First steps in synthetic guarded domain theory: step-indexing in the topos of trees

Lars Birkedal [1] Rasmus Ejlers Møgelberg [1]
Kristian Støvring [2] Jan Schwinghammer [3]

[1]IT University of Copenhagen

[2]DIKU, University of Copenhagen

[3]Saarland University

June 21, 2011

Overview

- A higher order dependent type theory with guarded recursion
- Model: the topos of trees
  - Combining dependent and guarded recursive types in the topos of trees
- Example: modeling higher order store
- Relations to metric spaces and step-indexing
  - Extending complete bisected ultrametric spaces with useful type constructors
  - Guarded recursion as the principle that makes step-indexing work

The topos of trees

- $\mathcal{S} = \mathbf{Set}^{\omega^{\mathrm{op}}}$
- Objects

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

- Morphism

$$\begin{array}{ccccccc} X(1) & \longleftarrow & X(2) & \longleftarrow & X(3) & \longleftarrow & \dots \\ \downarrow{\scriptstyle f_1} & & \downarrow{\scriptstyle f_2} & & \downarrow{\scriptstyle f_3} & & \\ Y(1) & \longleftarrow & Y(2) & \longleftarrow & Y(3) & \longleftarrow & \dots \end{array}$$

- Example: object of streams of natural numbers

$$\mathbb{N} \xleftarrow{\pi} \mathbb{N}^2 \xleftarrow{\pi} \mathbb{N}^3 \xleftarrow{\pi} \dots$$

- For $x \in X(m)$ define

$$x|_n = r_n \circ \cdots \circ r_{m-1}(x).$$

An endofunctor

- Define $\blacktriangleright X$ ("later X")

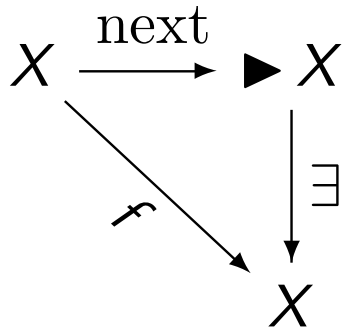
$$\{*\} \longleftarrow X(1) \longleftarrow X(2) \longleftarrow \dots$$

- Preserves limits, but not colimits
- Define $\mathrm{next} : X \to \blacktriangleright X$

$$\begin{array}{ccccccc}
X(1) & \xleftarrow{r_1} & X(2) & \xleftarrow{r_2} & X(3) & \longleftarrow & \dots \\
\downarrow & & \downarrow{\scriptstyle r_1} & & \downarrow{\scriptstyle r_2} & & \\
\{*\} & \longleftarrow & X(1) & \xleftarrow{r_1} & X(2) & \longleftarrow & \dots
\end{array}$$

Fixed points

- A morphism factoring through next is called *contractive*



- Contractive morphisms have *unique* fixed points
- Fixed point operator

$$\mathrm{fix}_X : (\blacktriangleright X \to X) \to X$$

Internal logic

- Toposes model higher order logic

$$\phi, \psi ::= [s = t] \mid \phi \wedge \psi \mid \phi \vee \psi \mid \phi \to \psi \mid$$
$$\exists x : X.\phi \mid \forall x : X.\phi \mid \exists \psi : \mathsf{Pred}(X).\phi \mid \forall \psi : \mathsf{Pred}(X).\phi$$

- Predicates interpreted as subobjects

$$\begin{array}{ccccccc} [\![\phi]\!](1) & \longleftarrow & [\![\phi]\!](2) & \longleftarrow & [\![\phi]\!](3) & \longleftarrow & \dots \\ \big| \bigcap & & \big| \bigcap & & \big| \bigcap & & \\ X(1) & \longleftarrow & X(2) & \longleftarrow & X(3) & \longleftarrow & \dots \end{array}$$

- Subobject classifier Ω

$$\{0,1\} \xleftarrow{\min(1,-)} \{0,1,2\} \xleftarrow{\min(2,-)} \{0,1,2,3\} \longleftarrow \dots$$

- Think of Ω as type of propositions

Forcing relation

- Given $\varphi : \mathrm{Pred}(X)$, $n \in \omega$, and $\alpha \in X(n)$
- Define $n \models \varphi(\alpha)$ iff $\alpha \in [\![\varphi]\!](n)$
- Kripke-Joyal semantics

$$n \models (\varphi \vee \psi)(\alpha) \iff n \models \varphi(\alpha) \vee n \models \psi(\alpha)$$
$$n \models (\varphi \Longrightarrow \psi)(\alpha) \iff \forall k \leq n.\, k \models \varphi(\alpha|_k) \Longrightarrow k \models \psi(\alpha|_k)$$

An operator on predicates

- Define $\triangleright : \Omega \to \Omega$

$$\underset{n}{\triangleright} = \min(n, (-) + 1) : \{0, \ldots, n\} \to \{0, \ldots, n\}$$

- $1 \models \triangleright\, \varphi(\alpha)$ and

$$n + 1 \models \triangleright\, \varphi(\alpha) \iff n \models \varphi(\alpha|_n).$$

- Connection to $\blacktriangleright$

$$\begin{array}{ccc} \triangleright\, m & \longrightarrow & \blacktriangleright A \\ \downarrow & \lrcorner & \downarrow \blacktriangleright m \\ X & \xrightarrow{\text{next}} & \blacktriangleright X \end{array}$$

Recursive predicates

- $\triangleright : \Omega \to \Omega$ is contractive
- If f or g is contractive so is fg
- Suppose $r : \mathrm{Pred}(X) \vdash \phi : \mathrm{Pred}(X)$ has every occurrence of r guarded by $\triangleright$
- Then ϕ contractive
- So has unique fixed point $\mu r.\phi : \mathrm{Pred}(X)$

Internal logic

- Monotonicity

$$\forall p : \Omega.\, p \Longrightarrow \triangleright\, p$$

- Löb rule

$$\forall p : \Omega.\, (\triangleright\, p \Longrightarrow p) \Longrightarrow p.$$

- Internal notion of contractiveness

$$\mathrm{Contr}(f) \overset{\mathrm{def}}{\Longleftrightarrow} \forall x, x' : X.\, \triangleright (x = x') \Longrightarrow f(x) = f(x').$$

- Externally contractive implies internally contractive
- Internal Banach Fixed-Point Theorem

$$(\exists x : X.\, \top) \wedge \mathrm{Contr}(f) \Longrightarrow \exists! x : X.\, f(x) = x.$$

- Follows from

$$\mathrm{Contr}(f) \Longrightarrow \exists n : N. \forall x, x' : X.\, f^n(x) = f^n(x').$$

Recursive domain equations

- Recall $F : \mathcal{S} \to \mathcal{S}$ *strong* if exists

$$F_{X,Y} : Y^X \to FY^{FX}$$

- Say F locally contractive if each $F_{X,Y}$ contractive
- Generalises to mixed variance functors of many variables
- **Theorem:** If $F : \mathcal{S}^{\mathrm{op}} \times \mathcal{S} \to \mathcal{S}$ is locally contractive then there exists X such that $F(X, X) \cong X$. Moreover, X unique up to isomorphism
- Solutions are initial dialgebras

Modeling dependent types

- Recall that any topos models dependent type theory
- E.g. recall rules

$$\frac{\Gamma, i : I \vdash A : \mathsf{Type}}{\Gamma \vdash \prod_{i:I} A : \mathsf{Type}} \qquad \frac{\Gamma, i : I \vdash A : \mathsf{Type}}{\Gamma \vdash \sum_{i:I} A : \mathsf{Type}}$$

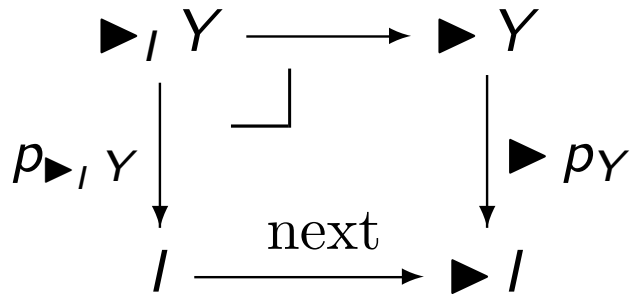
- Combined with subset types

$$\frac{\Gamma, x : A \vdash \phi : \mathsf{Prop}}{\Gamma \vdash \{x : A \mid \phi\} : \mathsf{Type}}$$

- Will extend this with guarded recursive types

Generalising ▶ to dependent types

- Dependent type judgements $\Gamma \vdash A$ interpreted as objects of $\mathcal{S}/[\![\Gamma]\!]$
- $\blacktriangleright_I : \mathcal{S}/I \to \mathcal{S}/I$ maps $p_Y : Y \to I$ to $p_{\blacktriangleright_I Y}$:



- Behaves well wrt. reindexing: can use ▶ as type constructor in dependent internal type theory
- Results on domain equations generalise to slices

Functorial types

$$A(\vec{X}) ::= X_i \mid C \mid A(\vec{X}) \times A(\vec{X}) \mid A(\vec{X}) \to A(\vec{X}) \mid$$
$$\prod_{i:I} A(\vec{X}) \mid \sum_{i:I} A(\vec{X}) \mid \{a : A(\vec{X}) \mid \phi_{\vec{X}}(a)\} \mid$$
$$\blacktriangleright A(\vec{X}) \mid \mu X.A(\vec{X}, X)$$

- where
$$\phi_{\vec{X}_0}(a) \Longrightarrow \phi_{\vec{X}_1}(A(\vec{f})(a))$$
provably holds for all $\vec{f}, a$.
- Recursive type well-defined if X only occurs under $\blacktriangleright$

Example application

Example application

- Define interpretation of CBV language with higher order store entirely inside internal language of $\mathcal{S}$
- Previous models
  - Step-indexing
  - Using domain equations solved in metric spaces
- Here:
  - Everything in one universe
  - Simple set-like interpretation
  - No explicit steps but $\blacktriangleright$ operators certain places
- We see guarded recursion as the principle that makes step-indexed models work

The language

- Types

$$\tau ::= 1 \mid \tau_1 \times \tau_2 \mid \mu\alpha.\tau \mid \forall\alpha.\tau \mid \alpha \mid \tau_1 \to \tau_2 \mid \mathsf{ref}\,\tau$$

- Standard small-step operational CBV semantics
- Sets of types, terms and values can be read as definitions in the internal language of $\mathcal{S}$
- Likewise sets Store, $\mathrm{Config} = \mathrm{Term} \times \mathrm{Store}$
- Non-standard encoding of transitive closure of operational semantics

A recursively defined universe of types

- Idea: interpret types as predicates on Value
- But the predicate should depend on the world
- Would like to solve (but can not)

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

- Can solve this equation

$$\widehat{\mathcal{T}} = \mu X.\, \blacktriangleright((N \to_{\mathrm{fin}} X) \to_{\mathrm{mon}} \mathcal{P}(\mathrm{Value}))$$

Semantics, overview

$$\mathcal{W} = \mathbb{N} \to_{\mathrm{fin}} \widehat{\mathcal{T}}$$
$$\mathcal{T} = \mathcal{W} \to_{\mathrm{mon}} \mathcal{P}(\mathrm{Value})$$

- Define $[\![\tau]\!] : \mathrm{TEnv}(\tau) \to \mathcal{T}$ by induction on τ
- Simple set-like definitions except for $\mu\alpha.\tau$, ref τ

$$[\![\tau_1 \times \tau_2]\!]\varphi = \lambda w.\, \{(v_1,\, v_2) \mid v_1 \in [\![\tau_1]\!]\varphi(w) \wedge v_2 \in [\![\tau_2]\!]\varphi(w)\}$$
$$[\![\mu\alpha.\tau]\!]\varphi = \mathit{fix}\,(\lambda\nu.\, \lambda w.\, \{\,\mathsf{fold}\, v \mid \triangleright(v \in [\![\tau]\!]\varphi[\alpha \mapsto \nu]\,(w))\})$$

- **Theorem.** If $\vdash t : \tau$, then for all $w \in \mathcal{W}$ we have $t \in \mathit{comp}([\![\tau]\!]\emptyset)(w)$.
- where

$$\mathit{comp} : \mathcal{T} \to \mathcal{T}^{\mathrm{c}}$$
$$\mathcal{T}^{\mathrm{c}} = \mathcal{W} \to \mathcal{P}(\mathrm{Term})$$

Partial correctness predicate

- Define $\mathrm{eval} : \mathcal{P}(\mathrm{Term} \times \mathrm{Store} \times \mathcal{P}(\mathrm{Value} \times \mathrm{Store}))$,

$$\begin{aligned}
&\mathrm{eval}(t, s, Q) \\
&\quad \overset{\mathrm{def}}{\Longleftrightarrow} (t \in \mathrm{Value} \ \wedge \ Q(t, s)) \vee \\
&\qquad (\exists t_1 : \mathrm{Term}, s_1 : \mathrm{Store}. \\
&\qquad\quad \mathrm{step}((t, s), (t_1, s_1)) \ \wedge \ \triangleright \mathrm{eval}(t_1, s_1, Q))
\end{aligned}$$

- $n \models \mathrm{eval}(t, s, Q)$ iff the following property holds: for all $m < n$, if (t, s) reduces to (v, s') in m steps, then $(n - m) \models Q(v, s')$.

Conclusions to example

- Composite interpretation

  Source language $\longrightarrow$ Internal language of $\mathcal{S}$ $\longrightarrow$ Set theory

  is essentially a known step-indexing-model
- We see guarded recursion as the principle that makes step-indexed models work
- Internal language expressive enough for advanced example
- This could be a way to implement such models: need extensions of e.g. Coq with guarded recursion
- This is just one example: guarded recursion occurs many other places in computer science

Conclusions

- Topos of trees models recursion on
  - term level
  - predicate level
  - type level
- Recursive types can be combined with dependent types
- Powerful internal language sufficient for modeling programming languages with higher order store
- Factorize step indexing models through guarded recursion
- Relates to ultrametric spaces, but gives a richer universe