# Partiality, Revisited

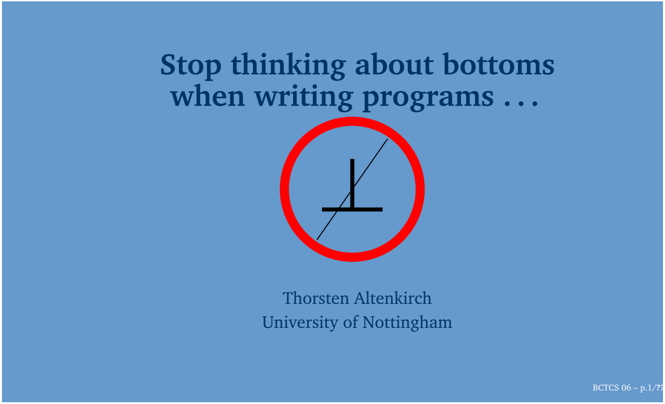
Thorsten Altenkirch

Functional Programming Laboratory
School of Computer Science
University of Nottingham

jww Paolo Capriotti, Nils Anders Danielsson, Nicolai Kraus

May 26, 2016

# BCTCS 2006


Stop thinking about bottoms
when writing programs . . .
Thorsten Altenkirch
University of Nottingham
BCTCS 06 – p.1/??

# Partiality is an effect

- $A_\perp$ — partial computations over $A$.
- Computational monad
- $\eta : A \to A_\perp$ — embed values into partial computations.
- $\perp : A_\perp$ — non-terminating computation.
- $A \neq A + 1$ !
- Given $f : (A \to B_\perp) \to (A \to B_\perp)$
  compute $\mathit{fix}(f) : A \to B_\perp$
  satisfying $\mathit{fix}(f) = f(\mathit{fix}(f))$.
- We need that $f$ is continuous.

## Capretta's solution

- Defining the Delay monad coinductively:

$$Delay : \mathbf{Set} \to \mathbf{Set}$$

$$\eta : A \to Delay(A)$$
$$later : \infty Delay(A) \to Delay(A)$$

- Divergent computation:

$$\bot = later(\bot)$$

- Want to identify computations that differ in a finite number of *later*.

**Paper**

Venanzio Capretta
*General Recursion via Coinductive Types*
Logical Methods in Computer Science, 2005

## Weak bisimilarity

- Inductively define

$$\downarrow : A_\perp \to A \to \textbf{Prop}$$

$$\eta(a) \downarrow a$$
$$p \downarrow a \to later(p) \downarrow a$$

- Equivalence relation:

$$\approx : Delay(A) \to Delay(A) \to \textbf{Prop}$$

$$p \approx q := \Pi_{a:A}(p \downarrow a \leftrightarrow q \downarrow a)$$

# Quotient types

- Capretta used setoids = a type + an equivalence relation.
- In 2005 we (A.,Capretta,Uustalu) suggested to use quotient types

$$A_\perp :\equiv \mathit{Delay}(A) / \approx$$

- We never published a paper about this ...

# Desired properties

- $(-)_\bot$ should be a monad.

$$\gg\!= : A_\bot \to (A \to B_\bot) \to B_\bot$$

- $A_\bot$ should be a $\omega$-CPO.

$$\bigsqcup : \Pi_{f:\mathbb{N}\to A_\bot}(\Pi_{n:\mathbb{N}} f(n) \sqsubseteq f(n+1)) \to A_\bot$$

# TYPES 2015

**Paper based on TYPES talk**

James Chapman, Tarmo Uustalu and Niccolò
Quotienting the Delay Monad by Weak Bisimilarity
Theoretical Aspects of Computing – ICTAC 2015

- Using countable axiom of choice $\mathrm{AC}_\omega$:

$$(\Pi x : \mathbb{N}.\exists y : B.R(x,y) \to (\exists f : \mathbb{N} \to B.\Pi_{x:\mathbb{N}} R(x, f(x)))$$

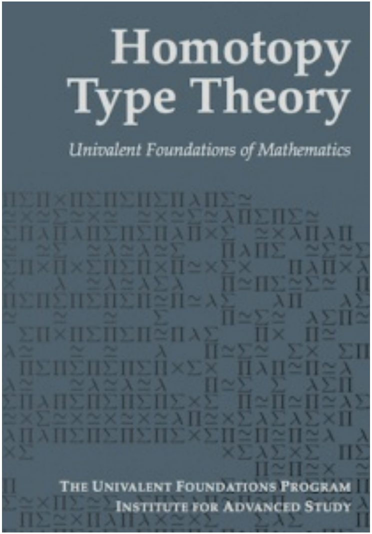
  they show that $(-)_\perp$ is a monad.
- $\exists x : A.\Phi(x) :\equiv ||\Sigma x : A.\Phi(x)||$
- $\mathrm{AC}_\omega$ is not provable in Type Theory.
- But it can be constructively justified.
  (unlike general AC)

# Dejavue ?

- Similar problem with the Cauchy Reals.

$$S :\equiv \Sigma f : \mathbb{N} \to \mathbb{Q}.\Pi\epsilon : \mathbb{Q}.\epsilon > 0 \to \exists n : \mathbb{N}.|f(n+1) - f(n)| < \epsilon$$
$$(f, -) \sim (g, -) :\equiv \Pi\epsilon : \mathbb{Q}.\epsilon > 0 \to \exists n.n \in \mathbb{N}.|f(n) - g(n)| < \epsilon$$
$$\mathbb{R} :\equiv S/\sim$$

- Cannot prove in Type Theory that $\mathbb{R}$ is Cauchy complete.
  Every convergent sequence of reals has a limit.
- Unless we assume countable choice.


Homotopy
Type Theory
Univalent Foundations of Mathematics
The Univalent Foundations Program
Institute for Advanced Study

# HITs to the rescue

- Using (set-truncated) higher inductive types we can avoid $AC_\omega$.
- We define $\mathbb{R}$ as:

  $\eta : \mathbb{Q} \to \mathbb{R}$

  Every convergent sequence in $\mathbb{R} \to \mathbb{R}$

- We define
  - the elements,
  - the order relation,
  - and equality

  at the same time.
- We call this a
  *Quotient Inductive Type*
  since it isn't higher dimensional in the sense of HoTT.

# Defining $A_\bot$ as a QIT

$$A_\bot : \mathbf{Set}$$
$$\sqsubseteq : A_\bot \to A_\bot \to \mathbf{Prop}$$

$$\bot : A_\bot$$
$$\eta : A \to A_\bot$$
$$\bigsqcup : \Pi_{f:\mathbb{N}\to A_\bot}(\Pi_{n:\mathbb{N}} f(n) \sqsubseteq f(n+1)) \to A_\bot$$

$$\frac{}{d \sqsubseteq d} \qquad \frac{}{\bot \sqsubseteq d} \qquad \frac{\bigsqcup(f,p) \sqsubseteq d}{\Pi_{n:\mathbb{N}} f(n) \sqsubseteq d} \qquad \frac{\Pi_{n:\mathbb{N}} f(n) \sqsubseteq d}{\bigsqcup(f,p) \sqsubseteq d}$$

$$\frac{d \sqsubseteq d' \qquad d' \sqsubseteq d}{d = d'}$$

# Results

- $(-)_\perp$ is a monad.
  *formalized in Agda*
- $A_\perp$ is non-trivial.
  $\perp \neq \eta(a)$
- $A_\perp$ is an $\omega$-CPO
  trivial
  Indeed we define $A_\perp$ as the free $\omega$-CPO over $A$.
- Assuming $AC_\omega$ the definition is equivalent to the previous one.
- Case study:
  Danielsson has ported the Agda code related to his paper
  *Operational Semantics using the Partiality Monad*
  to the new definition.