## Towards guarded recursion in HoTT

Rasmus Ejlers Møgelberg

IT University of Copenhagen

September 25, 2015

◆□▶ ◆□▶ ◆臣▶ ◆臣▶ 臣 の�?

#### Overview

- Guarded recursion applications
  - Computing with streams
  - Modelling programming languages
- Model: the topos of trees
- Intensional models
- Coinduction via guarded recursion
- Problems combining HoTT and GR

## Motivation

◆□ ▶ < 圖 ▶ < 圖 ▶ < 圖 ▶ < 圖 • 의 Q @</p>

#### Motivation 1: Computing with streams

• Which of the following streams are well-defined:

zeros = 0::zeros xs = xs ys = 0::(tail(ys)) nats = 0::(map (+1) nats)

◆□▶ ◆□▶ ◆三▶ ◆三▶ 三三 のへぐ

Capturing productivity in types

Introduce modal operator ►

$$\begin{split} & \texttt{S(int)} = \mu\texttt{X}.\texttt{int} \times \blacktriangleright \texttt{X} \\ & \texttt{hd:} \texttt{S(int)} \to \texttt{int} \\ & \texttt{tail:} \texttt{S(int)} \to \blacktriangleright \texttt{S(int)} \\ & \texttt{cons:} \texttt{int} \times \blacktriangleright \texttt{S(int)} \to \texttt{S(int)} \end{split}$$

Fixed points

$$ext{fix:} (\blacktriangleright S( ext{int}) 
ightarrow S( ext{int})) 
ightarrow S( ext{int})$$
  
zeros =  $ext{fix}(\lambda xs.0::xs)$ 

Capturing productivity in types

• • is an applicative functor

$$\texttt{next: } X \to \blacktriangleright X$$
$$\circledast: \blacktriangleright (X \to Y) \to \blacktriangleright X \to \blacktriangleright Y$$

• Typing nats

nats = fix( $\lambda$ xs.0::(next(map (+1))  $\circledast$  xs))

• Fixed point property

$$\begin{aligned} \text{fix:} \left(\blacktriangleright \text{S(int)} \rightarrow \text{S(int)}\right) \rightarrow \text{S(int)} \\ \text{fix(f)} &= \text{f(next(fix(f)))} \end{aligned}$$

Motivation 2: Modelling higher-order store

• Would like to solve (but can not)

$$\mathcal{W} = \mathcal{N} \rightarrow_{\mathrm{fin}} \mathcal{T} \qquad \qquad \mathcal{T} = \mathcal{W} \rightarrow_{\mathrm{mon}} \mathcal{P}(\mathrm{Value})$$

Suffices to solve this equation

$$\widehat{\mathcal{T}}\cong \blacktriangleright((N\to_{\mathrm{fin}}\widehat{\mathcal{T}})\to_{\mathrm{mon}}\mathcal{P}(\mathrm{Value}))$$

- Can model higher-order store in expressive type theory with guarded recursion
- Synthetic presentation of step-indexed model!
- (Birkedal, M. et al LICS 2011)

Synthetic guarded domain theory

• Lifting monad

$$LX \cong X + \blacktriangleright LX$$

Model of PCF in type theory with guarded recursion

$$Y M \Downarrow^{k+1} v =_{\mathrm{def}} \blacktriangleright (M(Y M) \Downarrow^k v)$$

- Proved (intensional) computational adequacy
- (Paviotti, M, Birkedal, MFPS 2015)

## The topos of trees

The topos of trees (**Set**<sup> $\omega^{op}$ </sup>)

• Objects

$$X(1) \xleftarrow{r_1} X(2) \xleftarrow{r_2} X(3) \longleftarrow \dots$$

• Example: object of streams of integers S(int)

$$\mathbb{Z} \stackrel{\pi}{\longleftarrow} \mathbb{Z}^2 \stackrel{\pi}{\longleftarrow} \mathbb{Z}^3 \stackrel{\pi}{\longleftarrow} \dots$$

◆□▶ ◆□▶ ◆三▶ ◆三▶ 三三 のへぐ

The topos of trees (**Set**<sup> $\omega^{op}$ </sup>)

Objects

$$X(1) \xleftarrow{r_1} X(2) \xleftarrow{r_2} X(3) \longleftarrow \dots$$

• Example: object of streams of integers S(int)

$$\mathbb{Z} \stackrel{\pi}{\longleftarrow} \mathbb{Z}^2 \stackrel{\pi}{\longleftarrow} \mathbb{Z}^3 \stackrel{\pi}{\longleftarrow} \dots$$

• Define ► X

$$\{*\} \longleftarrow X(1) \longleftarrow X(2) \longleftarrow \ldots$$

• Note that  $S(int) \cong \mathbb{Z} \times \triangleright S(int)$ :

$$\mathbb{Z} \times 1 \stackrel{\mathbb{Z} \times !}{\longleftarrow} \mathbb{Z} \times \mathbb{Z} \stackrel{\mathbb{Z} \times \pi}{\longleftarrow} \mathbb{Z} \times \mathbb{Z}^2 \stackrel{\mathbb{Z} \times \pi}{\longleftarrow} \dots$$

◆□▶ ◆□▶ ◆三▶ ◆三▶ 三三 のへぐ

Construction of fixed points

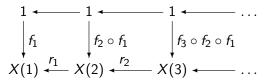
• Given 
$$f : \triangleright X \to X$$
:

$$\{*\} \longleftarrow X(1) \xleftarrow{r_1} X(2) \longleftarrow \dots$$

$$f_1 \downarrow \qquad f_2 \downarrow \qquad f_3 \downarrow$$

$$X(1) \xleftarrow{r_1} X(2) \xleftarrow{r_2} X(3) \longleftarrow \dots$$

• Construct  $fix_X(f) : 1 \to X$ :



Fixed points are unique

Universe closed under ►

$$\frac{\Gamma \vdash A : U}{\Gamma \vdash \rhd A : U}$$
$$\mathrm{El}(\rhd(A)) = \blacktriangleright \mathrm{El}(A)$$

◆□▶ ◆□▶ ◆臣▶ ◆臣▶ 臣 のへぐ

Universe closed under ►

 $\frac{\Gamma \vdash A : \blacktriangleright U}{\Gamma \vdash \rhd A : U}$ El( $\rhd$ (next(A))) =  $\blacktriangleright$  El(A)

◆□▶ ◆□▶ ◆臣▶ ◆臣▶ 臣 のへぐ

Universe closed under ►

 $\frac{\Gamma \vdash A : \blacktriangleright U}{\Gamma \vdash \rhd A : U}$ 

$$\operatorname{El}(\rhd(\operatorname{next}(A))) = \blacktriangleright \operatorname{El}(A)$$

• Type of streams as fixed point for universe map

$$S(int) = fix(\lambda X : \blacktriangleright U.\mathbb{Z} \times \rhd X)$$

Then

$$\begin{split} \mathrm{El}(\mathtt{S(int)}) &= \mathrm{El}(\mathbb{Z} \times \triangleright(\mathrm{next}(\mathtt{S(int)}))) \\ &= \mathbb{Z} \times \blacktriangleright \mathrm{El}(\mathtt{S(int)}) \end{split}$$

◆□▶ ◆□▶ ◆三▶ ◆三▶ 三三 のへぐ

Universe closed under ►

 $\frac{\Gamma \vdash A : \blacktriangleright U}{\Gamma \vdash \triangleright A : U}$ 

$$\operatorname{El}(\rhd(\operatorname{next}(A))) = \blacktriangleright \operatorname{El}(A)$$

• Type of streams as fixed point for universe map

$$S(int) = fix(\lambda X : \blacktriangleright U.\mathbb{Z} \times \rhd X)$$

Then

$$\operatorname{El}(\operatorname{S(int)}) = \operatorname{El}(\mathbb{Z} \times \triangleright(\operatorname{next}(\operatorname{S(int)})))$$
  
=  $\mathbb{Z} \times \blacktriangleright \operatorname{El}(\operatorname{S(int)})$ 

Model using Hofmann-Streicher universe construction

## Intensional models

◆□ ▶ < 圖 ▶ < 圖 ▶ < 圖 ▶ < 圖 • 의 Q @</p>

## Intensional models

• **Theorem** (Shulman). If  $\mathbb{C}$  is a model of intensional type theory, so is  $\mathbb{C}^{\omega^{\mathrm{op}}}$ . Univalent universes in  $\mathbb{C}$  lift to univalent universes in  $\mathbb{C}^{\omega^{\mathrm{op}}}$  by Hofmann-Streicher

・ロト・日本・モート モー うへぐ

#### Intensional models

- Theorem (Shulman). If C is a model of intensional type theory, so is C<sup>ω<sup>op</sup></sup>. Univalent universes in C lift to univalent universes in C<sup>ω<sup>op</sup></sup> by Hofmann-Streicher
- Model construction uses Reedy model structure on  $\mathbb{C}^{\omega^{\mathrm{op}}}$
- Closed types are sequences

$$A(1) \stackrel{r_1^A}{\longleftarrow} A(2) \stackrel{r_2^A}{\longleftarrow} A(3) \stackrel{r_3^A}{\longleftarrow} A(4) \longleftarrow \dots$$

where each  $r_n^A$  is a fibration

i.e., closed types modelled as infinite contexts

Uniqueness of fixed points, propositionally

- Theorem.  $\mathbb{C}^{\omega^{\mathrm{op}}}$  models guarded recursion plus uniqueness of fixed points

 $\frac{\Gamma \vdash f : \blacktriangleright A \to A \quad \Gamma \vdash M : A \quad \Gamma \vdash p : \mathrm{Id}_A(M, f(\mathrm{next}(M)))}{\Gamma \vdash \mathrm{UFP}(p) : \mathrm{Id}_A(M, \mathrm{fix}(f))}$ 

• Univalence plus UFP implies

$$(F(\operatorname{next}(A)) \simeq A) \longleftrightarrow (A \simeq \operatorname{fix}(F))$$

- for  $F : \triangleright U \to U$
- (Birkedal and M, LICS 2013)

Coinductive types via guarded recursive types

<□ > < @ > < E > < E > E のQ @

## A problem

$$\begin{split} & \texttt{S(int)} = \mu\texttt{X}.\texttt{int} \times \blacktriangleright \texttt{X} \\ & \texttt{hd:} \texttt{S(int)} \to \texttt{int} \\ & \texttt{tail:} \texttt{S(int)} \to \blacktriangleright \texttt{S(int)} \\ & \texttt{cons:} \texttt{int} \times \blacktriangleright \texttt{S(int)} \to \texttt{S(int)} \\ & \texttt{next:} \texttt{X} \to \blacktriangleright \texttt{X} \\ & \circledast: \blacktriangleright (\texttt{X} \to \texttt{Y}) \to \blacktriangleright \texttt{X} \to \blacktriangleright \texttt{Y} \end{split}$$

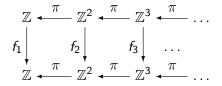
• Computing the second element

snd =  $(\lambda xs.next(hd) \circledast tail(xs)): S(int) \rightarrow \blacktriangleright int$ 

• Can not get rid of ▶!

Guarded recursion vs coinduction in model

• All maps  $f: S(int) \rightarrow S(int)$  are causal



 Observation: Limit of guarded recursive streams is set of real (coinductive) streams!

$$\mathbb{Z} \stackrel{\pi}{\longleftarrow} \mathbb{Z}^2 \stackrel{\pi}{\longleftarrow} \mathbb{Z}^3 \stackrel{\pi}{\longleftarrow} \dots$$

#### Syntax: multiple clocks

- Idea originally due to Atkey and McBride
- Clock variable context  $\Delta = \kappa_1, \ldots, \kappa_n$

$$\frac{\Gamma \vdash_{\Delta} A : \mathsf{Type} \qquad \vdash_{\Delta} \kappa}{\Gamma \vdash_{\Delta} \overset{\kappa}{\blacktriangleright} A : \mathsf{Type}}$$
  
fix<sup>\kappa</sup> : (\vec{\kappa} X \to X) \to X

◆□▶ ◆□▶ ◆臣▶ ◆臣▶ 臣 のへぐ

etc

## Universal quantification over clocks

$$\begin{array}{c} \Gamma \vdash_{\Delta,\kappa} A : \mathsf{Type} \quad \kappa \notin \mathsf{fc}(\Gamma) \\ \hline \Gamma \vdash_{\Delta} \forall \kappa.A : \mathsf{Type} \\ \hline \Gamma \vdash_{\Delta,\kappa} t : A \quad \kappa \notin \mathsf{fc}(\Gamma) \\ \hline \Gamma \vdash_{\Delta} \Lambda \kappa.t : \forall \kappa.A \\ \hline \Gamma \vdash_{\Delta} t : \forall \kappa.A \quad \vdash_{\Delta} \kappa' \\ \hline \Gamma \vdash_{\Delta} t[\kappa'] : A[\kappa'/\kappa] \end{array}$$

• Clock quantification is right adjoint to clock weakening

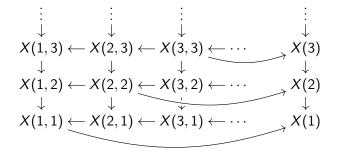
・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・
 ・

#### An extensional model (Bizjak and M, MFPS 2015)

- Type in context  $\Delta = \emptyset$  is a set
- Type in context  $\Delta = \kappa$  is an object in the topos of trees

$$X(1) \xleftarrow{r_1} X(2) \xleftarrow{r_2} X(3) \longleftarrow \dots$$

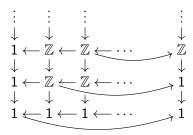
• Type in context  $\Delta = \kappa, \kappa'$  is a diagram of form



◆□▶ ◆□▶ ◆臣▶ ◆臣▶ = 臣 = のへで

An extensional model (Bizjak and M, MFPS 2015)

• Interpretation of  $\mathbf{b}^{\kappa} \mathbf{b}^{\kappa'}$  int



(日)、

< ∃⇒

э

#### Universes

Model forces universes to be indexed by sets of clocks

 $\frac{\Delta'\subseteq\Delta}{\Gamma\vdash_{\Delta}\mathrm{U}_{\Delta'}:\mathsf{Type}}$ 

- E.g.  $\llbracket \vdash_{\kappa} U_{\emptyset} :$  Type]] universe of small constant presheaves in  $\mathbf{Set}^{\omega^{\mathrm{op}}}$  ,
- and  $\llbracket \vdash_{\kappa} U_{\kappa} : Type \rrbracket$  universe of all small presheaves in **Set**<sup> $\omega^{op}$ </sup>

◆□▶ ◆□▶ ◆臣▶ ◆臣▶ 臣 の�?

#### Universes

Model forces universes to be indexed by sets of clocks

 $\frac{\Delta'\subseteq\Delta}{\Gamma\vdash_{\Delta}\mathrm{U}_{\Delta'}:\mathsf{Type}}$ 

- E.g.  $\llbracket \vdash_{\kappa} U_{\emptyset}$  : Type] universe of small constant presheaves in Set<sup> $\omega^{op}$ </sup>,
- and  $\llbracket \vdash_{\kappa} U_{\kappa} : Type \rrbracket$  universe of all small presheaves in **Set**<sup> $\omega^{op}$ </sup>
- Implicit inclusion of universes

$$\frac{\Delta'' \subseteq \Delta' \subseteq \Delta \quad \Gamma \vdash_{\Delta} A : U_{\Delta''}}{\Gamma \vdash_{\Delta} A : U_{\Delta'}}$$

 Requires universe inclusion to commute with operations on universe

# Universes in $\mathbf{Set}^{\omega^{\mathrm{op}}}$

- Assume given universe U in **Set**
- Construct universes  $V_{\emptyset}$ ,  $V_{\kappa}$  in  $\mathbf{Set}^{\omega^{\mathrm{op}}}$

$$V_{\emptyset} = U \longleftarrow U \longleftarrow U \longleftarrow U \longleftarrow \dots$$
  

$$i \downarrow \qquad i_1 \downarrow \qquad i_2 \downarrow \qquad i_3 \downarrow \qquad i_4 \downarrow$$
  

$$V_{\kappa} = V_{\kappa}(1) \longleftarrow V_{\kappa}(2) \longleftarrow V_{\kappa}(3) \longleftarrow V_{\kappa}(4) \longleftarrow \dots$$

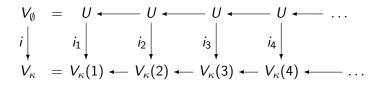
• Where 
$$V_\kappa(1) = U$$

$$V_{\kappa}(n+1) = \{X_1 \xleftarrow{f_1} X_2 \dots \xleftarrow{f_n} X_{n+1} \mid \forall i. X_i \in U\}$$
$$i_n(X) = (X \xleftarrow{id} X \dots \xleftarrow{id} X)$$

◆□▶ ◆□▶ ◆臣▶ ◆臣▶ 臣 の�?

• ( $V_{\kappa}$  is Hofmann-Streicher universe)

## Universes in $\mathbf{Set}^{\omega^{\mathrm{op}}}$



- Problem: *i* does not commute with forming function spaces.
- Solution: Quotient each V<sub>κ</sub>(n) by equivalence classes ensuring uniqueness of codes

- Requires choice in meta logic
- Will probably not work for intensional models

## Conclusions

- · Guarded recursive types useful for
  - modelling advanced programming languages: higher order store, non-determinism, concurrency etc
  - computing with streams (encoding productivity in types)
- Would like to combine this with HoTT
  - Expressive type theory for formalising programming language models

- Better treatment of coinductive types in HoTT
- Topos of trees model exists in intensional variant
- Not so clear how to model clocks in intensional type theory

# Thanks!

▲□▶▲圖▶▲≣▶▲≣▶ ■ のへの