Amb breaks well-pointedness, ground amb doesn't

Paul Blain Levy

School of Computer Science, University of Birmingham, UK

Nondeterministic Operators

- We can extend a functional language with:

  binary erratic choice M `or` N

  countable erratic choice `choose` $n \in \mathbb{N}.\ M_n$

  ambiguous choice M `amb` N

- To evaluate M `amb` N, we run M and N on an arbitrary fair scheduler, and return whatever we get first.
- Thus M `amb` N can diverge iff M and N can both diverge.

Small Language

A call-by-name language, with two ground types, and (unary) sum types.

$$\textbf{Types} \quad A \;::=\; \texttt{bool} \mid 1 \mid LA$$

$$\begin{array}{lll}
\textbf{Terms} \quad M & ::= & \texttt{x} \mid \texttt{rec x.}\ M \mid \\
 & & M\ \texttt{or}\ M \mid M\ \texttt{amb}\ M \mid \\
 & & \texttt{true} \mid \texttt{false} \mid \texttt{if}\ M\ \texttt{then}\ M\ \texttt{else}\ M \mid \\
 & & \texttt{top} \mid M; M \mid \\
 & & \texttt{up}\ M \mid \texttt{pm}\ M\ \texttt{as up x.}\ M
\end{array}$$

Operational Semantics

- **Terminal Terms**

$$T \quad ::= \quad \mathtt{true} \mid \mathtt{false} \mid \mathtt{top} \mid \mathtt{up}\, M$$

- Remember: in a call-by-name sum type, we don't evaluate under the constructor.
- $M \Downarrow T$ is defined inductively.
- $M \Uparrow$ is defined coinductively.

(Crude) Meaning Of A Type

- For each type B, we define the set $[B]$ by induction on B:

$$\begin{array}{rcl} [\mathtt{bool}] & = & \mathcal{P}(\{\mathsf{true}, \mathsf{false}, \bot\}) \\ [1] & = & \mathcal{P}(\{\top, \bot\}) \\ [LA] & = & \mathcal{P}([A]_\bot) \end{array}$$

- Could restrict to nonempty sets—doesn't matter.
- For each closed term $M : B$, we define the operational meaning $[M] \in [B]$, by induction on B.

  E.g. $[M] \stackrel{\text{def}}{=} \{\mathtt{up}\ [N] \mid M \Downarrow \mathtt{up}\ N\} \cup \{\bot \mid M \Uparrow\}$ for $B = LA$.

Big Question

- Programs $\vdash M, M' : \texttt{bool}$ are behaviourally equivalent when $[M] = [M']$.
- We would like a denotational semantics such that for programs $\vdash M, M' : \texttt{bool}$, we have

  $[\![M]\!] = [\![M']\!]$ iff M and M' are behaviourally equivalent.

- Is this possible?

What doesn't work (1)

- A semantics is divergence-least when
  - terms denote element of a poset
  - all constructs are monotone
  - $\mathtt{diverge} \stackrel{\text{def}}{=} \mathtt{rec}\ \mathtt{x}.\ \mathtt{x}$ denotes least element $\perp$.
- This is the case if `rec` denotes least prefixed point.
- Example: domain semantics

What goes wrong with divergence-least

(folklore, also Lassen, Levy, Panangaden, APPSEM 2005)

- On the one hand

$$\mathtt{true\ or\ diverge} \leqslant \mathtt{true\ or\ true} = \mathtt{true}$$

- On the other hand, monotonicity of `amb` gives

$$\begin{aligned}
\mathtt{true} &= \mathtt{if}\ (\mathtt{false\ amb\ diverge})\ \mathtt{then\ diverge\ else\ true} \\
&\leqslant \mathtt{if}\ (\mathtt{false\ amb\ true})\ \mathtt{then\ diverge\ else\ true} \\
&= \mathtt{true\ or\ diverge}
\end{aligned}$$

- So $\mathtt{true\ or\ diverge} = \mathtt{true}$
- Each powerdomain theory either gives this equation, or makes `amb` non-monotone.

What doesn't work (2)

- Well-pointed semantics is one where a term in context Γ denotes a function from a set of environments.
- E.g. a term $\mathtt{x} : L1 \vdash M : L1$ should denote a function from $[L1]$ to itself.
- And $[\![\mathtt{rec\ x}.\ M]\!]$ should be a fixpoint of this function.
- We need some way of computing this fixpoint.

Operational Question 1 (Lassen 1998)

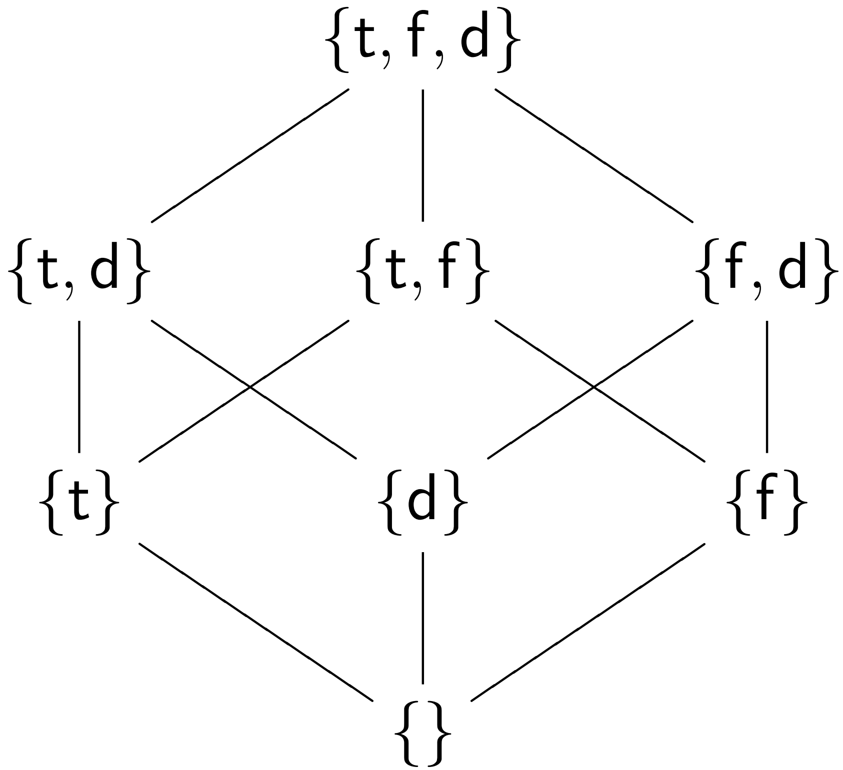
- Two closed terms $\vdash M, M : A$ are convex bisimilar when $[M] = [M']$.
- This is robust (preserved by every context).
- Two terms $\Gamma \vdash M, M' : A$ are convex applicatively bisimilar when $M[\overrightarrow{N/\mathtt{x}}]$ and $M'[\overrightarrow{N/\mathtt{x}}]$ are convex bisimilar for every Γ-environment $\overrightarrow{N/\mathtt{x}}$.
- Is this a congruence?
- Without `amb`, the answer is yes.

Operational Question 2 (Lassen 1999)

- Two terms $\Gamma \vdash M, M' : A$ are contextually equivalent when $\mathcal{C}M$ and $\mathcal{C}M'$ are behaviourally equivalent for every ground context $\mathcal{C}[\cdot]$.
- Two $\Gamma \vdash M, M' : A$ terms are Closed Instantiation equivalent when $M[\overrightarrow{N/\mathtt{x}}]$ and $M'[\overrightarrow{N/\mathtt{x}}]$ are contextually equivalent for every Γ-environment $\overrightarrow{N/\mathtt{x}}$.
- The context lemma says that CI equivalence implies contextual equivalence. This is true without `amb`.
- Is it true in the presence of `amb`?

Inclusion Ordering

Both of these questions has a variant where we use inclusion of behaviour sets rather than equality.



This makes `amb` monotone.

Divergence Ordering

Alternatively, we can use equality for convergence, but inclusion for divergence.

$$
\begin{tikzcd}
\{\mathsf{d}\} \arrow[d, dash] & \{\mathsf{t},\mathsf{d}\} \arrow[d, dash] & \{\mathsf{f},\mathsf{d}\} \arrow[d, dash] & \{\mathsf{t},\mathsf{f},\mathsf{d}\} \arrow[d, dash] \\
\{\} & \{\mathsf{t}\} & \{\mathsf{f}\} & \{\mathsf{t},\mathsf{f}\}
\end{tikzcd}
$$

This makes `amb` monotone.

Breaking well-pointedness

- We will exhibit two terms $\mathtt{x} : L1 \vdash M, M'' : L1$ and a context $\mathcal{C}[\cdot] : 1$
  such that
  - $\mathcal{C}[M] \not\Uparrow$ and $\mathcal{C}[M''] \Uparrow$
  - but M and M'' represent the same selfmap f on $[L1]$.
- This refutes all 6 operational conjectures, and shows the impossibility of a well-pointed denotational semantics.
- The terms $\mathtt{rec}\ \mathtt{x}.\ M$ and $\mathtt{rec}\ \mathtt{x}.\ M''$ represent different fixpoints of f.

The Terms

$$M \stackrel{\text{def}}{=} (\mathtt{up}\ \mathtt{top})\ \mathtt{amb}\ (\mathtt{pm}\ \mathtt{x}\ \mathtt{as}\ \mathtt{up}\ \mathtt{z}.\mathtt{up}\ (\mathtt{top}\ \mathtt{or}\ \mathtt{z}))$$

$$M' \stackrel{\text{def}}{=} \mathtt{up}\ (\mathtt{top}\ \mathtt{or}\ \mathtt{pm}\ (\mathtt{x}\ \mathtt{amb}\ \mathtt{up}\ \mathtt{top})\ \mathtt{as}\ \mathtt{up}\ \mathtt{y}.\mathtt{y})$$

$$M'' \stackrel{\text{def}}{=} M\ \mathtt{or}\ M'$$

Consider $M[N/\mathtt{x}]$ and $M''[N/\mathtt{x}]$.

- Neither may diverge.
- Both may return $\mathtt{up}\ P$, where $P \Downarrow \mathtt{top}$ and $P \not\Uparrow$
- Neither may return $\mathtt{up}\ P$, where $P \not\Downarrow \mathtt{top}$.
- If $N \Downarrow \mathtt{up}\ Q$, where $Q \Uparrow$, then both may return $\mathtt{up}\ P$, where $P \Downarrow \mathtt{top}$ and $P \Uparrow$.
- Otherwise, neither may.

The distinguishing context

$$\begin{aligned}
M &\stackrel{\text{def}}{=} (\mathtt{up\ top})\ \mathtt{amb}\ (\mathtt{pm\ x\ as\ up\ z.up}\ (\mathtt{top\ or\ z}))\\
M' &\stackrel{\text{def}}{=} \mathtt{up}\ (\mathtt{top\ or\ pm}\ (\mathtt{up\ top\ amb\ x})\ \mathtt{as\ up\ y.\ y})\\
M'' &\stackrel{\text{def}}{=} M\ \mathtt{or}\ M'\\
\mathcal{C}[\cdot] &\stackrel{\text{def}}{=} \mathtt{pm}\ (\mathtt{up\ top\ amb}\ (\mathtt{rec\ x.}[\cdot]))\ \mathtt{as\ up\ y.\ y}
\end{aligned}$$

$\mathcal{C}[M'']$ may diverge: just keep choosing to go right, using

$$\mathtt{rec\ x.}M'' \Downarrow \mathtt{up}\ (\mathtt{top\ or}\ \mathcal{C}[M''])$$

$\mathcal{C}[M]$ cannot diverge because if $\mathtt{rec\ x.}M \Downarrow \mathtt{up}\ N$ then $N = (\mathtt{top\ or}\ )^n\ \mathtt{top}$, which cannot diverge.

That Big Question

- We would like a denotational semantics such that for programs $\vdash M, M' : \texttt{bool}$, we have

$$[\![M]\!] = [\![M']\!] \text{ iff } [M] = [M']$$

- Is this possible?

Still open.

General Amb vs Ground Amb

- Our example uses `amb` at type $L1$, not just at ground type.
- All 6 conjectures are true if we restrict to ground `amb`.
- The proofs are mild adaptations of the proofs without `amb`.
- These results can be extended to a full type system with recursive types.
- It can be call-by-name, call-by-value or call-by-push-value.

Cf. O'Hearn's monad for ground storage, Laird's semantics of ground control.

Uses

- A use is a special kind of ground context.
  - A use for $\texttt{bool}$ is $\texttt{if}\ [\cdot]\ \texttt{then}\ N\ \texttt{else}\ N'$
  - A use for 1 is $[\cdot]; N$
  - A use for LA is $\texttt{pm}\ [\cdot]\ \texttt{as up x.}\ N$.
- Closed terms $M, M' : A$ are Uses equivalent when they are behaviourally equivalent under every use.
- The Uses theorem says that Uses equivalence implies contextual equivalence.
- Again 2 variants using inclusion.
- Context lemma + Uses theorem = CIU theorem

Amb breaks Uses

We define two terms $\vdash M, M' : L1$ and a context $\mathcal{C}[\cdot] : 1$ such that

- M and M' are Uses equivalent
- $\mathcal{C}[M] \not\Uparrow$ but $\mathcal{C}[M'] \Uparrow$

$$
\begin{aligned}
M &\stackrel{\text{def}}{=} \texttt{diverge or up top} \\
M' &\stackrel{\text{def}}{=} M \;\texttt{or up}\;(\texttt{top or diverge}) \\
\mathcal{C}[\cdot] &\stackrel{\text{def}}{=} \texttt{pm}\;([\cdot]\;\texttt{amb up top})\;\texttt{as x. x}
\end{aligned}
$$

With ground amb, the CIU theorem holds.

Conclusion (denotational slant)

- `amb` cannot have a well-pointed denotational semantics.
- ground `amb` might have.