Rewrite Semantics for Guarded Recursion in Type Theory

Patrick Bahr

IT University of Copenhagen

joint work with Rasmus Møgelberg and Hans Bugge Grathwohl

Overview

Guarded Recursive Types

Dependent Types

Reduction Semantics

Guarded Recursive Types

H. Nakano. A modality for recursion. In: LICS, 2000

Guarded Recursion

- type modality $\triangleright$ (pronounced "later")
- $\triangleright$ is an applicative functor

$$\begin{aligned} \text{next} &: A \to \triangleright A \\ \circledast &: \triangleright(A \to B) \to \triangleright A \to \triangleright B \end{aligned}$$

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

Example

$$
\begin{aligned}
\mathsf{Str} &= \mu X.\mathsf{Nat} \times \triangleright X \\
\mathsf{cons}\colon &\ \mathsf{Nat} \to \triangleright\mathsf{Str} \to \mathsf{Str} \\
\mathsf{cons} &= \lambda x.\lambda y.\langle x, y\rangle \\
\mathsf{nats}\colon &\ \mathsf{Nat} \to \mathsf{Str} \\
\mathsf{nats} &= \mathsf{fix}(\lambda f\, n.\mathsf{cons}\, n\, (f \circledast (\mathsf{next}(n+1))))
\end{aligned}
$$

$$
\begin{aligned}
\mathsf{inter}\colon &\ \mathsf{Str} \to \triangleright\mathsf{Str} \to \mathsf{Str} \\
\mathsf{inter} &= \mathsf{fix}(\lambda f\, s\, t.\mathsf{cons}\, (\pi_1 s)\, (f \circledast t \circledast (\mathsf{next}(\pi_2 s))))) \\
\mathsf{foo}\colon &\ \mathsf{Str} \\
\mathsf{foo} &= \mathsf{fix}(\lambda x.\mathsf{inter}\, (\mathsf{nats}\, 0), x)
\end{aligned}
$$

Motivation

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

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 $\triangleright$

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

Combining Π and $\triangleright$

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

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

Combining Π and ▷

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

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

Combining Π and $\triangleright$

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

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

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

Combining Π and $\triangleright$

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

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

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

Delayed Substitutions

Instead of

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

we have

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

Delayed Substitutions

Instead of

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

we have

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

In general

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

Delayed Substitutions

Instead of

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

we have

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

In general

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

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

Equalities

$$\triangleright \; [x \leftarrow \mathsf{next} \quad u] \,.A = \triangleright \quad A\,[u/x]$$

Equalities

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

Equalities

$$
\begin{aligned}
\triangleright\xi\,[x \leftarrow \mathsf{next}\xi.u]\,.A &= \triangleright\xi.A\,[u/x] & \\
\triangleright\,\xi\,[x \leftarrow u]\,.A &= \triangleright\xi.A & \text{if } x \notin \mathsf{fv}(A) \\
\triangleright\,\xi\,[x \leftarrow u, y \leftarrow v]\,\xi'.A &= \triangleright\xi\,[y \leftarrow v, x \leftarrow u]\,\xi'.A & \text{if } \ldots
\end{aligned}
$$

Equalities

$$\begin{aligned}
\triangleright\xi\left[x \leftarrow \mathsf{next}\xi.u\right].A &= \triangleright\xi.A\left[u/x\right] \\
\triangleright\,\xi\left[x \leftarrow u\right].A &= \triangleright\xi.A && \text{if } x \notin \mathsf{fv}(A) \\
\triangleright\,\xi\left[x \leftarrow u, y \leftarrow v\right]\xi'.A &= \triangleright\xi\left[y \leftarrow v, x \leftarrow u\right]\xi'.A && \text{if } \ldots
\end{aligned}$$

$$\begin{aligned}
\mathsf{next}\xi\left[x \leftarrow \mathsf{next}\xi.u\right].t &= \mathsf{next}\xi.t\left[u/x\right] \\
\mathsf{next}\xi\left[x \leftarrow u\right].t &= \mathsf{next}\xi.t && \text{if } x \notin \mathsf{fv}(t) \\
\mathsf{next}\xi\left[x \leftarrow u, y \leftarrow v\right]\xi'.t &= \mathsf{next}\xi\left[y \leftarrow v, x \leftarrow u\right]\xi'.t && \text{if } \ldots
\end{aligned}$$

Typing rule

Simple Case

$$\frac{\Gamma, x : A \vdash t : B \qquad \Gamma \vdash u : \triangleright A}{\Gamma \vdash \mathsf{next}\left[x \leftarrow u\right].t : \triangleright\left[x \leftarrow u\right].B}$$

Typing rule

Simple Case

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

In General

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

Applicative Structure

Applicative structure can be defined in terms of delayed substitutions:

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

Applicative Structure

Applicative structure can be defined in terms of delayed substitutions:

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

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

Applicative Functor Laws

We need to add the following equality

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

Applicative Functor Laws

We need to add the following equality

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

We can then derive the applicative functor laws:

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

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!

$$\mathsf{fix}\, t = t(\mathsf{next}(\mathsf{fix}\, t))$$

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

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

Problems with Confluence

$$\mathsf{next}\xi\,[x \leftarrow \mathsf{next}\xi.s]\,.t = \mathsf{next}\xi.t\,[s/x]$$

Problems with Confluence

$$\mathsf{next}\xi\,[x \leftarrow \mathsf{next}\xi.s]\,.t \rightarrow \mathsf{next}\xi.t\,[s/x]$$

Problems with Confluence

$$\mathsf{next}\xi\,[x \leftarrow \mathsf{next}\xi.s]\,.t \rightarrow \mathsf{next}\xi.t\,[s/x]$$

Problems with Confluence

$$\mathsf{next}\xi\left[x \leftarrow \mathsf{next}\xi.s\right].t \rightarrow \mathsf{next}\xi.t\left[s/x\right]$$

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

$$\begin{aligned} t &\rightarrow \mathsf{next}\left[x_1 \leftarrow y\right].x_1 0 \\ t &\rightarrow \mathsf{next}\left[x_1 \leftarrow y, x_2 \leftarrow \mathsf{next}.0\right].x_1 x_2 \end{aligned}$$

Alternative Calculus without Delayed Substitutions

Idea

- controlled conversion $\text{prev} \colon \triangleright A \to A$.

Alternative Calculus without Delayed Substitutions

Idea

- controlled conversion prev : $\triangleright A \to A$.
- $\mathsf{next}\,[x \leftarrow t]\,.u \quad \rightsquigarrow \quad \mathsf{next}\;\; u\,[\mathsf{prev}\; t/x]$
- $\triangleright\,[x \leftarrow t]\,.A \quad \rightsquigarrow \quad \triangleright\;\; A\,[\mathsf{prev}\; t/x]$

Alternative Calculus without Delayed Substitutions

Idea

- controlled conversion $\mathsf{prev} : \triangleright A \to A$.
- $\mathsf{next}\,[x \leftarrow t]\,.u \quad \rightsquigarrow \quad \mathsf{next}\,l.u\,[\mathsf{prev}_l t/x]$
- $\triangleright\,[x \leftarrow t]\,.A \quad \rightsquigarrow \quad \triangleright l.A\,[\mathsf{prev}_l t/x]$

Alternative Calculus without Delayed Substitutions

Idea

- controlled conversion prev : $\triangleright A \to A$.
- $\text{next}\,[x \leftarrow t]\,.u \quad \rightsquigarrow \quad \text{next}\,l.u\,[\text{prev}_l t/x]$
- $\triangleright\,[x \leftarrow t]\,.A \quad \rightsquigarrow \quad \triangleright l.A\,[\text{prev}_l t/x]$

$$\frac{\Gamma \vdash^{\mathcal{L}} t :_{\mathcal{I}} \triangleright l.A \qquad l \in \mathcal{L}}{\Gamma \vdash^{\mathcal{L}} \text{prev}_l t :_{\mathcal{I},l} A} \qquad \frac{\Gamma \vdash^{\mathcal{L},l} t :_{\mathcal{I},l} A \qquad \Gamma \vdash^{\mathcal{L}}}{\Gamma \vdash^{\mathcal{L}} \text{next}\,l.t :_{\mathcal{I}} \triangleright l.A}$$

Alternative Calculus without Delayed Substitutions

Idea

- controlled conversion $\mathsf{prev} : \triangleright A \to A$.
- $\mathsf{next}\,[x \leftarrow t]\,.u \quad \rightsquigarrow \quad \mathsf{next}\,l.u\,[\mathsf{prev}_l\,t/x]$
- $\triangleright\,[x \leftarrow t]\,.A \quad \rightsquigarrow \quad \triangleright l.A\,[\mathsf{prev}_l\,t/x]$

$$\frac{\Gamma \vdash^{\mathcal{L}} t :_{\mathcal{I}} \triangleright l.A \qquad l \in \mathcal{L}}{\Gamma \vdash^{\mathcal{L}} \mathsf{prev}_l\,t :_{\mathcal{I},l} A} \qquad \frac{\Gamma \vdash^{\mathcal{L},l} t :_{\mathcal{I},l} A \qquad \Gamma \vdash^{\mathcal{L}}}{\Gamma \vdash^{\mathcal{L}} \mathsf{next}\,l.t :_{\mathcal{I}} \triangleright l.A}$$

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

Reduction rules

$$\mathsf{prev}_{l'}(\mathsf{next}\,l.t) \to t\,[l'/l]$$

$$\mathsf{next}\,l.(\mathsf{prev}_l t) \to t \qquad l \notin \mathsf{fl}(t)$$

Reduction rules

$$\mathsf{prev}_{l'}(\mathsf{next}\,l.t) \rightarrow t\,[l'/l]$$

$$\mathsf{next}\xi\,[x \leftarrow \mathsf{next}\xi.u]\,.A = \mathsf{next}\xi.A\,[u/x]$$

$$\mathsf{next}\,l.(\mathsf{prev}_l t) \rightarrow t \qquad l \notin \mathsf{fl}(t)$$

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

η-rule for $\triangleright$

$$\mathsf{next}\,[x \leftarrow t]\,\xi.u\,[\mathsf{next}\,x/y] = \mathsf{next}\,[x \leftarrow t]\,\xi.u\,[t/y]$$

η-rule for $\triangleright$

$$\mathsf{next}\,[x \leftarrow t]\,\xi.u\,[\mathsf{next}\,x/y] = \mathsf{next}\,[x \leftarrow t]\,\xi.u\,[t/y]$$

$$\mathsf{next}\,l.(\mathsf{prev}_l\,t) \rightarrow t$$

η-rule for $\triangleright$

$$\mathsf{next}\,[x \leftarrow t]\,\xi.u\,[\mathsf{next}\,x/y] = \mathsf{next}\,[x \leftarrow t]\,\xi.u\,[t/y]$$

$$\mathsf{next}\,l.(\mathsf{prev}_{l'}\,t) \to t \qquad l \notin \mathsf{fl}(t)$$

η-rule for $\triangleright$

$$\mathsf{next}\,[x \leftarrow t]\,\xi.u\,[\mathsf{next}\,x/y] = \mathsf{next}\,[x \leftarrow t]\,\xi.u\,[t/y]$$

$$\mathsf{next}\,l.(\mathsf{prev}_{l'}\,t) \to t \qquad l \notin \mathsf{fl}(t)$$

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