Primitive Recursive Dependent Type Theory

LICS 2024

Johannes Schipp von Branitz[1]
Ulrik Buchholtz[2]

University of Nottingham

July 09, 2024

[1]https://jsvb.xyz
[2]https://ulrikbuchholtz.dk

Primitive Recursion

Reminder

The *basic primitive recursive functions* are constant functions, the successor function and projections of type $\mathbb{N}^n \to \mathbb{N}$. A *primitive recursive function* is obtained by finite applications of composition of the basic p.r. functions and the *primitive recursion operator*

$$\begin{aligned} \text{primrec} : \mathbb{N} \to (\mathbb{N} \times \mathbb{N} \to \mathbb{N}) \to \mathbb{N} \to \mathbb{N} \\ \text{primrec}(g, h, 0) = g \\ \text{primrec}(g, h, k+1) = h(k, \text{primrec}(g, h, k)). \end{aligned}$$

The *Ackermann function* $A : \mathbb{N} \to (\mathbb{N} \to \mathbb{N})$ given by

$$\begin{aligned} A(0) &= (n \mapsto n+1) \\ A(m+1) &= \begin{cases} 0 & \mapsto A(m, 1) \\ n+1 & \mapsto A(m, A(m+1, n)) \end{cases} \end{aligned}$$

grows faster than any p.r. function. It requires elimination into a function type.

Primitive Recursion

Reminder

The *basic primitive recursive functions* are constant functions, the successor function and projections of type $\mathbb{N}^n \to \mathbb{N}$. A *primitive recursive function* is obtained by finite applications of composition of the basic p.r. functions and the *primitive recursion operator*

$$\begin{aligned}\mathrm{primrec} : \mathbb{N} \to (\mathbb{N} \times \mathbb{N} \to \mathbb{N}) \to \mathbb{N} \to \mathbb{N}\\ \mathrm{primrec}(g, h, 0) &= g\\ \mathrm{primrec}(g, h, k+1) &= h(k, \mathrm{primrec}(g, h, k)).\end{aligned}$$

The *Ackermann function* $A : \mathbb{N} \to (\mathbb{N} \to \mathbb{N})$ given by

$$\begin{aligned}A(0) &= (n \mapsto n+1)\\ A(m+1) &= \begin{cases} 0 & \mapsto A(m,1)\\ n+1 & \mapsto A(m, A(m+1, n)) \end{cases}\end{aligned}$$

grows faster than any p.r. function. It requires elimination into a function type.

Primitive Recursion

Reminder

The *basic primitive recursive functions* are constant functions, the successor function and projections of type $\mathbb{N}^n \to \mathbb{N}$. A *primitive recursive function* is obtained by finite applications of composition of the basic p.r. functions and the *primitive recursion operator*

$$\begin{aligned} \mathrm{primrec} : \mathbb{N} \to (\mathbb{N} \times \mathbb{N} \to \mathbb{N}) \to \mathbb{N} \to \mathbb{N} \\ \mathrm{primrec}(g, h, 0) &= g \\ \mathrm{primrec}(g, h, k+1) &= h(k, \mathrm{primrec}(g, h, k)). \end{aligned}$$

The *Ackermann function* $A : \mathbb{N} \to (\mathbb{N} \to \mathbb{N})$ given by

$$\begin{aligned} A(0) &= (n \mapsto n+1) \\ A(m+1) &= \begin{cases} 0 & \mapsto A(m, 1) \\ n+1 & \mapsto A(m, A(m+1, n)) \end{cases} \end{aligned}$$

grows faster than any p.r. function. It requires elimination into a function type.

Primitive Recursive Dependent Type Theory

Main Theorem

Let T be a dependent type theory with

- dependent pair types $\sum_{a:A} B(a)$ and function types $\prod_{a:A} B(a)$,
- inductive identity types,
- a universe U_0 closed under Σ- and identity types (but not Π-types),
- a U_0-small closed type N with the standard elimination principle for natural numbers

$$\frac{n : \mathrm{N} \vdash X(n) : U_0 \qquad \vdash g : X(0) \qquad n : \mathrm{N}, x : X(n) \vdash h(n,x) : X(n+1)}{n : \mathrm{N} \vdash \mathrm{ind}_{g,h}(n) : X(n)}$$

  restricted to U_0-small type families $n : \mathrm{N} \vdash X(n) : U_0$,
- larger universes U_α closed under all type constructors,
- and the rule

$$\Pi_{n:\mathrm{N}} X(n) : U_1.$$

Then the definable terms

$$n : \mathrm{N} \vdash f(n) : \mathrm{N}$$

in T are exactly the primitive recursive functions.

Primitive Recursive Dependent Type Theory

Main Theorem

Let T be a dependent type theory with

- dependent pair types $\sum_{a:A} B(a)$ and function types $\prod_{a:A} B(a)$,
- inductive identity types,
- a universe U_0 closed under Σ- and identity types (but not Π-types),
- a U_0-small closed type N with the standard elimination principle for natural numbers

$$\frac{n : \mathrm{N} \vdash X(n) : U_0 \qquad \vdash g : X(0) \qquad n : \mathrm{N}, x : X(n) \vdash h(n, x) : X(n+1)}{n : \mathrm{N} \vdash \mathrm{ind}_{g,h}(n) : X(n)}$$

  restricted to U_0-small type families $n : \mathrm{N} \vdash X(n) : U_0$,
- larger universes U_α closed under all type constructors,
- and the rule

$$\Pi_{n:\mathrm{N}} X(n) : U_1.$$

Then the definable terms

$$n : \mathrm{N} \vdash f(n) : \mathrm{N}$$

in T are exactly the primitive recursive functions.

Let T be a dependent type theory with

- dependent pair types $\sum_{a:A} B(a)$ and function types $\prod_{a:A} B(a)$,
- inductive identity types,
- a universe U_0 closed under Σ- and identity types (but not Π-types),
- a U_0-small closed type N with the standard elimination principle for natural numbers

$$\frac{n : \mathrm{N} \vdash X(n) : U_0 \qquad \vdash g : X(0) \qquad n : \mathrm{N}, x : X(n) \vdash h(n,x) : X(n+1)}{n : \mathrm{N} \vdash \mathrm{ind}_{g,h}(n) : X(n)}$$

  restricted to U_0-small type families $n : \mathrm{N} \vdash X(n) : U_0$,
- larger universes U_α closed under all type constructors,
- and the rule

$$\Pi_{n:\mathrm{N}} X(n) : U_1.$$

Then the definable terms

$$n : \mathrm{N} \vdash f(n) : \mathrm{N}$$

in T are exactly the primitive recursive functions.

Primitive Recursive Dependent Type Theory

Main Theorem

Let T be a dependent type theory with

- dependent pair types $\sum_{a:A} B(a)$ and function types $\prod_{a:A} B(a)$,
- inductive identity types,
- a universe U_0 closed under Σ- and identity types (but not Π-types),
- a U_0-small closed type N with the standard elimination principle for natural numbers

$$\frac{n : \mathrm{N} \vdash X(n) : U_0 \qquad \vdash g : X(0) \qquad n : \mathrm{N}, x : X(n) \vdash h(n,x) : X(n+1)}{n : \mathrm{N} \vdash \mathrm{ind}_{g,h}(n) : X(n)}$$

  restricted to U_0-small type families $n : \mathrm{N} \vdash X(n) : U_0$,
- larger universes U_α closed under all type constructors,
- and the rule

$$\Pi_{n:\mathrm{N}} X(n) : U_1.$$

Then the definable terms

$$n : \mathrm{N} \vdash f(n) : \mathrm{N}$$

in T are exactly the primitive recursive functions.

Primitive Recursive Dependent Type Theory

Main Theorem

Let T be a dependent type theory with

- dependent pair types $\sum_{a:A} B(a)$ and function types $\prod_{a:A} B(a)$,
- inductive identity types,
- a universe U_0 closed under Σ- and identity types (but not Π-types),
- a U_0-small closed type N with the standard elimination principle for natural numbers

$$\frac{n : \mathrm{N} \vdash X(n) : U_0 \qquad \vdash g : X(0) \qquad n : \mathrm{N}, x : X(n) \vdash h(n,x) : X(n+1)}{n : \mathrm{N} \vdash \mathrm{ind}_{g,h}(n) : X(n)}$$

  restricted to U_0-small type families $n : \mathrm{N} \vdash X(n) : U_0$,
- larger universes U_α closed under all type constructors,
- and the rule

$$\Pi_{n:\mathrm{N}} X(n) : U_1.$$

Then the definable terms

$$n : \mathrm{N} \vdash f(n) : \mathrm{N}$$

in T are exactly the primitive recursive functions.

Let T be a dependent type theory with

- dependent pair types $\sum_{a:A} B(a)$ and function types $\prod_{a:A} B(a)$,
- inductive identity types,
- a universe U_0 closed under Σ- and identity types (but not Π-types),
- a U_0-small closed type N with the standard elimination principle for natural numbers

$$\frac{n : \mathrm{N} \vdash X(n) : U_0 \qquad \vdash g : X(0) \qquad n : \mathrm{N}, x : X(n) \vdash h(n,x) : X(n+1)}{n : \mathrm{N} \vdash \mathrm{ind}_{g,h}(n) : X(n)}$$

  restricted to U_0-small type families $n : \mathrm{N} \vdash X(n) : U_0$,
- larger universes U_α closed under all type constructors,
- and the rule

$$\Pi_{n:\mathrm{N}} X(n) : U_1.$$

Then the definable terms

$$n : \mathrm{N} \vdash f(n) : \mathrm{N}$$

in T are exactly the primitive recursive functions.

Let T be a dependent type theory with

- dependent pair types $\sum_{a:A} B(a)$ and function types $\prod_{a:A} B(a)$,
- inductive identity types,
- a universe U_0 closed under Σ- and identity types (but not Π-types),
- a U_0-small closed type N with the standard elimination principle for natural numbers

$$\frac{n : \mathrm{N} \vdash X(n) : \mathrm{U}_0 \qquad \vdash g : X(0) \qquad n : \mathrm{N}, x : X(n) \vdash h(n,x) : X(n+1)}{n : \mathrm{N} \vdash \mathrm{ind}_{g,h}(n) : X(n)}$$

  restricted to U_0-small type families $n : \mathrm{N} \vdash X(n) : \mathrm{U}_0$,
- larger universes U_α closed under all type constructors,
- and the rule

$$\Pi_{n:\mathrm{N}} X(n) : \mathrm{U}_1.$$

Then the definable terms

$$n : \mathrm{N} \vdash f(n) : \mathrm{N}$$

in T are exactly the primitive recursive functions.

Motivation for Conservative Extension

- Variants of primitive recursion are used as base theory for reverse mathematics in which theorems are encoded in a weak base system
- More expressive base system means less encoding
- Syntax is closer to proof assistants enabling formal verification

Motivation for Conservative Extension

- Variants of primitive recursion are used as base theory for reverse mathematics in which theorems are encoded in a weak base system
- More expressive base system means less encoding
- Syntax is closer to proof assistants enabling formal verification

Motivation for Conservative Extension

- Variants of primitive recursion are used as base theory for reverse mathematics in which theorems are encoded in a weak base system
- More expressive base system means less encoding
- Syntax is closer to proof assistants enabling formal verification

Potential Further Extensions

- Finitary inductive types and type families, finitary induction-recursion, e.g. lists
- Primitive recursive universe of types – judgemental variant of internal p.r. Gödel encoding of the codes in U_0
- Primitive Recursive Homotopy/Cubical Type Theory – not clear how to adapt our adequacy proof

Potential Further Extensions

- Finitary inductive types and type families, finitary induction-recursion, e.g. lists
- Primitive recursive universe of types – judgemental variant of internal p.r. Gödel encoding of the codes in U_0
- Primitive Recursive Homotopy/Cubical Type Theory – not clear how to adapt our adequacy proof

Potential Further Extensions

- Finitary inductive types and type families, finitary induction-recursion, e.g. lists
- Primitive recursive universe of types – judgemental variant of internal p.r. Gödel encoding of the codes in U_0
- Primitive Recursive Homotopy/Cubical Type Theory – not clear how to adapt our adequacy proof

Proof of Conservativity by Logical Relations

Ingredients

- Synthetic Tait Computability
- Standard model

$$[\![-]\!]_{\mathrm{Set}} : \mathrm{T} \to \mathrm{Set}$$

with

$$[\![\mathrm{N}]\!]_{\mathrm{Set}} = \mathbb{N}$$

- Model

$$[\![-]\!]_{\mathcal{R}} : \mathrm{T} \to \mathcal{R}$$

in a topos of sheaves on a category of arities and p.r. functions with coverage generated by finite jointly surjective families, with the property that

$$\mathcal{R}([\![\mathrm{N}]\!]_{\mathcal{R}}, [\![\mathrm{N}]\!]_{\mathcal{R}})$$

are exactly the primitive recursive functions $\mathbb{N} \to \mathbb{N}$.

Proof of Conservativity by Logical Relations

Ingredients

- Synthetic Tait Computability
- Standard model

$$[\![-]\!]_{\mathrm{Set}} : \mathrm{T} \to \mathrm{Set}$$

with

$$[\![\mathrm{N}]\!]_{\mathrm{Set}} = \mathbb{N}$$

- Model

$$[\![-]\!]_{\mathcal{R}} : \mathrm{T} \to \mathcal{R}$$

in a topos of sheaves on a category of arities and p.r. functions with coverage generated by finite jointly surjective families, with the property that

$$\mathcal{R}([\![\mathrm{N}]\!]_{\mathcal{R}}, [\![\mathrm{N}]\!]_{\mathcal{R}})$$

are exactly the primitive recursive functions $\mathbb{N} \to \mathbb{N}$.

Proof of Conservativity by Logical Relations

Ingredients

- Synthetic Tait Computability
- Standard model

  $$[\![-]\!]_{\mathrm{Set}} : \mathrm{T} \to \mathrm{Set}$$

  with

  $$[\![\mathrm{N}]\!]_{\mathrm{Set}} = \mathbb{N}$$

- Model

  $$[\![-]\!]_{\mathcal{R}} : \mathrm{T} \to \mathcal{R}$$

  in a topos of sheaves on a category of arities and p.r. functions with coverage generated by finite jointly surjective families, with the property that

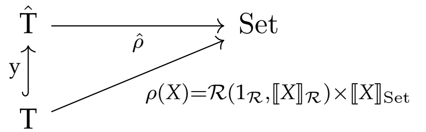
  $$\mathcal{R}([\![\mathrm{N}]\!]_{\mathcal{R}}, [\![\mathrm{N}]\!]_{\mathcal{R}})$$

  are exactly the primitive recursive functions $\mathbb{N} \to \mathbb{N}$.

Proof of Conservativity by Logical Relations

Combining the Data

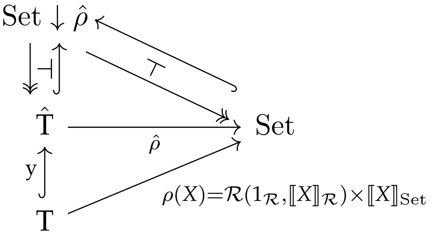
Using the global sections functor $\mathcal{R}(1_{\mathcal{R}}, -)$ we can combine the data of both models into a single functor which we extend along the Yoneda embedding.



Proof of Conservativity by Logical Relations

Form the Artin Gluing

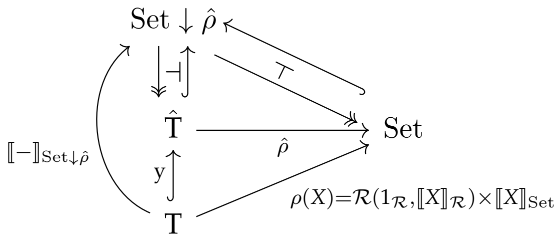
The comma category $\mathrm{Set} \downarrow \hat{\rho}$ is a topos with an open geometric embedding $\hat{\mathrm{T}} \hookrightarrow \mathrm{Set} \downarrow \hat{\rho}$ and closed geometric embedding $\mathrm{Set} \hookrightarrow \mathrm{Set} \downarrow \hat{\rho}$.



Proof of Conservativity by Logical Relations

Synthetically Define a Logical Relation

Use the internal language of the topos $\mathrm{Set} \downarrow \hat{\rho}$ to construct another model $[\![-]\!]_{\mathrm{Set}\downarrow\hat{\rho}}$ which assigns computability predicates to objects $\rho(X)$.



Proof of Conservativity by Logical Relations

Externalise

Any term

$$n : \mathrm{N} \vdash f(n) : \mathrm{N}$$

of T is interpreted in $\mathrm{Set} \downarrow \hat{\rho}$ as

$$\begin{array}{ccc}
\mathbb{N} \cong \mathrm{T}(1_{\mathrm{T}}, \mathrm{N}) & \xrightarrow{\mathrm{T}(1_{\mathrm{T}}, f)} & \mathrm{T}(1_{\mathrm{T}}, \mathrm{N}) \cong \mathbb{N} \\
\Big\downarrow \Delta_{\mathrm{N}} & [\![f]\!]_{\mathrm{Set}\downarrow\hat{\rho}} & \Big\downarrow \Delta_{\mathrm{N}} \\
\mathbb{N} \times \mathbb{N} \cong \rho(\mathrm{N}) & \xrightarrow[\mathcal{R}(1_{\mathcal{R}}, \widehat{[\![f]\!]}_{\mathcal{R}}) \times \widehat{[\![f]\!]}_{\mathrm{Set}}]{} & \rho(\mathrm{N}) \cong \mathbb{N} \times \mathbb{N}.
\end{array}$$

Since $\widehat{[\![f]\!]}_{\mathcal{R}}$ is primitive recursive, so is $[\![f]\!]_{\mathrm{Set}}$.

$\square$