1.** Does the construction generalising MTT to MATT apply to the semantics in $\mathrm{Pr}(Q \rtimes R)$?

Finally, we remark that $Q \rtimes R$ seems to be the ideal setup in that reversing the roles of $Q$ and $R$, or simply taking the product category does not lead to a string of three adjoint functors.

## 8.3 Primitive Recursive Cubical Type Theory

An obvious question is whether the restriction of MLTT to $\mathrm{T}_{\mathrm{pr}}$ can be extended to (a variant of) homotopy type theory. In plain HoTT with univalence added as an axiom there is not much hope since it is not computational; terms containing the irreducible univalence term can hardly be considered primitive recursive. Currently, our only hope is cubical type theory. Luckily, its abstract syntax is formally defined in [Ste21]. It is easy to make the necessary adjustments to its first universe. However, it is not clear how the soundness proof of Section 6 should be adapted because CTT cannot be interpreted in Set. One could for example require that the structure maps of a cubical set be primitive recursive and embed Set into the resulting category. Alternatively, one could try the gluing of normal forms outlined at the end of Section 6.

### 8.4 Polytime Dependent Type Theory

It is likely that the construction proposed in Section 8.2 can be adapted to extend Hofmann's calculus $\mathrm{BC}^\omega$, defined in [Hof97], to a dependently typed system with safe recursion.

The calculus $\mathrm{BC}^\omega$ captures exactly the functions computable in polynomial time. In $\mathrm{BC}^\omega$ natural numbers W are represented as finite binary sequences. The system allows unbounded recursion on notation; however, the recursive argument of a function must be *normal*, while the recursive calls are merely allowed via *safe* variables. This is enforced by the type system: the recursor has type

$$\mathcal{R} : \Box\mathrm{W} \to \mathrm{W} \to (\Box\mathrm{W} \to \mathrm{W} \to \mathrm{W}) \to \mathrm{W}.$$

Hofmann proves soundness of the ptime computability claim by constructing a model in Set, and in a category similar to $Q \rtimes R$ from Section 8.2, except that the roles of $Q$ and $R$ are switched, set-theoretic functions are replaced by polytime computable functions, and primitive recursive functions are replaced by polymax bounded functions. He then uses a logical relation to show that the two interpretations coincide.

In fact, his construction inspired ours for primitive recursion. Since both syntax and semantics are similar to that of $\mathrm{T}_{\mathrm{pr}}$, we expect that the $\mathrm{BC}^\omega$ can be extended to a dependently typed system using MTT (or MATT) in a similar fashion.

We remark that further variants of $\mathrm{BC}^\omega$ have been developed (c.f. [Hof98; Hof00]). However, these critically hinge on linear type systems. While variants of linear dependent type theory have been developed [Ril22; Sch14], there is no general framework such as MTT which admits linear type formers, dependent types and homotopical interpretations. The closest approximation is [LSR17], but the syntax is very complicated and it does not support dependent types yet. Therefore, we do not believe that a dependently typed calculus for complexity classes using substructural type formers is currently within reach.

Rencently Hofmann's linear recursive calclulus for polytime programming has been extended to a dependent type theory (see [Atk23]) using an extension of quantitative type theory (c.f. [Atk18]). Riley [Ril22] gives some reasons why that type theory is unsatisfactory.

## 9 Related Work

The novel contribution of this work is the conservativity of primitive recursion with respect to dependent types. Extension of MLTT by additional recursion operators for well-founded relations has been studied by Paulson (c.f. [Pau86]).

In recursion theory one often considers partial recursive functions. These have been studied in the context of type theory as well.

In [Bov03; BC01; BC05a; BC05b] the authors use an inductive domain predicate that characterises the inputs on which a function terminates.

Alternative approaches such as [BC07; Cap05] associate to each data type a coinductive type of partial elements.

Variants of these systems have been designed for use in proof assistants (c.f. [CS87]). We remark that since $\mathrm{T}_{\mathrm{pr}}$ is given in abstract syntax, the extra step of defining concrete syntax would be necessary before it could be implemented in a proof assistant.

Our system $\mathrm{T}_{\mathrm{pr}}$ is an extension of simply typed lambda calculus to full dependent type theory, potentially enhanced using a comonadic modality. One important intermediate system is the modal lambda calculus defined in [DP01] in order to give a formal account of *staged computation* and it is a precursor to [Hof98].

The universe without dependent products $\mathrm{U}_0$ of $\mathrm{T}_{\mathrm{pr}}$ can be seen as a system of less proof-theoretic strength than $\mathrm{U}_1$ since there are fewer expressible and provable propositions in the former. One typically encounters systems of reduced strength in the field of applied proof theory (c.f. [Koh08]). There, one translates proofs in classical systems to proofs in weaker but constructive systems. This does not appear to be directly applicable here, because our stronger system $\mathrm{U}_1$ is already constructive and in contrast with $\mathrm{U}_0$, the weak systems considered in applied proof theory still have functionals of higher type.

The primitive recursive functions can be organised by growth speed in the so-called Grzegorczyk Hierarchy. This hierarchy and its variants are often used in computability theory (see [Boa89]). It is likely that the constructions of Section 8.2 can be adapted to such a hierarchy. This would pose as another tool for synthetic reasoning in computability theory.

# 10 Conclusion

# References


[Ack28] Wilhelm Ackermann. "Zum Hilbertschen Aufbau der reellen Zahlen". In: *Mathematische Annalen* 99.1 (Dec. 1928), pp. 118–133. ISSN: 1432-1807. DOI: 10.1007/BF01459088.

[Atk18] Robert Atkey. "Syntax and Semantics of Quantitative Type Theory". In: *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*. LICS '18. Oxford, United Kingdom: Association for Computing Machinery, 2018, pp. 56–65. ISBN: 9781450355834. DOI: 10.1145/3209108.3209189.

[Atk23] Robert Atkey. *Polynomial Time and Dependent Types*. 2023. arXiv: 2307.09145 [cs.LO].

[BC01] Ana Bove and Venanzio Capretta. "Nested General Recursion and Partiality in Type Theory". In: *Theorem Proving in Higher Order Logics*. Ed. by Richard J. Boulton and Paul B. Jackson. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 121–125. ISBN: 978-3-540-44755-9.

[BC05a] ANA BOVE and VENANZIO CAPRETTA. "Modelling general recursion in type theory". In: *Mathematical Structures in Computer Science* 15.4 (2005), pp. 671–708. DOI: 10.1017/S0960129505004822.

[BC05b] Ana Bove and Venanzio Capretta. "Recursive Functions with Higher Order Domains". In: *Typed Lambda Calculi and Applications*. Ed. by Paweł Urzyczyn. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 116–130. ISBN: 978-3-540-32014-2.

[BC07] Ana Bove and Venanzio Capretta. "Computation by Prophecy". In: *International Conference on Typed Lambda Calculus and Applications*. 2007.

[Boa89] Peter van Emde Boas. "K. Wagner and G. Wechsung, Computational complexity, Mathematics and its applications. VEB Deutscher Verlag der Wissenschaften, Berlin, and D. Reidel Publishing Company, Dordrecht etc., 1986, 551 pp." In: *The Journal of Symbolic Logic* 54.2 (1989), pp. 622–624. DOI: 10.2307/2274881.

[Bov03] Ana Bove. "General Recursion in Type Theory". In: *Types for Proofs and Programs*. Ed. by Herman Geuvers and Freek Wiedijk. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 39–58. ISBN: 978-3-540-39185-2.

[Cap05] Venanzio Capretta. "General recursion via coinductive types". In: *Log. Methods Comput. Sci.* 1 (2005).

[CS87] Robert L. Constable and Scott F. Smith. "Partial Objects In Constructive Type Theory". In: *Logic in Computer Science*. 1987.

[DP01] Rowan Davies and Frank Pfenning. "A Modal Analysis of Staged Computation". In: *J. ACM* 48.3 (May 2001), pp. 555–604. ISSN: 0004-5411. DOI: 10.1145/382780.382785.

[Gra+21] Daniel Gratzer et al. "Multimodal Dependent Type Theory". en. In: *Logical Methods in Computer Science* Volume 17, Issue 3 (July 2021), p. 7571. ISSN: 1860-5974. DOI: 10.46298/lmcs-17(3:11)2021. (Visited on 11/24/2022).

[Hof00] Martin Hofmann. "Safe recursion with higher types and BCK-algebra". en. In: *Annals of Pure and Applied Logic* 104.1-3 (July 2000). Number: 1-3, pp. 113–166. ISSN: 01680072. DOI: 10.1016/S0168-0072(00)00010-5. (Visited on 12/06/2022).

[Hof97] Martin Hofmann. "An Application of Category-Theoretic Semantics to the Characterisation of Complexity Classes Using Higher-Order Function Algebras". en. In: *Bulletin of Symbolic Logic* 3.4 (Dec. 1997). Number: 4, pp. 469–486. ISSN: 1079-8986, 1943-5894. DOI: 10.2307/421100. (Visited on 12/06/2022).

[Hof98] Martin Hofmann. "A mixed modal/linear lambda calculus with applications to bellantoni-cook safe recursion". en. In: *Computer Science Logic*. Ed. by Mogens Nielsen et al. Vol. 1414. Series Editors: _:n1470 Series Title: Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 275–294. ISBN: 978-3-540-64570-2 978-3-540-69353-6. DOI: 10.1007/BFb0028020. (Visited on 12/07/2022).

[HS20] Pieter Hofstra and Philip Scott. *Aspects of categorical recursion theory*. en. Issue: arXiv:2001.05778 arXiv:2001.05778 [math]. Jan. 2020. URL: http://arxiv.org/abs/2001.05778 (visited on 12/01/2022).

[Koh08] Ulrich Kohlenbach. *Applied Proof Theory: Proof Interpretations and their Use in Mathematics*. Springer Berlin, Heidelberg, 2008. ISBN: 978-3-540-77532-4. DOI: 10.1007/978-3-540-77533-1.

[LM94] Saunders Mac Lane and Ieke Moerdijk. *Sheaves in Geometry and Logic*. Springer New York, 1994. DOI: 10.1007/978-1-4612-0927-0.

[LSR17] Daniel R. Licata, Michael Shulman, and Mitchell Riley. "A Fibrational Framework for Substructural and Modal Logics". In: *International Conference on Formal Structures for Computation and Deduction*. 2017.

[Pau86] Lawrence C. Paulson. "Constructing recursion operators in intuitionistic type theory". In: *Journal of Symbolic Computation* 2.4 (1986), pp. 325–355. ISSN: 0747-7171. DOI: https://doi.org/10.1016/S0747-7171(86)80002-5.

[Ril22] Mitchell Riley. "A Bunched Homotopy Type Theory for Synthetic Stable Homotopy Theory". PhD Thesis. PhD Thesis, 2022.

[Sch14] Urs Schreiber. *Quantization via Linear homotopy types*. 2014. arXiv: 1402.7041 [math-ph].

[Ste21] Jonathan Sterling. "First Steps in Synthetic Tait Computability: The Objective Metatheory of Cubical Type Theory". Issue: CMU-CS-21-142. PhD Thesis. Carnegie Mellon University, 2021. DOI: 10.5281/zenodo.6990769.