

Rewrite Semantics for Guarded Recursion

Patrick Bahr

IT University of Copenhagen

joint work with Rasmus Møgelberg

Guarded Recursive Types

Dependent Types

Reduction Semantics

Guarded Recursive Types

Guarded Recursion

- ▶ type modality \triangleright (pronounced “later”)
- ▶ \triangleright is an applicative functor

$$\text{next} : A \rightarrow \triangleright A$$

$$\circledast : \triangleright(A \rightarrow B) \rightarrow \triangleright A \rightarrow \triangleright B$$

- ▶ fixed-point operator $\text{fix} : (\triangleright A \rightarrow A) \rightarrow A$
- ▶ guarded recursive types: $\mu X.A$

Example

guarded streams

$$\begin{aligned}\text{Str} &= \mu X. \text{Nat} \times \triangleright X \\ \text{cons} &: \text{Nat} \rightarrow \triangleright \text{Str} \rightarrow \text{Str} \\ \text{cons} &= \lambda x. \lambda y. \langle x, y \rangle \\ \text{ones} &: \text{Str} \\ \text{ones} &= \text{fix}(\lambda x. \text{cons } 1 \ x)\end{aligned}$$

Motivation

- ▶ functional reactive programming
- ▶ productive coprogramming
(clocks & clock quantification)
- ▶ solving recursive domain equations
(\rightarrow synthetic domain theory)

Motivation

- ▶ functional reactive programming
- ▶ productive coprogramming
(clocks & clock quantification)
- ▶ solving recursive domain equations
(\rightarrow synthetic domain theory)

$$X \simeq \text{Nat} + (X \times X) + (X \rightarrow X)$$

Motivation

- ▶ functional reactive programming
- ▶ productive coprogramming
(clocks & clock quantification)
- ▶ solving recursive domain equations
(\rightarrow synthetic domain theory)

$$X \simeq \text{Nat} + (\blacktriangleright X \times \blacktriangleright X) + (\blacktriangleright X \rightarrow \blacktriangleright X)$$

Dependent Types

A. Bizjak, H. B. Grathwohl, R. Clouston, R. E. Møgelberg, and L. Birkedal.
Guarded dependent type theory with coinductive types. In FoSSaCS, 2016.

Combining Π and \triangleright

$$\frac{\Gamma \vdash s : \Pi x : A. B \quad \Gamma \vdash t : A}{\Gamma \vdash s t : B[t/x]}$$

Combining Π and \triangleright

$$\frac{\Gamma \vdash s : \Pi x : A. B \quad \Gamma \vdash t : A}{\Gamma \vdash s t : B[t/x]}$$

$$\frac{\Gamma \vdash s : \triangleright (\Pi x : A. B) \quad \Gamma \vdash t : \triangleright A}{\Gamma \vdash s \circledast t : ???}$$

Combining Π and \triangleright

$$\frac{\Gamma \vdash s : \Pi x : A. B \quad \Gamma \vdash t : A}{\Gamma \vdash s t : B[t/x]}$$

$$\frac{\Gamma \vdash s : \triangleright (\Pi x : A. B) \quad \Gamma \vdash t : \triangleright A}{\Gamma \vdash s \circledast t : \triangleright B[t/x]}$$

Combining Π and \triangleright

$$\frac{\Gamma \vdash s : \Pi x : A. B \quad \Gamma \vdash t : A}{\Gamma \vdash s t : B[t/x]}$$

$$\frac{\Gamma \vdash s : \triangleright (\Pi x : A. B) \quad \Gamma \vdash t : \triangleright A}{\Gamma \vdash s \circledast t : \triangleright B[t/x]}$$

- ▶ Problem: $t : \triangleright A$, but $x : A$

Combining Π and \triangleright

$$\frac{\Gamma \vdash s : \Pi x : A. B \quad \Gamma \vdash t : A}{\Gamma \vdash s t : B[t/x]}$$

$$\frac{\Gamma \vdash s : \triangleright (\Pi x : A. B) \quad \Gamma \vdash t : \triangleright A}{\Gamma \vdash s \circledast t : \triangleright B[t/x]}$$

- ▶ **Problem:** $t : \triangleright A$, but $x : A$
- ▶ needed: getting rid of \triangleright in a controlled way

Delayed Substitutions

Instead of

$$\frac{\Gamma \vdash s : \triangleright (\Pi x : A.B) \quad \Gamma \vdash t : \triangleright A}{\Gamma \vdash s \circledast t : \triangleright B[t/x]}$$

we have

$$\frac{\Gamma \vdash s : \triangleright (\Pi x : A.B) \quad \Gamma \vdash t : \triangleright A}{\Gamma \vdash s \circledast t : \triangleright [x \leftarrow t].B}$$

Delayed Substitutions

Instead of

$$\frac{\Gamma \vdash s : \triangleright (\Pi x : A. B) \quad \Gamma \vdash t : \triangleright A}{\Gamma \vdash s \circledast t : \triangleright B[t/x]}$$

we have

$$\frac{\Gamma \vdash s : \triangleright (\Pi x : A. B) \quad \Gamma \vdash t : \triangleright A}{\Gamma \vdash s \circledast t : \triangleright [x \leftarrow t]. B}$$

In general

$$\triangleright [x_1 \leftarrow t_1, \dots, x_n \leftarrow t_n]. A$$

Delayed Substitutions

Instead of

$$\frac{\Gamma \vdash s : \triangleright (\Pi x : A.B) \quad \Gamma \vdash t : \triangleright A}{\Gamma \vdash s \circledast t : \triangleright B[t/x]}$$

we have

$$\frac{\Gamma \vdash s : \triangleright (\Pi x : A.B) \quad \Gamma \vdash t : \triangleright A}{\Gamma \vdash s \circledast t : \triangleright [x \leftarrow t].B}$$

In general

$$\triangleright [x_1 \leftarrow t_1, \dots, x_n \leftarrow t_n].A$$

$$\text{next } [x_1 \leftarrow t_1, \dots, x_n \leftarrow t_n].t$$

Equalities

$$\triangleright [x \leftarrow \text{next } u].A = \triangleright A[u/x]$$

Equalities

$$\triangleright_{\xi} [x \leftarrow \text{next}_{\xi}.u] .A = \triangleright_{\xi} .A [u/x]$$

Equalities

$$\triangleright \xi [x \leftarrow \text{next}\xi.u] .A = \triangleright \xi .A [u/x]$$

$$\triangleright \xi [x \leftarrow u] .A = \triangleright \xi .A \quad \text{if } x \notin \text{fv}(A)$$

$$\triangleright \xi [x \leftarrow u, y \leftarrow v] \xi' .A = \triangleright \xi [y \leftarrow v, x \leftarrow u] \xi' .A \quad \text{if } \dots$$

Equalities

$$\triangleright \xi [x \leftarrow \text{next}\xi.u] .A = \triangleright \xi .A [u/x]$$

$$\triangleright \xi [x \leftarrow u] .A = \triangleright \xi .A \quad \text{if } x \notin \text{fv}(A)$$

$$\triangleright \xi [x \leftarrow u, y \leftarrow v] \xi' .A = \triangleright \xi [y \leftarrow v, x \leftarrow u] \xi' .A \quad \text{if } \dots$$

$$\text{next}\xi [x \leftarrow \text{next}\xi.u] .t = \text{next}\xi .t [u/x]$$

$$\text{next}\xi [x \leftarrow u] .t = \text{next}\xi .t \quad \text{if } x \notin \text{fv}(t)$$

$$\text{next}\xi [x \leftarrow u, y \leftarrow v] \xi' .t = \text{next}\xi [y \leftarrow v, x \leftarrow u] \xi' .t \quad \text{if } \dots$$

Typing rule

Simple Case

$$\frac{\Gamma, x : A \vdash t : B \quad \Gamma \vdash u : \triangleright A}{\Gamma \vdash \text{next } [x \leftarrow u].t : \triangleright [x \leftarrow u].B}$$

Typing rule

Simple Case

$$\frac{\Gamma, x : A \vdash t : B \quad \Gamma \vdash u : \triangleright A}{\Gamma \vdash \text{next } [x \leftarrow u].t : \triangleright [x \leftarrow u].B}$$

In General

$$\frac{\Gamma, x_1 : A_1, \dots, x_n : A_n \vdash t : B \quad \Gamma \vdash t_i : \triangleright [x_1 \leftarrow t_1, \dots, x_{i-1} \leftarrow t_{i-1}].A_i \text{ for all } 1 \leq i \leq n}{\Gamma \vdash \text{next } [x_1 \leftarrow t_1, \dots, x_n \leftarrow t_n].t : \triangleright [x_1 \leftarrow t_1, \dots, x_n \leftarrow t_n].B}$$

Applicative Structure

Applicative structure can be defined in terms of delayed substitutions:

$$s \circledast t = \text{next } [x \leftarrow s, y \leftarrow t] . x y$$

Applicative Structure

Applicative structure can be defined in terms of delayed substitutions:

$$s \circledast t = \text{next } [x \leftarrow s, y \leftarrow t] .x y$$

$$\begin{aligned} & \text{next } u \circledast \text{next } v \\ &= \text{next } [x \leftarrow \text{next } u, y \leftarrow \text{next } v] .x y \\ &= \text{next } [x \leftarrow \text{next } u] .x v \\ &= \text{next}(u v) \end{aligned}$$

Applicative Functor Laws

We need to add the following equality

$$\text{next}\xi [x \leftarrow t].x = t$$

Applicative Functor Laws

We need to add the following equality

$$\text{next}\xi [x \leftarrow t].x = t$$

We can then derive the applicative functor laws:

$$\begin{aligned}\text{next}(\lambda x.x) \circledast t &= t \\ \text{next}(\lambda f.\lambda g.\lambda x.f (g x)) \circledast s \circledast t \circledast u &= s \circledast (t \circledast u) \\ \text{next } s \circledast \text{next } t &= \text{next } (s t) \\ s \circledast \text{next } t &= \text{next}(\lambda f.f t) \circledast s\end{aligned}$$

Multiple Clocks and Clock Quantification

$$\Gamma \vdash_{\Delta} t : B \quad \kappa \in \Delta$$

$$\Gamma \vdash_{\Delta} \text{next}^{\kappa} t : \triangleright^{\kappa} B$$

Multiple Clocks and Clock Quantification

$$\frac{\begin{array}{l} \Gamma, x_1 : A_1, \dots, x_n : A_n \vdash_{\Delta} t : B \quad \kappa \in \Delta \\ \Gamma \vdash_{\Delta} t_i : \triangleright^{\kappa} [x_1 \leftarrow t_1, \dots, x_{i-1} \leftarrow t_{i-1}]. A_i \text{ for all } 1 \leq i \leq n \end{array}}{\Gamma \vdash_{\Delta} \text{next}^{\kappa} [x_1 \leftarrow t_1, \dots, x_n \leftarrow t_n]. t : \triangleright^{\kappa} [x_1 \leftarrow t_1, \dots, x_n \leftarrow t_n]. B}$$

Multiple Clocks and Clock Quantification

$$\frac{\Gamma, x_1 : A_1, \dots, x_n : A_n \vdash_{\Delta} t : B \quad \kappa \in \Delta \quad \Gamma \vdash_{\Delta} t_i : \triangleright^{\kappa} [x_1 \leftarrow t_1, \dots, x_{i-1} \leftarrow t_{i-1}]. A_i \text{ for all } 1 \leq i \leq n}{\Gamma \vdash_{\Delta} \text{next}^{\kappa} [x_1 \leftarrow t_1, \dots, x_n \leftarrow t_n]. t : \triangleright^{\kappa} [x_1 \leftarrow t_1, \dots, x_n \leftarrow t_n]. B}$$

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

$$\frac{\Gamma \vdash_{\Delta, \kappa} t : \triangleright^{\kappa} A \quad \Gamma \vdash_{\Delta}}{\Gamma \vdash_{\Delta} \text{prev} \kappa. t : \forall \kappa. A}$$

Multiple Clocks and Clock Quantification

$$\frac{\Gamma, x_1 : A_1, \dots, x_n : A_n \vdash_{\Delta} t : B \quad \kappa \in \Delta \quad \Gamma \vdash_{\Delta} t_i : \triangleright^{\kappa} [x_1 \leftarrow t_1, \dots, x_{i-1} \leftarrow t_{i-1}]. A_i \text{ for all } 1 \leq i \leq n}{\Gamma \vdash_{\Delta} \text{next}^{\kappa} [x_1 \leftarrow t_1, \dots, x_n \leftarrow t_n]. t : \triangleright^{\kappa} [x_1 \leftarrow t_1, \dots, x_n \leftarrow t_n]. B}$$

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

$$\frac{\Gamma \vdash_{\Delta, \kappa} t : \triangleright^{\kappa} A \quad \Gamma \vdash_{\Delta}}{\Gamma \vdash_{\Delta} \text{prev} \kappa. t : \forall \kappa. A}$$

$$\text{prev} \kappa. (\text{next}^{\kappa} t) = \Lambda \kappa. t$$

Multiple Clocks and Clock Quantification

$$\frac{\Gamma, x_1 : A_1, \dots, x_n : A_n \vdash_{\Delta} t : B \quad \kappa \in \Delta \quad \Gamma \vdash_{\Delta} t_i : \triangleright^{\kappa} [x_1 \leftarrow t_1, \dots, x_{i-1} \leftarrow t_{i-1}]. A_i \text{ for all } 1 \leq i \leq n}{\Gamma \vdash_{\Delta} \text{next}^{\kappa} [x_1 \leftarrow t_1, \dots, x_n \leftarrow t_n]. t : \triangleright^{\kappa} [x_1 \leftarrow t_1, \dots, x_n \leftarrow t_n]. B}$$

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

$$\frac{\Gamma \vdash_{\Delta, \kappa} t : \triangleright^{\kappa} A \quad \Gamma \vdash_{\Delta}}{\Gamma \vdash_{\Delta} \text{prev} \kappa. t : \forall \kappa. A}$$

$$\text{prev} \kappa. (\text{next}^{\kappa} \xi. t) = \Lambda \kappa. t(\text{adv}^{\kappa}(\xi))$$

Multiple Clocks and Clock Quantification

$$\frac{\Gamma, x_1 : A_1, \dots, x_n : A_n \vdash_{\Delta} t : B \quad \kappa \in \Delta \quad \Gamma \vdash_{\Delta} t_i : \triangleright^{\kappa} [x_1 \leftarrow t_1, \dots, x_{i-1} \leftarrow t_{i-1}]. A_i \text{ for all } 1 \leq i \leq n}{\Gamma \vdash_{\Delta} \text{next}^{\kappa} [x_1 \leftarrow t_1, \dots, x_n \leftarrow t_n]. t : \triangleright^{\kappa} [x_1 \leftarrow t_1, \dots, x_n \leftarrow t_n]. B}$$

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

$$\frac{\Gamma \vdash_{\Delta, \kappa} t : \triangleright^{\kappa} A \quad \Gamma \vdash_{\Delta}}{\Gamma \vdash_{\Delta} \text{prev} \kappa. t : \forall \kappa. A}$$

$$\text{prev} \kappa. (\text{next}^{\kappa} \xi. t) = \Lambda \kappa. t(\text{adv}^{\kappa}(\xi))$$

$$\text{adv}^{\kappa} ([x_1 \leftarrow t_1, \dots, x_n \leftarrow t_n]) = [(\text{prev} \kappa. t_1)[\kappa]/x_1, \dots, (\text{prev} \kappa. t_n)[\kappa]/x_n]$$

Example

$$\text{Str}^\kappa = \mu X. \text{Nat} \times \triangleright^\kappa X$$

$$\text{Str} = \forall \kappa. \text{Str}^\kappa$$

$$tl : \text{Str} \rightarrow \text{Str}$$

$$tl = \lambda x. \text{prev}^\kappa. \pi_2(x[\kappa])$$

$$hd : \text{Str} \rightarrow \text{Nat}$$

$$hd = \lambda x. \pi_1(x[\kappa_0])$$

$$tln : \text{Nat} \rightarrow \text{Str} \rightarrow \text{Str}$$

$$tln = \lambda x. \lambda y. \text{rec } x \ y \ (\lambda z_1. \lambda z_2. tl \ z_2)$$

$$nth : \text{Nat} \rightarrow \text{Str} \rightarrow \text{Nat}$$

$$nth = \lambda x. \lambda y. hd \ (tln \ x \ y)$$

Example

$$\text{Str}^{\kappa} = \mu X. \text{Nat} \times \triangleright^{\kappa} X$$

$$\text{Str} = \forall \kappa. \text{Str}^{\kappa}$$

$$tl : \text{Str} \rightarrow \text{Str}$$

$$tl = \lambda x. \text{prev}^{\kappa}. \pi_2(x[\kappa])$$

$$hd : \text{Str} \rightarrow \text{Nat}$$

$$hd = \lambda x. \pi_1(x[\kappa_0])$$

$$tln : \text{Nat} \rightarrow \text{Str} \rightarrow \text{Str}$$

$$tln = \lambda x. \lambda y. \text{rec } x \ y \ (\lambda z_1. \lambda z_2. tl \ z_2)$$

$$nth : \text{Nat} \rightarrow \text{Str} \rightarrow \text{Nat}$$

$$nth = \lambda x. \lambda y. hd \ (tln \ x \ y)$$

$$skip : \text{Str} \rightarrow \text{Str}$$

$$skip = \lambda x. \Lambda \kappa. (\text{fix}^{\kappa} (\lambda f. \lambda z. \langle hd \ z, \text{next}^{\kappa} [y \leftarrow f] . y \ (tl \ (tl \ z)) \rangle)) \ x$$

Reduction Semantics

Motivation

- ▶ we want to implement a type checker for dependent type theory with guarded recursion
- ▶ we need to decide the equality theory
- ▶ possible approach: reduction relation that is
 - ▶ strongly normalising
 - ▶ confluent

Problems with Normalisation

- ▶ Fixed-point combinator!

$$\text{fix}^{\kappa} t = t(\text{next}^{\kappa}(\text{fix}^{\kappa} t))$$

- ▶ We cannot turn this equation into a normalising rewrite rule:

$$\text{next}\xi [x \leftarrow u, y \leftarrow v] \xi'.A = \text{next}\xi [y \leftarrow v, x \leftarrow u] \xi'.A$$

Problems with Confluence

$$\text{next}^k \xi [x \leftarrow \text{next}^k \xi . s] . t = \text{next}^k \xi . t [s/x]$$

Problems with Confluence

$$\text{next}^{\kappa\xi} [x \leftarrow \text{next}^{\kappa\xi}.s] .t \rightarrow \text{next}^{\kappa\xi}.t [s/x]$$

Problems with Confluence

$$\text{next}^{\kappa} \xi [x \leftarrow \text{next}^{\kappa} \xi . s] . t \rightarrow \text{next}^{\kappa} \xi . t [s/x]$$

Problems with Confluence

$$\text{next}^{\kappa} \xi [x \leftarrow \text{next}^{\kappa} \xi . s] . t \rightarrow \text{next}^{\kappa} \xi . t [s/x]$$

$$t = [x_1 \leftarrow y, x_2 \leftarrow [x_1 \leftarrow y] . 0] . x_1 x_2$$

$$t \rightarrow \text{next}^{\kappa} [x_1 \leftarrow y] . x_1 0$$

$$t \rightarrow \text{next}^{\kappa} [x_1 \leftarrow y, x_2 \leftarrow \text{next}^{\kappa} . 0] . x_1 x_2$$

Alternative Calculus without Delayed Substitutions

Idea

- ▶ controlled conversion freeze ^{κ} : $\triangleright^{\kappa} A \rightarrow A$.

Alternative Calculus without Delayed Substitutions

Idea

- ▶ controlled conversion $\text{freeze}^{\kappa} : \triangleright^{\kappa} A \rightarrow A$.
- ▶ $\text{next}^{\kappa} [x \leftarrow t] . u \rightsquigarrow \text{next}^{\kappa} u [\text{freeze}^{\kappa} t/x]$
- ▶ $\triangleright^{\kappa} [x \leftarrow t] . A \rightsquigarrow \triangleright^{\kappa} A [\text{freeze}^{\kappa} t/x]$

Alternative Calculus without Delayed Substitutions

Idea

- ▶ controlled conversion $\text{freeze}^{\kappa} : \triangleright^{\kappa} A \rightarrow A$.
- ▶ $\text{next}^{\kappa} [x \leftarrow t] . u \rightsquigarrow \text{next}^{\kappa} / . u [\text{freeze}_{/}^{\kappa} t / x]$
- ▶ $\triangleright^{\kappa} [x \leftarrow t] . A \rightsquigarrow \triangleright^{\kappa} / . A [\text{freeze}_{/}^{\kappa} t / x]$

Alternative Calculus without Delayed Substitutions

Idea

- ▶ controlled conversion $\text{freeze}^\kappa : \triangleright^\kappa A \rightarrow A$.
- ▶ $\text{next}^\kappa [x \leftarrow t].u \rightsquigarrow \text{next}^\kappa / .u [\text{freeze}_l^\kappa t/x]$
- ▶ $\triangleright^\kappa [x \leftarrow t].A \rightsquigarrow \triangleright^\kappa / .A [\text{freeze}_l^\kappa t/x]$

$$\frac{\Gamma \vdash_{\Delta}^{\mathcal{L}} t :_{\mathcal{I}} \triangleright^\kappa / .A \quad l : \kappa \in \mathcal{L}}{\Gamma \vdash_{\Delta}^{\mathcal{L}} \text{freeze}_l^\kappa t :_{\mathcal{I},l} A}$$

$$\frac{\Gamma \vdash_{\Delta}^{\mathcal{L},l:\kappa} t :_{\mathcal{I},l} A \quad \kappa \in \Delta \quad \Gamma \vdash_{\Delta}}{\Gamma \vdash_{\Delta}^{\mathcal{L}} \text{next}^\kappa / .t :_{\mathcal{I}} \triangleright^\kappa / .A}$$

Other Typing Rules

$$\frac{\mathcal{J} \subseteq \mathcal{I}}{\Gamma, x :_{\mathcal{J}} A, \Gamma' \vdash_{\Delta}^{\mathcal{L}} x :_{\mathcal{I}} A} \qquad \frac{\Gamma, x :_{\mathcal{I}} A \vdash_{\Delta}^{\mathcal{L}} t :_{\mathcal{I}} B}{\Gamma \vdash_{\Delta}^{\mathcal{L}} \lambda x. t :_{\mathcal{I}} A \rightarrow B}$$

$$\frac{\Gamma \vdash_{\Delta, \kappa}^{\mathcal{L}} t :_{\mathcal{I}} \triangleright^{\kappa} A \quad \Gamma \vdash_{\Delta}}{\Gamma \vdash_{\Delta}^{\mathcal{L}} \text{prev} \kappa. t :_{\mathcal{I}} \forall \kappa. A}$$

Reduction rules

$$\text{freeze}_{l'}^{\kappa}(\text{next}^{\kappa} l.t) \rightarrow t [l'/l]$$

$$\text{next}^{\kappa} l.(\text{freeze}_l^{\kappa} t) \rightarrow t$$

$$\text{prev}_{\kappa}.\text{next}^{\kappa} l.t \rightarrow \Lambda_{\kappa}.t[\text{freeze}_l^{\kappa} x \mapsto (\text{prev}_{\kappa}.x)[\kappa]]$$

Reduction rules

$$\begin{aligned} & \text{freeze}_j^{\kappa}(\text{next}^{\kappa} l.t) \rightarrow t [l'/l] \\ \text{next}^{\kappa} \xi [x \leftarrow \text{next}^{\kappa} \xi.u].A &= \text{next}^{\kappa} \xi.A [u/x] \end{aligned}$$

$$\begin{aligned} \text{next}^{\kappa} l.(\text{freeze}_j^{\kappa} t) &\rightarrow t \\ \text{next}^{\kappa} \xi [x \leftarrow t].x &= t \end{aligned}$$

$$\begin{aligned} \text{prev}_{\kappa}.\text{next}^{\kappa} l.t &\rightarrow \Lambda_{\kappa}.t[\text{freeze}_j^{\kappa} x \mapsto (\text{prev}_{\kappa}.x)[\kappa]] \\ \text{prev}_{\kappa}.\text{next}^{\kappa} \xi.t &= \Lambda_{\kappa}.t(\text{adv}^{\kappa}(\xi)) \end{aligned}$$

$$\text{adv}^{\kappa}([x_1 \leftarrow t_1, \dots, x_n \leftarrow t_n]) = [(\text{prev}_{\kappa}.t_1)[\kappa]/x_1, \dots, (\text{prev}_{\kappa}.t_n)[\kappa]/x_n]$$

η -rule for \triangleright

$$\text{next}^{\kappa} [x \leftarrow t] \xi.u [\text{next}^{\kappa} x/y] = \text{next}^{\kappa} [x \leftarrow t] \xi.u [t/y]$$

η -rule for \triangleright

$$\text{next}^{\kappa} [x \leftarrow t] \xi.u [\text{next}^{\kappa} x/y] = \text{next}^{\kappa} [x \leftarrow t] \xi.u [t/y]$$

$$\text{next}^{\kappa} /.(\text{freeze}_j^{\kappa} t) \rightarrow t$$

η -rule for \triangleright

$$\text{next}^{\kappa} [x \leftarrow t] \xi.u [\text{next}^{\kappa} x/y] = \text{next}^{\kappa} [x \leftarrow t] \xi.u [t/y]$$

$$\text{next}^{\kappa} l.(\text{freeze}_{l'}^{\kappa} t) \rightarrow t \quad l \notin \text{fl}(t)$$

η -rule for \triangleright

$$\text{next}^{\kappa} [x \leftarrow t] \xi.u [\text{next}^{\kappa} x/y] = \text{next}^{\kappa} [x \leftarrow t] \xi.u [t/y]$$

$$\text{next}^{\kappa} l.(\text{freeze}_{l'}^{\kappa} t) \rightarrow t \quad l \notin \text{fl}(t)$$

This rule breaks confluence!

Future Work

What we have

- ▶ confluence proof
- ▶ strong normalisation without dependent types
- ▶ completeness w.r.t. delayed substitution calculus

Future Work

What we have

- ▶ confluence proof
- ▶ strong normalisation without dependent types
- ▶ completeness w.r.t. delayed substitution calculus

What is missing

- ▶ strong normalisation of dependently typed calculus
- ▶ soundness w.r.t. delayed substitution calculus