# Denotational Semantics of recursive types in Synthetic Guarded Domain Theory

Marco Paviotti

IT University of Copenhagen

July 6th, 2016

Joint work with Rasmus Ejlers Møgelberg

# Guarded Type theory

- is a type theory with a time modality $\blacktriangleright$ pronounced "later"

$$\mathrm{fix} : (\blacktriangleright X \to X) \to X$$
$$\mathrm{fix}(f) = f(\mathrm{next}(\mathrm{fix}(f)))$$
$$\mathrm{next} : X \to \blacktriangleright X$$

# Guarded Type theory

- is a type theory with a time modality $\blacktriangleright$ pronounced "later"

$$\begin{aligned} \mathrm{fix} &: (\blacktriangleright X \to X) \to X \\ \mathrm{fix}(f) &= f(\mathrm{next}(\mathrm{fix}(f))) \\ \mathrm{next} &: X \to \blacktriangleright X \end{aligned}$$

- can solve type equations as long as the $\blacktriangleright$ modality guards the recursive variable

$$X \cong \blacktriangleright((N \to X) \to 2)$$

- It is an abstract form of step-indexing

# Guarded Type theory

- is a type theory with a time modality $\blacktriangleright$ pronounced "later"

$$\begin{aligned} \mathrm{fix} &: (\blacktriangleright X \to X) \to X \\ \mathrm{fix}(f) &= f(\mathrm{next}(\mathrm{fix}(f))) \\ \mathrm{next} &: X \to \blacktriangleright X \end{aligned}$$

- can solve type equations as long as the $\blacktriangleright$ modality guards the recursive variable

$$X \cong \blacktriangleright((N \to X) \to 2)$$

- It is an abstract form of step-indexing
- The topos of trees $\mathbf{Set}^{\omega^{\mathrm{op}}}$ is a model for Guarded Type Theory

# Guarded Type Theory

- Useful for checking *productivity*, e.g. Guarded Streams

$$\mathsf{Str}_A^g \cong A \times \blacktriangleright \mathsf{Str}_A^g$$

- *Proving properties* about guarded recursive definitions, $\Pi$, $\Sigma$
- Operational reasoning for languages with advanced features
  - $\mu\alpha.\tau$,ref $\sigma$, non-determinism, shared-memory concurrency
  - using an abstract form of step-indexing

# Goal: denotational semantics in Guarded Type Theory

- Denotational semantics of recursion under the slogan

  Recursion in Guarded Recursion

- Previous: Recursion on terms (PCF) [MFPS '15]
- This talk : Recursive Types (FPC)

$$\frac{\Gamma \vdash M : \mu\alpha.\tau}{\Gamma \vdash \texttt{unfold}\ M : \tau[\mu\alpha.\tau/\alpha]} \qquad \frac{\Gamma \vdash M : \tau[\mu\alpha.\tau/\alpha]}{\Gamma \vdash \texttt{fold}\ M : \mu\alpha.\tau}$$

- The development is entirely inside the type theory
  - Operational semantics, Denotational semantics and proof of adequacy

# Denotational semantics in Guarded Type theory

Types as domains

- both synthetic and type theoretic approach
- Homotopy Theory to HoTT as Domain Theory to GdTT [1]
- more abstract representations and easier proofs

A small price to pay is intentionality

- the model counts steps
- we define a logical relation to prove extensional computational adequacy

[1] Guarded dependent Type Theory with Coinductive Types, FoSSaCS, 2016

# FPC in guarded type theory

# Big-step operational semantics

- Big-step call-by-name operational semantics encoded as an inductive type of the form $M \Downarrow^k v$
- Such that

$$\texttt{unfold}\,(\texttt{fold}\,(M)) \Downarrow^k v = \blacktriangleright(M \Downarrow^{k-1} v)$$

# Denotational semantics

# Lifting monad

$$LA =_{\mathrm{def}} A + \blacktriangleright LA$$

- $\eta : A \to LA$
- $\theta : \blacktriangleright LA \to LA$
- $\delta = \Theta \circ \mathrm{next} : LA \to LA$
- Satisfies $\bot = \delta(\bot)$ where $\bot = \mathrm{fix}(\theta)$.

## Related

Our is a guarded recursive variant of Capretta's Coinductive lifting monad

$$LA \cong A + LA$$

# Denotational Semantics – Types

- Define

$$[\![\Theta \vdash \tau]\!] : U^{|\Theta|} \to U$$

- By

$$\begin{aligned}
[\![\Theta \vdash \alpha]\!](\rho) &=_{\text{def}} \rho(\alpha) \\
[\![\Theta \vdash 1]\!](\rho) &=_{\text{def}} L1 \\
&\dots \\
[\![\Theta \vdash \tau_1 + \tau_2]\!](\rho) &=_{\text{def}} L([\![\Theta \vdash \tau_1]\!](\rho) + [\![\Theta \vdash \tau_2]\!](\rho)) \\
[\![\Theta \vdash \mu\alpha.\tau]\!](\rho) &=_{\text{def}} \blacktriangleright([\![\Theta \vdash \tau]\!](\rho, [\![\mu\alpha.\tau]\!](\rho)))
\end{aligned}$$

- **Theorem.** $[\![\Theta \vdash \mu\alpha.\tau]\!](\rho) = \blacktriangleright([\![\Theta \vdash \tau[\mu\alpha.\tau/\alpha]]\!](\rho))$
- $\Theta_\tau : \blacktriangleright[\![\tau]\!] \to [\![\tau]\!]$
- $\delta_\tau : [\![\tau]\!] \to [\![\tau]\!]$ defined as $\delta_\tau = \Theta_\tau \circ \text{next}$

# Denotational Semantics

- Interpretation of terms

$$[\![x_1 : A_1, \ldots, x_1 : A_1 \vdash t : B]\!] : [\![A_1]\!] \times \cdots \times [\![A_n]\!] \to [\![B]\!]$$

- Case of the folding and unfolding operations

$$[\![\Gamma \vdash \texttt{fold}\ M]\!](\gamma) = \mathrm{next}([\![M]\!](\gamma))$$

$$[\![\Gamma \vdash \texttt{unfold}\ M]\!](\gamma) = \theta_{\tau[\mu\alpha.\tau/\alpha]}([\![M]\!](\gamma))$$

## Recall

$$[\![\Theta \vdash \mu\alpha.\tau]\!](\rho) = \blacktriangleright([\![\Theta \vdash \tau[\mu\alpha.\tau/\alpha]]\!](\rho))$$

# Denotational Semantics

- Interpretation of terms

$$[\![x_1 : A_1, \ldots, x_1 : A_1 \vdash t : B]\!] : [\![A_1]\!] \times \cdots \times [\![A_n]\!] \to [\![B]\!]$$

- Case of the folding and unfolding operations

$$[\![\Gamma \vdash \texttt{fold}\ M]\!](\gamma) = \mathrm{next}([\![M]\!](\gamma))$$

$$[\![\Gamma \vdash \texttt{unfold}\ M]\!](\gamma) = \theta_{\tau[\mu\alpha.\tau/\alpha]}([\![M]\!](\gamma))$$

## Recall

$$[\![\Theta \vdash \mu\alpha.\tau]\!](\rho) = \blacktriangleright([\![\Theta \vdash \tau[\mu\alpha.\tau/\alpha]]\!](\rho))$$

$[\![M]\!](\gamma)) : [\![\tau[\mu\alpha.\tau/\alpha]]\!]$

# Denotational Semantics

- Interpretation of terms

$$[\![x_1 : A_1, \ldots, x_1 : A_1 \vdash t : B]\!] : [\![A_1]\!] \times \cdots \times [\![A_n]\!] \to [\![B]\!]$$

- Case of the folding and unfolding operations

$$[\![\Gamma \vdash \texttt{fold}\ M]\!](\gamma) = \mathrm{next}([\![M]\!](\gamma))$$

$$[\![\Gamma \vdash \texttt{unfold}\ M]\!](\gamma) = \theta_{\tau[\mu\alpha.\tau/\alpha]}([\![M]\!](\gamma))$$

**Recall**

$$[\![\Theta \vdash \mu\alpha.\tau]\!](\rho) = \blacktriangleright([\![\Theta \vdash \tau[\mu\alpha.\tau/\alpha]]\!](\rho))$$

$[\![M]\!](\gamma)) : [\![\mu\alpha.\tau]\!]$

# Denotational Semantics

- Interpretation of terms

$$[\![x_1 : A_1, \ldots, x_1 : A_1 \vdash t : B]\!] : [\![A_1]\!] \times \cdots \times [\![A_n]\!] \to [\![B]\!]$$

- Case of the folding and unfolding operations

$$[\![\Gamma \vdash \texttt{fold}\ M]\!](\gamma) = \mathrm{next}([\![M]\!](\gamma))$$

$$[\![\Gamma \vdash \texttt{unfold}\ M]\!](\gamma) = \theta_{\tau[\mu\alpha.\tau/\alpha]}([\![M]\!](\gamma))$$

# Denotational Semantics

- Interpretation of terms

$$[\![x_1 : A_1, \ldots, x_1 : A_1 \vdash t : B]\!] : [\![A_1]\!] \times \cdots \times [\![A_n]\!] \to [\![B]\!]$$

- Case of the folding and unfolding operations

$$[\![\Gamma \vdash \texttt{fold}\ M]\!](\gamma) = \mathrm{next}([\![M]\!](\gamma))$$

$$[\![\Gamma \vdash \texttt{unfold}\ M]\!](\gamma) = \theta_{\tau[\mu\alpha.\tau/\alpha]}([\![M]\!](\gamma))$$

## Note

**Lemma.** $[\![\texttt{unfold}\ (\texttt{fold}\ M)]\!] = \delta[\![M]\!]$

# Computational adequacy

# Adequacy

**Theorem.** If $\vdash M : \mathbf{nat}$ then $M \Downarrow^k v \iff [\![M]\!] = \delta^k([\![v]\!])$

# Adequacy

**Theorem.** If $\vdash M : \mathbf{nat}$ then $M \Downarrow^k v \iff [\![M]\!] = \delta^k([\![v]\!])$

- If $M = \mathtt{unfold}\,(\mathtt{fold}\,w)$ and $w \neq v$ the following holds

$$[\![\mathtt{unfold}\,(\mathtt{fold}\,w)]\!] = \delta([\![v]\!])$$

# Adequacy

**Theorem.** If $\vdash M : \mathbf{nat}$ then $M \Downarrow^k v \iff [\![M]\!] = \delta^k([\![v]\!])$

- If $M = \mathtt{unfold}\,(\mathtt{fold}\,w)$ and $w \neq v$ the following holds

$$[\![\mathtt{unfold}\,(\mathtt{fold}\,w)]\!] = \delta([\![v]\!])$$

- as it is logically equivalent to $\blacktriangleright([\![w]\!] = [\![v]\!])$

# Adequacy

**Theorem.** If $\vdash M : \mathbf{nat}$ then $M \Downarrow^k v \iff [\![M]\!] = \delta^k([\![v]\!])$

- If $M = \mathtt{unfold}\,(\mathtt{fold}\,w)$ and $w \neq v$ the following holds

$$[\![\mathtt{unfold}\,(\mathtt{fold}\,w)]\!] = \delta([\![v]\!])$$

- as it is logically equivalent to $\blacktriangleright([\![w]\!] = [\![v]\!])$
- So also need

$$\mathtt{unfold}\,(\mathtt{fold}\,w) \Downarrow^1 v$$

# Adequacy

**Theorem.** If $\vdash M : \mathbf{nat}$ then $M \Downarrow^k v \iff [\![M]\!] = \delta^k([\![v]\!])$

- If $M = \mathtt{unfold}\,(\mathtt{fold}\,w)$ and $w \neq v$ the following holds

$$[\![\mathtt{unfold}\,(\mathtt{fold}\,w)]\!] = \delta([\![v]\!])$$

- as it is logically equivalent to $\blacktriangleright([\![w]\!] = [\![v]\!])$
- So also need

$$\mathtt{unfold}\,(\mathtt{fold}\,w) \Downarrow^1 v$$

- we defined it as $\blacktriangleright(w \Downarrow^0 v)$

# Adequacy

**Theorem.** If $\vdash M : \mathbf{nat}$ then $M \Downarrow^k v \iff [\![M]\!] = \delta^k([\![v]\!])$

- If $M = \mathtt{unfold}\,(\mathtt{fold}\,w)$ and $w \neq v$ the following holds

$$[\![\mathtt{unfold}\,(\mathtt{fold}\,w)]\!] = \delta([\![v]\!])$$

- as it is logically equivalent to $\blacktriangleright\,([\![w]\!] = [\![v]\!])$
- So also need

$$\mathtt{unfold}\,(\mathtt{fold}\,w) \Downarrow^1 v$$

- we defined it as $\blacktriangleright(w \Downarrow^0 v)$

## Proof by Logical Relation argument

**Lemma.** If $M : \sigma$ closed then $[\![M]\!]\ \mathcal{R}_\sigma\ M$

# Logical relation

$$\mathcal{R}_\tau : [\![\tau]\!] \times \mathtt{Term}_{\mathtt{FPC}} \to U$$

- Defined by guarded recursion and induction on the types

$$\eta(*)\ \mathcal{R}_1\ M =_{\mathrm{def}} M \Downarrow^0 \langle\rangle$$
$$\Theta_1(\alpha)\ \mathcal{R}_1\ M =_{\mathrm{def}} M \to_*^1 M' \text{ and } \alpha \blacktriangleright \mathcal{R}_1\ \mathrm{next}(M')$$

- For recursive types

$$\alpha\ \mathcal{R}_{\mu\alpha.\tau}\ M =_{\mathrm{def}} \mathtt{unfold}\ M \to_*^1 M' \text{ and } \alpha \blacktriangleright \mathcal{R}_{\tau[\mu\alpha.\tau/\alpha]}\ \mathrm{next}(M')$$

- $x \blacktriangleright \mathcal{R}_\tau\ M = \blacktriangleright[y \leftarrow x, N \leftarrow M].(y\ \mathcal{R}_\tau\ N)$

# Extensional adequacy

# Weak Bisimulation Logical relation

- Relation on denotable terms

$$\approx_\tau : [\![\tau]\!] \to [\![\tau]\!] \to U$$

  defined by guarded recursion and induction on the types
- Relates computations produce the same value or diverge

$$\begin{aligned}
\eta(v) \approx_1 \eta(v') &=_{\text{def}} v = v' \\
\eta(v) \approx_1 \beta &=_{\text{def}} \Sigma n.\beta = (\delta_1)^n([\![v]\!]) \\
\alpha \approx_1 \eta(v) &=_{\text{def}} \Sigma n.\alpha = (\delta_1)^n([\![v]\!]) \\
\Theta_1(\alpha') \approx_1 \Theta_1(\beta') &=_{\text{def}} \alpha' \blacktriangleright \approx_1 \beta' \\
&\dots \\
x \approx_{\mu\alpha.\tau} y &=_{\text{def}} x \blacktriangleright \approx_{\tau[\mu\alpha.\tau/\alpha]} y
\end{aligned}$$

# Recovering the global behaviour

- Idea originally due to Atkey and McBride
- We reformulate the interpretation s.t.

$$[\![1]\!]^{\mathrm{gl}} \cong 1 + [\![1]\!]^{\mathrm{gl}} A$$

- Lift $\approx_\sigma$ to

$$\approx_\sigma^{\mathrm{gl}} : [\![\sigma]\!]^{\mathrm{gl}} \to [\![\sigma]\!]^{\mathrm{gl}} \to U$$

- Interpretation of terms $\Gamma \vdash M : \sigma$

$$[\![M]\!]^{\mathrm{gl}} : ([\![\Gamma]\!]^{\mathrm{gl}} \to [\![\sigma]\!]^{\mathrm{gl}})$$

# Extensional Computational Adequacy

**Theorem.**

- If $\Gamma \vdash M, N : \tau$, and
- $[\![M]\!]^{\mathrm{gl}} \approx^{\mathrm{gl}}_{\Gamma,\tau} [\![N]\!]^{\mathrm{gl}}$, and
- $C[-] : (\Gamma, \tau) \to \mathbf{nat}$, then

$$\prod n.(\Sigma k.C[M] \Downarrow^k n) \iff (\Sigma l.C[N] \Downarrow^l n)$$

# Conclusions

# Conclusions

- Denotational semantics of Recursive Types using GdTT
- Simpler and more abstract
- The price to pay is intensionality
- Extensional adequacy using co-inductive logical relation

**Main message**

Guarded type theory is a natural setting for denotational semantics.

## Future work

- General References, Co-inductive Types and Effects

# Conclusions

- Denotational semantics of Recursive Types using GdTT
- Simpler and more abstract
- The price to pay is intensionality
- Extensional adequacy using co-inductive logical relation

**Main message**

Guarded type theory is a natural setting for denotational semantics.

## Future work

- General References, Co-inductive Types and Effects

Thanks!

# Big step operational semantics

$$
\begin{aligned}
v \Downarrow^0 Q &=_{\mathrm{def}} Q(v) \\
\text{pred}\ M \Downarrow^k Q &=_{\mathrm{def}} M \Downarrow^k (\lambda x.\Sigma n : \mathbb{N}.x = \underline{n} \text{ and } Q(\underline{n-1})) \\
\text{succ}\ M \Downarrow^k Q &=_{\mathrm{def}} M \Downarrow^k (\lambda x.\Sigma n : \mathbb{N}.x = \underline{n} \text{ and } Q(\underline{n+1})) \\
\mathsf{Y}_\sigma\ M \Downarrow^{k+1} Q &=_{\mathrm{def}} \blacktriangleright (M(\mathsf{Y}_\sigma\ M) \Downarrow^k Q) \\
MN \Downarrow^{k+m} Q &=_{\mathrm{def}} M \Downarrow^k Q' \\
&\quad \text{where } Q'(\lambda x.L) = L[N/x] \Downarrow^m Q \\
\text{ifz}\ L\ M\ N \Downarrow^{k+m} Q &=_{\mathrm{def}} L \Downarrow^k Q' \\
&\quad \text{where } Q'(\underline{0}) = M \Downarrow^m Q \text{ and } Q'(\underline{n+1}) = N \Downarrow^m Q
\end{aligned}
$$

# Big step operational semantics

$$v \Downarrow^0 Q =_{\mathrm{def}} Q(v)$$

$$\mathsf{pred}\ M \Downarrow^k Q =_{\mathrm{def}} M \Downarrow^k (\lambda x.\Sigma n : \mathbb{N}.x = \underline{n} \text{ and } Q(\underline{n-1}))$$

$$\mathsf{succ}\ M \Downarrow^k Q =_{\mathrm{def}} M \Downarrow^k (\lambda x.\Sigma n : \mathbb{N}.x = \underline{n} \text{ and } Q(\underline{n+1}))$$

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

$$MN \Downarrow^{k+m} Q =_{\mathrm{def}} M \Downarrow^k Q'$$
$$\text{where } Q'(\lambda x.L) = L[N/x] \Downarrow^m Q$$

$$\mathsf{ifz}\ L\ M\ N \Downarrow^{k+m} Q =_{\mathrm{def}} L \Downarrow^k Q'$$
$$\text{where } Q'(\underline{0}) = M \Downarrow^m Q \text{ and } Q'(\underline{n+1}) = N \Downarrow^m Q$$

Synchronising with the type theory

# Big-step operational semantics

$$v \Downarrow^k Q =_{\mathrm{def}} Q(v, k)$$

$$\texttt{case}\ L\ \texttt{of}\ \texttt{inl}\ x_1.M; \texttt{inr}\ x_2.N \Downarrow^k Q =_{\mathrm{def}} L \Downarrow^k Q'$$

$$\text{where}\quad \begin{aligned} Q'(\texttt{inl}\ L, l) &=_{\mathrm{def}} M[L/x_1] \Downarrow^l Q \\ Q'(\texttt{inr}\ L, l) &=_{\mathrm{def}} N[L/x_2] \Downarrow^l Q \end{aligned}$$

$$\texttt{fst}\ L \Downarrow^k Q =_{\mathrm{def}} L \Downarrow^k Q'$$

$$\text{where } Q'(\langle M, N\rangle, m) =_{\mathrm{def}} M \Downarrow^m Q$$

$$\texttt{snd}\ L \Downarrow^k Q =_{\mathrm{def}} L \Downarrow^k Q'$$

$$\text{where } Q'(\langle M, N\rangle, m) =_{\mathrm{def}} N \Downarrow^m Q$$

$$MN \Downarrow^k Q =_{\mathrm{def}} M \Downarrow^k Q'$$

$$\text{where } Q'(\lambda x.L, m) =_{\mathrm{def}} L[N/x] \Downarrow^m Q$$

$$\texttt{unfold}\ M \Downarrow^k Q =_{\mathrm{def}} M \Downarrow^k Q'$$

$$\text{where } Q'(\texttt{fold}\ N, m+1) =_{\mathrm{def}} \blacktriangleright (N \Downarrow^m Q)$$

# Weak Bisimulation Logical relation

$$\eta(v) \approx_1 \eta(v') =_{\text{def}} v = v'$$

$$\eta(v) \approx_1 \beta =_{\text{def}} \Sigma n.\beta = (\delta_1)^n([\![v]\!])$$

$$\alpha \approx_1 \eta(v) =_{\text{def}} \Sigma n.\alpha = (\delta_1)^n([\![v]\!])$$

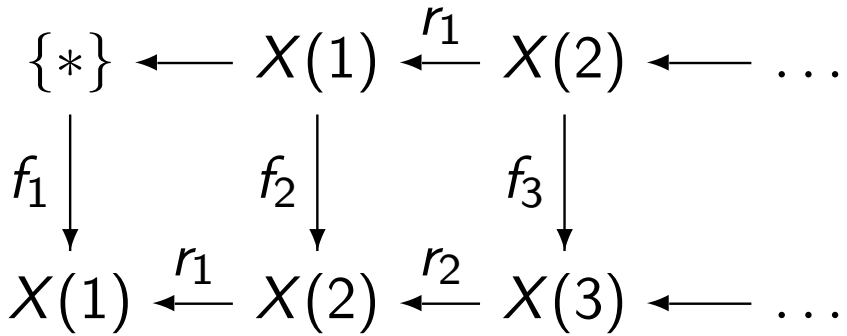
$$\Theta_1(\alpha') \approx_1 \Theta_1(\beta') =_{\text{def}} \alpha' \blacktriangleright \approx_1 \beta'$$

$$\dots$$

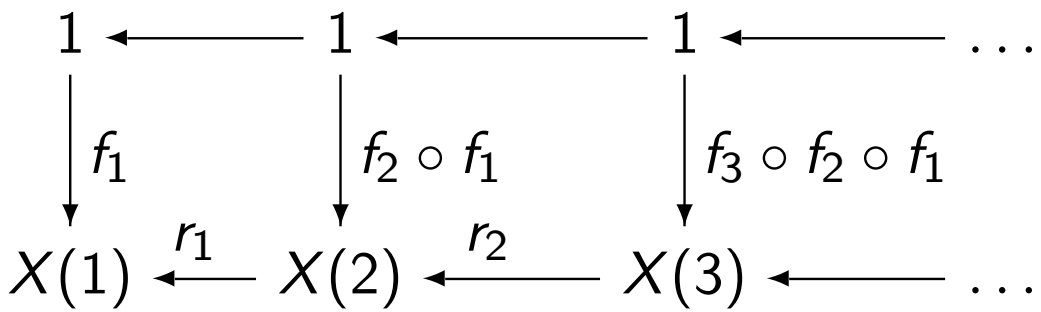
$$x \approx_{\mu\alpha.\tau} y =_{\text{def}} x \blacktriangleright \approx_{\tau[\mu\alpha.\tau/\alpha]} y$$

# Construction of fixed points

- Given $f : \blacktriangleright X \to X$:



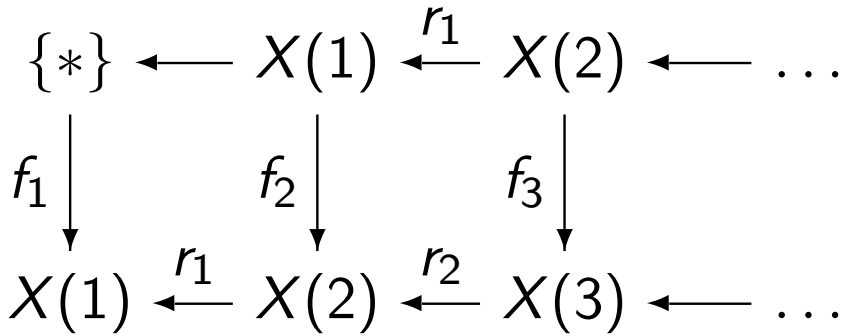

- Construct $\mathrm{fix}_X(f) : 1 \to X$:



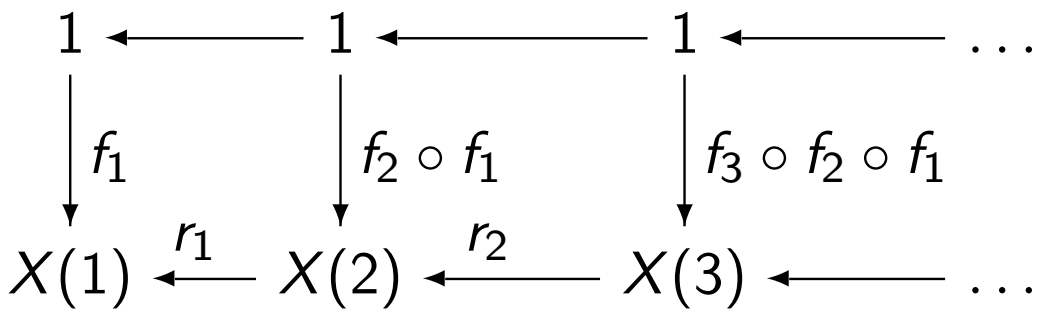

- Fixed points are unique

# Construction of fixed points

- Given $f : \blacktriangleright X \to X$:




- Construct $\mathrm{fix}_X(f) : 1 \to X$:




- Fixed points are unique

# Guarded recursive types as fixed points

Universe closed under $\blacktriangleright$

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

$$\mathrm{El}(\triangleright(\mathrm{next}(A))) = \blacktriangleright \mathrm{El}(A)$$

# Guarded recursive types as fixed points

Universe closed under $\blacktriangleright$

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

$$\mathrm{El}(\triangleright(\mathrm{next}(A))) = \blacktriangleright \mathrm{El}(A)$$

## Guarded Streams

- Type of streams as fixed point for universe map

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

- Then

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

# Guarded dependent type theory (gDTT)

$$\begin{aligned}
\mathcal{R}_\tau &: [\![\tau]\!] \to \mathtt{Term}_{\mathtt{PCF}} \to U \\
\mathrm{next}(\ \mathcal{R}_\tau\ ) &: \blacktriangleright([\![\tau]\!] \to \mathtt{Term}_{\mathtt{PCF}} \to U) \\
\mathrm{next}(\ \mathcal{R}_\tau\ ) \circledast (-) \circledast (-) &: \blacktriangleright[\![\tau]\!] \to \blacktriangleright \mathtt{Term}_{\mathtt{PCF}} \to \blacktriangleright U
\end{aligned}$$

- Define (using $\triangleright : \blacktriangleright U \to U$)

$$\begin{aligned}
\blacktriangleright \mathcal{R}_\tau &=_{\mathrm{def}} \triangleright \circ (\mathrm{next}(\ \mathcal{R}_\tau\ ) \circledast (-) \circledast (-)) \\
&: \blacktriangleright[\![\tau]\!] \to \blacktriangleright \mathtt{Term}_{\mathtt{PCF}} \to U
\end{aligned}$$

# Guarded dependent type theory (gDTT)

$$\begin{aligned}
\mathcal{R}_\tau &: [\![\tau]\!] \to \mathtt{Term}_{\mathtt{PCF}} \to U \\
\mathrm{next}(\ \mathcal{R}_\tau\ ) &: \blacktriangleright([\![\tau]\!] \to \mathtt{Term}_{\mathtt{PCF}} \to U) \\
\mathrm{next}(\ \mathcal{R}_\tau\ ) \circledast (-) \circledast (-) &: \blacktriangleright[\![\tau]\!] \to \blacktriangleright \mathtt{Term}_{\mathtt{PCF}} \to \blacktriangleright U
\end{aligned}$$

- Define (using $\triangleright : \blacktriangleright U \to U$)

$$\begin{aligned}
\blacktriangleright \mathcal{R}_\tau &=_{\mathrm{def}} \triangleright \circ (\mathrm{next}(\ \mathcal{R}_\tau\ ) \circledast (-) \circledast (-)) \\
&: \blacktriangleright[\![\tau]\!] \to \blacktriangleright \mathtt{Term}_{\mathtt{PCF}} \to U
\end{aligned}$$

- Special syntax

$$x\ \blacktriangleright \mathcal{R}_\tau\ M = \blacktriangleright [y \leftarrow x, N \leftarrow M].(y\ \mathcal{R}_\tau\ N)$$

# Syntax: multiple clocks

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

$$\frac{\Gamma \vdash_\Delta A : \mathsf{Type} \qquad \vdash_\Delta \kappa}{\Gamma \vdash_\Delta \overset{\kappa}{\blacktriangleright} A : \mathsf{Type}}$$

$$\mathtt{fix}^\kappa : (\overset{\kappa}{\blacktriangleright} X \to X) \to X$$

- etc

# Universal quantification over clocks

$$\frac{\Gamma \vdash_{\Delta,\kappa} A : \mathsf{Type} \quad \kappa \notin \mathsf{fc}(\Gamma)}{\Gamma \vdash_{\Delta} \forall \kappa.A : \mathsf{Type}}$$

$$\frac{\Gamma \vdash_{\Delta,\kappa} t : A \quad \kappa \notin \mathsf{fc}(\Gamma)}{\Gamma \vdash_{\Delta} \Lambda \kappa.t : \forall \kappa.A}$$

$$\frac{\Gamma \vdash_{\Delta} t : \forall \kappa.A \quad \vdash_{\Delta} \kappa'}{\Gamma \vdash_{\Delta} t[\kappa'] : A[\kappa'/\kappa]}$$

- Allows controlled elimination of $\blacktriangleright$

$$\mathsf{force} : \forall \kappa.\ \blacktriangleright^{\kappa} A \to \forall \kappa.A$$

- Clock quantification is right adjoint to clock weakening

# Recovering the global behaviour

- Idea originally due to Atkey and McBride
- We reformulate the interpretation
- $L^{\mathrm{gl}}1 = \forall\kappa.L1$ so

$$[\![1]\!]^{\mathrm{gl}} \cong 1 + [\![1]\!]^{\mathrm{gl}}$$

- Lift $\approx$

$$\approx^{\mathrm{gl}}_{\sigma} : \forall\kappa.[\![\sigma]\!] \to \forall\kappa.[\![\sigma]\!] \to U$$

$$x \approx^{\mathrm{gl}}_{\sigma} y = \forall\kappa.x[\kappa] \approx_{\sigma} y[\kappa]$$

- Interpretation of terms $\Gamma \vdash M : \sigma$

$$[\![M]\!]^{\mathrm{gl}} : ([\![\Gamma]\!]^{\mathrm{gl}} \to [\![\sigma]\!]^{\mathrm{gl}})$$

$$[\![M]\!]^{\mathrm{gl}} = \Lambda\kappa.[\![M]\!]$$

- Can prove

$$\text{if } \delta^{\mathrm{gl}}(x) \approx^{\mathrm{gl}}_{1} \delta^{\mathrm{gl}}(y) \text{ then } \forall\kappa.\ \blacktriangleright^{\kappa} (x[\kappa] \approx_{\sigma} y[\kappa]) \text{ then } x \approx^{\mathrm{gl}}_{1} y$$