Chasing Bottoms

A Case Study in Program Verification in the Presence of Partial and Infinite Values

Nils Anders Danielsson   Patrik Jansson

Chalmers

Context

- Cover project.
- Verification of real Haskell programs.
- Haskell non-strict $\Rightarrow$ partial and infinite values relatively common.
  1. Lazily generate infinite structure.
  2. Use selected parts.

Case study

$$\mathbf{data}\ T = L \mid B\ T\ T$$

$$\begin{aligned}
&\mathit{pretty}' :: T \to \mathit{String}\\
&\mathit{pretty}'\ L \qquad\quad = \texttt{"L"}\\
&\mathit{pretty}'\ (B\ l\ r) = \texttt{"B"} +\!\!+\ \mathit{pretty}'\ l +\!\!+\ \mathit{pretty}'\ r
\end{aligned}$$

$$\begin{aligned}
&\mathit{pretty} :: (T, \mathit{String}) \to \mathit{String}\\
&\mathit{pretty}\ (t, \mathit{cs}) = \mathit{pretty}'\ t +\!\!+\ \mathit{cs}
\end{aligned}$$

$$\mathit{pretty}\ (B\ L\ (B\ L\ L), \texttt{"x"}) = \texttt{"BLBLLx"}$$

Case study

$$\begin{aligned}
&\mathit{parse} :: \mathit{String} \to (T, \mathit{String})\\
&\mathit{parse}\ (\texttt{'L'} : \mathit{cs}) = (L, \mathit{cs})\\
&\mathit{parse}\ (\texttt{'B'} : \mathit{cs}) = (B\ l\ r, \mathit{cs}'')\\
&\quad \mathbf{where}\ (l, \mathit{cs}')\ \ = \mathit{parse}\ \mathit{cs}\\
&\qquad\qquad\quad (r, \mathit{cs}'') = \mathit{parse}\ \mathit{cs}'
\end{aligned}$$

Properties

For total, finite input:

- $parse \circ pretty = id :: (T, String) \to (T, String)$
- $pretty \circ parse \sqsubseteq id :: String \to String$

(Embedding-projection pair.)

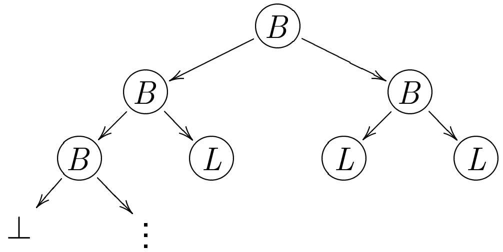
Properties

For arbitrary input:

- $parse \circ pretty = strictify \sqsubseteq id :: (T, String) \to (T, String)$
- $pretty \circ parse \sqsubseteq id :: String \to String$

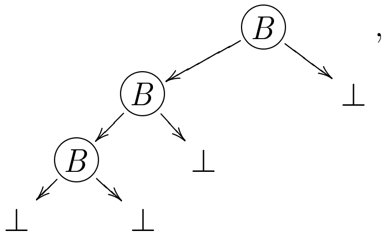
Properties

- $p = ($



$, \texttt{"x"})$

- $strictify\ p = ($



$, \perp)$

Properties

$$
\begin{aligned}
&strictify :: (T, a) \to (T, a) \\
&strictify\ (t, a) = t\ \text{`}seq\text{`}\ (t', tTotal\ \text{`}seq\text{`}\ a) \\
&\quad \mathbf{where}\ (t', tTotal) = strictify'\ t
\end{aligned}
$$

$$
\begin{aligned}
&strictify' :: T \to (T, ()) \\
&strictify'\ L \qquad\quad = (L, ()) \\
&strictify'\ (B\ l\ r) = (B\ l'\ (lTotal\ \text{`}seq\text{`}\ r'), lTotal\ \text{`}seq\text{`}\ rTotal) \\
&\quad \mathbf{where}\ (l', lTotal) \ = strictify'\ l \\
&\qquad\qquad\ \ (r', rTotal) = strictify'\ r
\end{aligned}
$$

seq

$seq :: a \to b \to b$

$seq\ \bot\ b = \bot$

$seq\ a\ \ b = b$ -- if $a \neq \bot$

Caveats

- η-equality is not valid.
  - $\bot \neq \lambda x.\bot$
- No surjective pairing.
  - $(fst\ \bot, snd\ \bot) = (\bot, \bot) \neq \bot$

Caveats

- Typical $Monad$ instances not monads.
  - $\bot \gg\!= return = \lambda x.\bot \neq \bot$
- Surprising (?) pattern matching semantics.
  - $\lambda True\ x.x \neq \lambda True.\lambda x.x$
  - $\mathbf{let}\ (x, y) = p\ \mathbf{in}\ f\ (x, y)$ equals $f\ p$
    iff $p \neq \bot$ or $f\ (\bot, \bot) = f\ \bot$

Testing partial values

- $isBottom :: a \to Bool$
  1. Force evaluation of argument.
  2. Catch any exceptions (of the right kind).
- Several flavours of bottom:
  - Non-termination.
  - $error$ "..."
  - Pattern match failure.

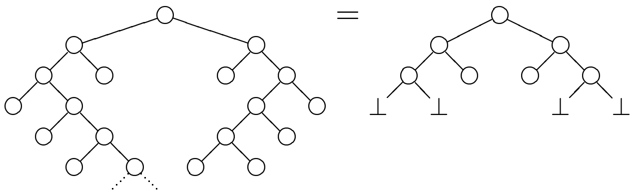
  Indistinguishable within pure subset of Haskell.
- $(isBottom :: a \to IO\ Bool)$

Testing partial values

- $\hat{=}, \hat{\sqsubseteq} :: Data\ a \Rightarrow a \rightarrow a \rightarrow Bool$
- Nice together with QuickCheck.

Testing infinite values

- $approx :: Nat \rightarrow T \rightarrow T$
- $approx\ 3$



- The approximation lemma:
  - $t_1 = t_2$ iff $\forall n \in \mathbb{N}.\ approx\ n\ t_1 = approx\ n\ t_2$

Testing partial and infinite values

- $pretty \circ parse \sqsubseteq id$

  $$\begin{aligned} &forAll\ string\ (\lambda cs. \\ &\quad approx\ n\ ((pretty \circ parse)\ cs)\ \hat{\sqsubseteq}\ approx\ n\ (id\ cs)) \end{aligned}$$

- $parse \circ pretty = strictify$

  $$\begin{aligned} &forAll\ pair\ (\lambda p. \\ &\quad approxPair\ n\ ((parse \circ pretty)\ p)\ \hat{=} \\ &\quad approxPair\ n\ (strictify\ p)) \\ &\quad \textbf{where}\ approxPair\ n\ (t, cs) = (approx\ n\ t, approx\ (2\hat{}n)\ cs) \end{aligned}$$

Proofs

- Fixpoint induction.
- The approximation lemma.
- Coinduction.
- Fusion.

Proofs

To prove

$$pretty \circ parse \sqsubseteq id$$

the following inductive hypothesis was used:

$$\forall m \in \mathbb{N}.\ pp_m \sqsubseteq id.$$

Here

$$\begin{aligned} pp_m\ cs = pretty'\ t_1 +\!\!+\ pretty'\ t_2 +\!\!+ \ldots +\!\!+\ pretty'\ t_m +\!\!+\ cs_m \\ \textbf{where}\ (t_1,\ cs_1) &= parse\ cs \\ (t_2,\ cs_2) &= parse\ cs_1 \\ &\vdots \\ (t_m, cs_m) &= parse\ cs_{m-1} \end{aligned}$$

Conclusions

- Testing reasonably easy, but test case coverage has to be checked.
- Proving tricky.
- Approximate semantics might be nice.
  - $\lambda x.\bot \rightsquigarrow \bot$
- Library available from `http://www.cs.chalmers.se/~nad/`.