Two Guarded Recursive Powerdomains for Applicative Simulation

Rasmus Ejlers Møgelberg Andrea Vezzosi

IT University of Copenhagen

Types 2022

How to model programming languages in type theory?

- λ-calculus with non-determinism

$$M, N ::= M\, N \mid \lambda x.M \mid M \text{ or } N$$

- May-convergence predicate

$$\frac{}{\lambda x.M \Downarrow_\Diamond \lambda x.M}$$

$$\frac{M \Downarrow_\Diamond \lambda x.M' \qquad N \Downarrow_\Diamond V' \qquad M'[V'/x] \Downarrow_\Diamond V}{MN \Downarrow_\Diamond V}$$

$$\frac{M \Downarrow_\Diamond V}{M \text{ or } N \Downarrow_\Diamond V} \qquad \frac{N \Downarrow_\Diamond V}{M \text{ or } N \Downarrow_\Diamond V}$$

How to model programming languages in type theory?

- Idea: Model non-determinism using powersets

$$[\![M]\!] : \mathsf{P}_{\mathsf{f}}(\mathsf{Val})$$

- Model divergence using coinductive partiality monad
  $\mathsf{L}A \simeq A + \mathsf{L}A$
- How to combine these?

Overview

- Guarded recursion
- Finite powerset as a HIT
- A powerdomain for may-convergence
  - Applicative may-similarity is a congruence
- A powerdomain for must-convergence
  - Applicative must-similarity is a congruence

Overview

- Guarded recursion
- Finite powerset as a HIT
- A powerdomain for may-convergence
  - Applicative may-similarity is a congruence
- A powerdomain for must-convergence
  - Applicative must-similarity is a congruence
- Everything done on paper in Clocked Cubical Type Theory
- Should be possible to formalise this in Guarded Cubical Agda

Guarded recursion

- Work in Clocked Cubical Type Theory[1]
- Guarded fixed point operator

$$\mathsf{fix}^\kappa : (\triangleright^\kappa A \to A) \to A$$

- Guarded recursive types

$$\mathsf{L}^\kappa A \simeq A + \triangleright^\kappa \mathsf{L}^\kappa A$$

- Define coinductive *partiality monad* $\mathsf{L}(A) = \forall\kappa.\mathsf{L}^\kappa A$

$$\mathsf{L}(A) \simeq A + \mathsf{L}(A)$$

- Both L and L^κ are monads on Set

[1] Kristensen, Møgelberg, Vezzosi: *Greatest HITs: Higher inductive types in coinductive definitions via induction under clocks*, LICS 2022. arXiv:2102.01969

Finite (non-empty) powersets as HITs

$$
\begin{aligned}
\{-\} &: A \to \mathsf{P_f}(A) \\
\cup &: \mathsf{P_f}(A) \to \mathsf{P_f}(A) \to \mathsf{P_f}(A) \\
\mathsf{assoc} &: \Pi(X\ Y\ Z : \mathsf{P_f}(A)).\, X \cup (Y \cup Z) = (X \cup Y) \cup Z \\
\mathsf{comm} &: \Pi(X\ Y : \mathsf{P_f}(A)).\, X \cup Y = Y \cup X \\
\mathsf{idem} &: \Pi(X : \mathsf{P_f}(A)).\, X \cup X = X \\
\mathsf{trunc} &: \mathsf{isSet}(\mathsf{P_f}(A))
\end{aligned}
$$

- $\mathsf{P_f}(A)$ is free semilattice structure on A
- $\mathsf{P_f}$ is a monad on Set

A guarded powerdomain for may-equivalence

- Define may-powerdomain

$$\mathsf{P}^{\kappa}_{\Diamond}(A) \simeq \mathsf{P}_{\mathsf{f}}(A + \triangleright^{\kappa}\mathsf{P}^{\kappa}_{\Diamond}(A))$$

- Operations

$$\begin{aligned} \cup &: \mathsf{P}^{\kappa}_{\Diamond}(A) \to \mathsf{P}^{\kappa}_{\Diamond}(A) \to \mathsf{P}^{\kappa}_{\Diamond}(A) \\ \mathsf{now}_{\Diamond} &: A \to \mathsf{P}^{\kappa}_{\Diamond}(A) \\ \mathsf{step}_{\Diamond} &: \triangleright^{\kappa}\mathsf{P}^{\kappa}_{\Diamond}(A) \to \mathsf{P}^{\kappa}_{\Diamond}(A) \end{aligned}$$

- Free delay-algebra and semilattice structure on A
- So monad on Set

Applicative similarity

- R is an applicative may-simulation if $M\,R\,N$ and $M \Downarrow_{\Diamond} \lambda x.M'$ implies

$$\exists N'.\, N \Downarrow_{\Diamond} \lambda y.N' \wedge (\forall (V : \mathsf{Val}).M'[V/x]\, R\, N'[V/x])$$

- $\leq_{\Diamond}$ is greatest (coinductive) applicative may-simulation

Proving applicative similarity is a congruence

- Adaptation of Pitts' method
- A semantic domain

$$\mathsf{SVal}^\kappa \stackrel{\text{def}}{=} \triangleright^\kappa(\mathsf{SVal}^\kappa \to \mathsf{P}^\kappa_\lozenge(\mathsf{SVal}^\kappa))$$
$$D^\kappa \stackrel{\text{def}}{=} \mathsf{P}^\kappa_\lozenge(\mathsf{SVal}^\kappa)$$

- Define

$$[\![-]\!]^\kappa : \Lambda \to D^\kappa$$
$$\preceq^\kappa : D^\kappa \times \Lambda \to \mathsf{Prop}$$

- **Lemma.** If M and N are closed terms then $M \leq_\lozenge N$ is equivalent to $\forall \kappa. [\![M]\!]^\kappa \preceq^\kappa N$.
- **Theorem.** If $M \leq_\lozenge N$ and $C[-]$ is a context then also $C[M] \leq_\lozenge C[N]$.

Must-convergence predicate

Define $\Downarrow_\Box : \Lambda \times \mathrm{P_f}(\mathrm{Val}) \to \mathrm{Prop}$ as

$$\frac{M \Downarrow_\Box X \qquad N \Downarrow_\Box Y}{M \text{ or } N \Downarrow_\Box X \cup Y} \qquad\qquad \frac{}{\lambda x.M \Downarrow_\Box \{\lambda x.M\}}$$

$$\frac{N \Downarrow_\Box Y \qquad \begin{array}{c} M \Downarrow_\Box X \\ \forall (\lambda y.M') \in X, V \in Y.\ M'[V/y] \Downarrow_\Box Z_{\lambda y.M',V} \end{array}}{M\,N \Downarrow_\Box \cup_{V' \in X, V \in Y} Z_{V',V}}$$

Powerdomain definition

- Define

$$\mathsf{P}^{\kappa}_{\square}(A) \stackrel{\text{def}}{=} \mathsf{L}^{\kappa}(\mathsf{P}_{\mathsf{f}}(A)) \\ \simeq \mathsf{P}_{\mathsf{f}}(A) + \triangleright^{\kappa}\mathsf{P}^{\kappa}_{\square}(A)$$

- Semilattice structure

$$\mathsf{now}_{\mathsf{L}}\,(x) \cup \mathsf{now}_{\mathsf{L}}\,(y) \stackrel{\text{def}}{=} \mathsf{now}_{\mathsf{L}}\,(x \cup y)$$

$$\mathsf{step}_{\square}(x) \cup \mathsf{step}_{\square}(y) \stackrel{\text{def}}{=} \mathsf{step}_{\square}(\lambda(\alpha : \kappa).x\,[\alpha] \cup y\,[\alpha])$$

$$\mathsf{step}_{\square}(x) \cup \mathsf{now}_{\mathsf{L}}\,(y) \stackrel{\text{def}}{=} \mathsf{step}_{\square}(\lambda(\alpha : \kappa).(x\,[\alpha] \cup \mathsf{now}_{\mathsf{L}}\,(y)))$$

Powerdomain definition

- Define

$$
\begin{aligned}
\mathsf{P}^{\kappa}_{\Box}(A) &\stackrel{\text{def}}{=} \mathsf{L}^{\kappa}(\mathsf{P}_{\mathsf{f}}(A)) \\
&\simeq \mathsf{P}_{\mathsf{f}}(A) + \triangleright^{\kappa}\mathsf{P}^{\kappa}_{\Box}(A)
\end{aligned}
$$

- Semilattice structure

$$
\begin{aligned}
\mathsf{now}_{\mathsf{L}}\,(x) \cup \mathsf{now}_{\mathsf{L}}\,(y) &\stackrel{\text{def}}{=} \mathsf{now}_{\mathsf{L}}\,(x \cup y) \\
\mathsf{step}_{\Box}(x) \cup \mathsf{step}_{\Box}(y) &\stackrel{\text{def}}{=} \mathsf{step}_{\Box}(\lambda(\alpha:\kappa).x\,[\alpha] \cup y\,[\alpha]) \\
\mathsf{step}_{\Box}(x) \cup \mathsf{now}_{\mathsf{L}}\,(y) &\stackrel{\text{def}}{=} \mathsf{step}_{\Box}(\lambda(\alpha:\kappa).(x\,[\alpha] \cup \mathsf{now}_{\mathsf{L}}\,(y)))
\end{aligned}
$$

- Not a monad!
- Bind is not associative
- Proved must-applicative similarity a congruence

Summary

- Defined two guarded powerdomains and related these to may- and must-similarity
- Unfortunately $P^{\kappa}_{\Box}$ is not a monad
- It is a monad up to weak bisimilarity
- Future work:
  - A theory of algebraic effects and guarded recursion