Many More Predecessors: A Representation Workout

Oleg Kiselyov
http://okmij.org/ftp/Computation/lambda-calc.html#predecessors

Tohoku University, Japan

J.Functional Progr., March 2020
LFCS seminar, 28 May 2021
WG2.1 meeting, 15 Dec 2021

Outline

Subject

Pure untyped lambda-calculus and Church numerals

Subject

Pure untyped lambda-calculus and Church numerals

$$
\begin{aligned}
\mathsf{c}_0 &:= \quad \lambda f.\lambda x.x \\
\mathsf{c}_1 &:= \quad \lambda f.\lambda x.f\ x \\
\mathsf{c}_2 &:= \quad \lambda f.\lambda x.f\ (f\ x) \\
&\quad \ldots \\
\mathsf{c}_n &:= \quad \lambda f.\lambda x.f^{(n)}x
\end{aligned}
$$

Successor	$\mathsf{succ}\ \mathsf{c}_n \rightsquigarrow^* \mathsf{c}_{(n+1)}$
	$\mathsf{succ} := \lambda n.\lambda f x.f\ (n\ f\ x)$
Predecessor	$\mathsf{pred}\ \mathsf{c}_{(n+1)} \rightsquigarrow^* \mathsf{c}_n$

Motivation

Why would anyone care?

Motivation

Why would anyone care?

- It's a good puzzle: even Church was stumped

Motivation

Why would anyone care?

- It's a good puzzle: even Church was stumped
- Colorful story: tooth fairies, altered states of mind, ...

Motivation

Why would anyone care?

- It's a good puzzle: even Church was stumped
- Colorful story: tooth fairies, altered states of mind, ...
- The commonly explained Kleene predecessor is called "very tricky" and takes a while to explain

Motivation

Why would anyone care?

- It's a good puzzle: even Church was stumped
- Colorful story: tooth fairies, altered states of mind, ...
- The commonly explained Kleene predecessor is called "very tricky" and takes a while to explain
- Kleene predecessor is one of the worst in every metric: hard to explain, hard to fit on one line, not efficient, ...

Motivation

Why would anyone care?

- It's a good puzzle: even Church was stumped
- Colorful story: tooth fairies, altered states of mind, ...
- The commonly explained Kleene predecessor is called "very tricky" and takes a while to explain
- Kleene predecessor is one of the worst in every metric: hard to explain, hard to fit on one line, not efficient, ...
- Searching for predecessors turns out very insightful

Motivation

Why would anyone care?

- It's a good puzzle: even Church was stumped
- Colorful story: tooth fairies, altered states of mind, ...
- The commonly explained Kleene predecessor is called "very tricky" and takes a while to explain
- Kleene predecessor is one of the worst in every metric: hard to explain, hard to fit on one line, not efficient, ...
- Searching for predecessors turns out very insightful

Why did I care?

Why did I care, initially

Pure untyped lambda-calculus and Church numerals

$$
\begin{aligned}
\mathsf{c}_0 &:= \quad \lambda f.\lambda x.x \\
\mathsf{c}_1 &:= \quad \lambda f.\lambda x.f\ x \\
\mathsf{c}_2 &:= \quad \lambda f.\lambda x.f\ (f\ x) \\
&\quad \ldots \\
\mathsf{c}_n &:= \quad \lambda f.\lambda x.f^{(n)}x
\end{aligned}
$$

Successor $\quad \mathsf{succ}\ \mathsf{c}_n \rightsquigarrow^* \mathsf{c}_{(n+1)}$

$\mathsf{succ} := \lambda n.\lambda f x.f\ (n\ f\ x)$

Predecessor $\quad \mathsf{pred}\ \mathsf{c}_{(n+1)} \rightsquigarrow^* \mathsf{c}_n$

$\mathsf{pred} := \lambda n.n\ (\lambda p.p\ (\lambda x.x)(\lambda f x.f\ (p\ f\ x)))\ (\lambda f x s z.z)$

(two days + tooth)

Summary

It is a representation-change problem

Results

- Six (+ 2) predecessors in plain sight
  All except two are new
- All have (distinct) normal forms

Methods

- Two general methods (extend beyond numbers: trees, . . .)
- Several specific methods: one is particularly elegant
- Un-application

Outline

Koan: The Fundamental Tautology

$$\mathsf{c}_n \doteq \mathsf{c}_n\ \mathsf{succ}\ \mathsf{c}_0 \qquad (*)$$

Number representations

Equality

$$\mathsf{c}_n \doteq \mathsf{c}_n\ \mathsf{succ}\ \mathsf{c}_0 \qquad (*)$$

Number representations

Definition; p_0 and supp are parameters

$$\mathsf{p}_n \;:\dot{=}\; \mathsf{c}_n\;\mathsf{supp}\;\mathsf{p}_0 \qquad n > 0 \qquad (**)$$

Number representations

Definition; p_0 and supp are parameters

$$\mathsf{p}_n \;:\doteq\; \mathsf{c}_n\;\mathsf{supp}\;\mathsf{p}_0 \qquad n > 0 \qquad (**)$$

In the sequence p_0, p_1, p_2,...

- p_0 is the initial element
- supp is the step: $\mathsf{supp}\;\mathsf{p}_n \rightsquigarrow^* \mathsf{p}_{(n+1)}$

Conversely, given p_0 and supp as parameters, (**) is the closed-form for the n-th element

General Recipe: Representation Change

By (**), transform c_n to a different number form p_n

If p_n are such that

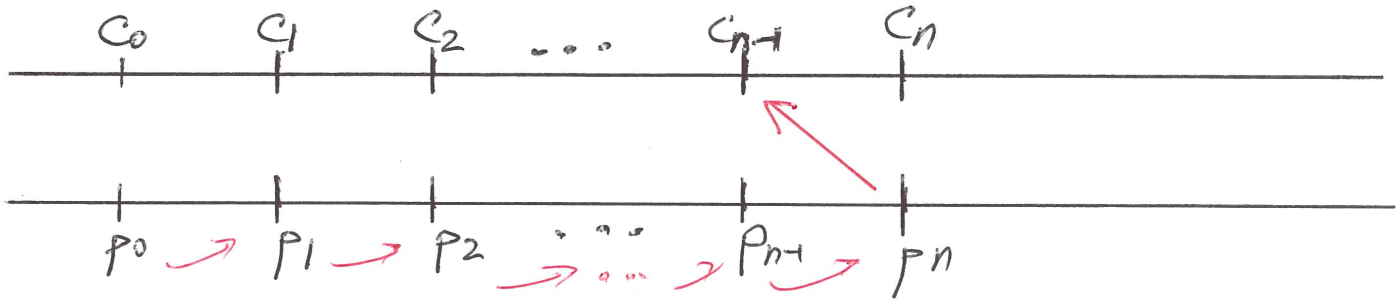
- $\mathsf{predp} : \mathsf{p}_{(n+1)} \mapsto \mathsf{p}_n$ and $\mathsf{proj} : \mathsf{p}_n \mapsto \mathsf{c}_n$ are easy to define, or
- $\mathsf{projpred} : \mathsf{p}_{(n+1)} \mapsto \mathsf{c}_n$ is easy to define

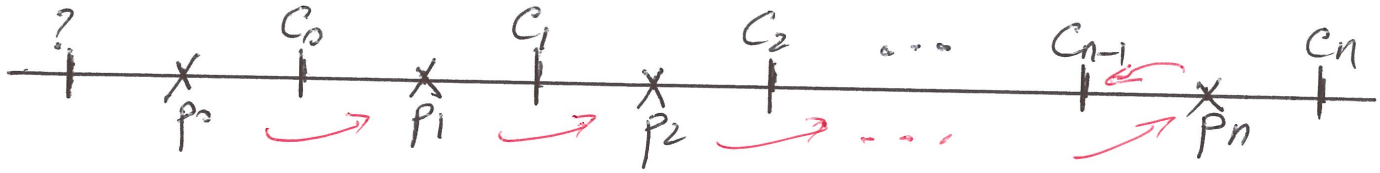
Then the predecessor is

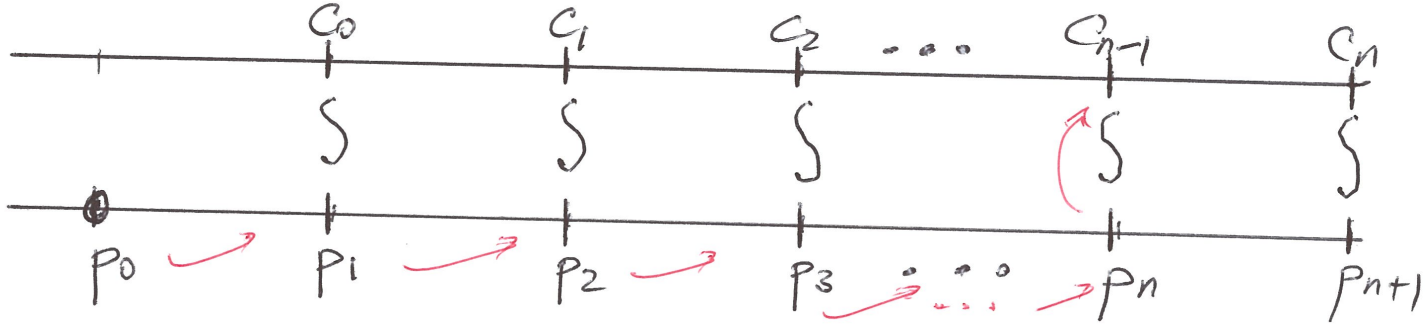
$$\mathsf{pred} :\doteq \lambda n.\mathsf{projpred}\ \ (n\ \mathsf{supp}\ \mathsf{p}_0)$$

Now to find the fitting supp and p_0!

Diagram


c_0 c_1 c_2 ... c_{n-1} c_n
p_0 p_1 p_2 ... p_{n-1} p_n


?
c_0 c_1 c_2 ... c_{n-1} c_n
p_0 p_1 p_2 ... p_n


c_0 c_1 c_2 ... c_{n-1} c_n
p_0 p_1 p_2 p_3 ... p_n p_{n+1}

Kleene Predecessor: midway numbers

Take p_n a point between two consecutive numbers c_n and c_{n-1}, represented as as pair $(\mathsf{c}_{n-1}\ ,\mathsf{c}_n)$:

$$\mathsf{p}_0 := (\mathsf{c}_{-1}\ ,\mathsf{c}_0) \qquad \mathsf{p}_1 := (\mathsf{c}_0\ ,\mathsf{c}_1) \qquad \mathsf{p}_2 := (\mathsf{c}_1\ ,\mathsf{c}_2) \qquad \ldots$$

The successor on p_n and $\mathsf{projpred}$

$$\begin{aligned} \mathsf{supp} &:\doteq \lambda p.(\mathsf{snd}\ p, \mathsf{succ}\ (\mathsf{snd}\ p)) \\ \mathsf{projpred} &:\doteq \qquad\qquad\qquad \mathsf{fst} \end{aligned}$$

Plugging into

$$\mathsf{pred} :\doteq \lambda n.\mathsf{projpred}\ \ (n\ \mathsf{supp}\ \mathsf{p}_0)$$

and normalizing gives the Kleene predecessor:

$$\lambda n.n\ (\lambda ps.s\ (p(\lambda xy.y))\ (\lambda fx.f\ (p(\lambda xy.y)f\ x)))\ (\lambda p.p\ (\lambda fx.x)\ (\lambda fx.x)$$

Even better General Recipe

Then the predecessor is

$$\mathsf{pred} :\doteq \lambda n.\, \mathsf{projpred}\ \ (n\ \mathsf{supp}\ \mathsf{p}_0)$$

where $\mathsf{projpred} : \mathsf{p}_{(n+1)} \mapsto \mathsf{c}_n$ and is easy to define

The simplest is to make $\mathsf{projpred}$ (a sort of) an identity!

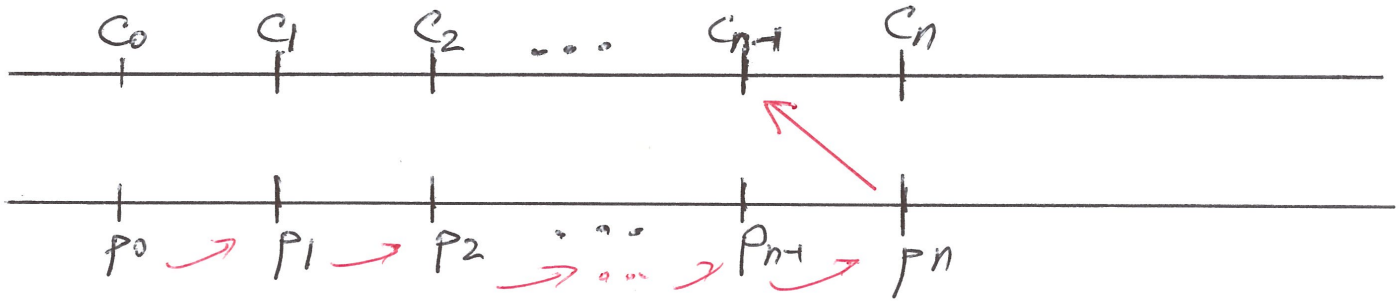
$$\mathsf{p}_{(n+1)} \sim \mathsf{c}_n$$

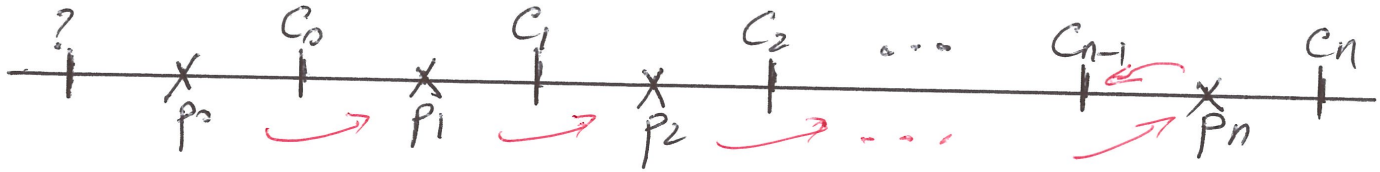
Now to somehow find p_0, and supp to go with it...

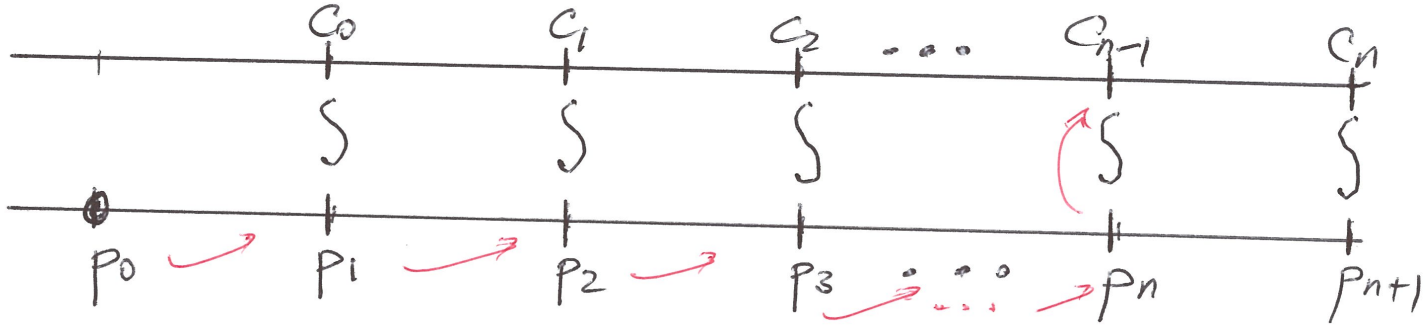
Plan for the next

- $\sim$ is a bijection: general methods
- $\sim$ is the identity: specific methods

Diagram


c_0 c_1 c_2 ... c_{n-1} c_n
p_0 p_1 p_2 ... p_{n-1} p_n


?
c_0 c_1 c_2 ... c_{n-1} c_n
p_0 p_1 p_2 ... p_n


c_0 c_1 c_2 ... c_{n-1} c_n
p_0 p_1 p_2 p_3 ... p_n p_{n+1}

Outline

Shifted-by-one numbers

$$\mathtt{p}_{(n+1)} \sim \mathtt{c}_n$$

Thus $\mathtt{p}_{(n+1)}$ are isomorphic to $\mathtt{c}_n$, plus one extra element, $\mathtt{p}_0$

Shifted-by-one numbers

$$\mathtt{p}_{(n+1)} \sim \mathtt{c}_n$$

Thus $\mathtt{p}_{(n+1)}$ are isomorphic to $\mathtt{c}_n$, plus one extra element, $\mathtt{p}_0$

read_int_opt : unit → int option

Shifted-by-one numbers

$$\mathsf{p}_{(n+1)} \sim \mathsf{c}_n$$

Thus $\mathsf{p}_{(n+1)}$ are isomorphic to c_n, plus one extra element, p_0

read_int_opt : unit → int option

Adding an extra element to a set: X option

$\mathsf{p}_0 := \mathsf{None}$ $\quad$ $\mathsf{p}_1 := \mathsf{Some}\,\mathsf{c}_0$ $\quad$ $\mathsf{p}_2 := \mathsf{Some}\,\mathsf{c}_1$ $\quad$ $\ldots$

Shifted-by-one numbers

$$\mathsf{p}_{(n+1)} \sim \mathsf{c}_n$$

Thus $\mathsf{p}_{(n+1)}$ are isomorphic to c_n, plus one extra element, p_0

$\mathsf{read\_int\_opt} : \mathsf{unit} \to \mathsf{int\ option}$

Adding an extra element to a set: X option

$\mathsf{p}_0 := \mathsf{None} \qquad \mathsf{p}_1 := \mathsf{Some}\,\mathsf{c}_0 \qquad \mathsf{p}_2 := \mathsf{Some}\,\mathsf{c}_1 \qquad \ldots$

$$\mathsf{supp} :\doteq \lambda p. \begin{cases} \mathsf{Some}\ \mathsf{c}_0 & \text{if } p = \mathsf{None} \\ \mathsf{Some}\ (\mathsf{succ}\ c) & \text{if } p = \mathsf{Some}\ \ c \end{cases}$$

Shifted-by-one numbers

$$\mathsf{p}_{(n+1)} \sim \mathsf{c}_n$$

Thus $\mathsf{p}_{(n+1)}$ are isomorphic to c_n, plus one extra element, p_0

read_int_opt : unit → int option

Adding an extra element to a set: X option

$\mathsf{p}_0 := \mathsf{None}$ $\quad \mathsf{p}_1 := \mathsf{Some}\,\mathsf{c}_0$ $\quad \mathsf{p}_2 := \mathsf{Some}\,\mathsf{c}_1$ $\quad \ldots$

$$\mathsf{supp} :\doteq \lambda p. \begin{cases} \mathsf{Some}\ \mathsf{c}_0 & \text{if } p = \mathsf{None} \\ \mathsf{Some}\ (\mathsf{succ}\ c) & \text{if } p = \mathsf{Some}\ \ c \end{cases}$$

And the predecessor is

$$\mathsf{pred} :\doteq \lambda n.\mathsf{fromSome}\ \ (n\ \mathsf{supp}\ \mathsf{None})$$

Shifted-by-one numbers

$$\mathsf{pred} \doteq \lambda n.\mathsf{fromSome}\ \ (n\ \mathsf{supp}\ \mathsf{None})$$

$$\mathsf{None} := \lambda k.\lambda y.y \qquad\qquad \mathsf{Some} := \lambda x.\lambda k.\lambda y.kx$$

$$\mathsf{supp} \doteq \lambda p.\mathsf{Some}\ (p\ \mathsf{succ}\ \mathsf{c}_0)$$

The predecessor is

$$\lambda n.(n\ \mathsf{supp}\ \mathsf{p}_0)\ \mathsf{id}\ \mathsf{c}_0$$

or, in the desugared, normal form

$$\lambda n.n\ (\lambda pky.k\ (p(\lambda nfx.f\ \ (n\ \ f\ \ x))(\lambda fx.x)))\ (\lambda ky.y)\ (\lambda x.x)\ (\lambda fx.x)$$

Optimized Koan

$$\mathsf{c}_n \doteq \mathsf{c}_n\ \mathsf{succ}\ \mathsf{c}_0 \qquad (*)$$

Optimized Koan

$$\mathsf{c}_n \doteq \lambda f z.\mathsf{c}_n\ \mathsf{succ}_{fz}\ \mathsf{c}_{0fz}$$

Optimized Koan

$$\mathsf{c}_n \doteq \lambda f\, z.\mathsf{c}_n\ f\ \ z$$

Shifted-by-one numbers

$$\mathsf{pred} \doteq \lambda n.\mathsf{fromSome}\ \ (n\ \mathsf{supp}\ \mathsf{None})$$

$$\mathsf{None} := \lambda k.\lambda y.y \qquad\qquad \mathsf{Some} := \lambda x.\lambda k.\lambda y.kx$$

$$\mathsf{supp} \doteq \lambda p.\mathsf{Some}\ (p\ \mathsf{succ}\ \mathsf{c}_0)$$

The predecessor is

$$\lambda n.(n\ \mathsf{supp}\ \mathsf{p}_0)\ \mathsf{id}\ \mathsf{c}_0$$

or, in the desugared, normal form

$$\lambda n.n\ (\lambda pky.k\ (p(\lambda nfx.f\ (n\ f\ x))(\lambda fx.x)))\ (\lambda ky.y)\ (\lambda x.x)\ (\lambda fx.x)$$

Shifted-by-one numbers, optimized

$$\mathsf{pred} \doteq \lambda n.\lambda f z.\mathsf{fromSome}_{fz}\ \ (n\ \mathsf{supp}_{fz}\ \mathsf{None}_{fz})$$

$$\mathsf{None}_{fz} := \lambda k.z \qquad\qquad \mathsf{Some}_{fz} := \lambda x.\lambda k.kx$$

$$\mathsf{supp}_{fz} \doteq \lambda p.\mathsf{Some}_{fz}\ (p\ f)$$

The predecessor is

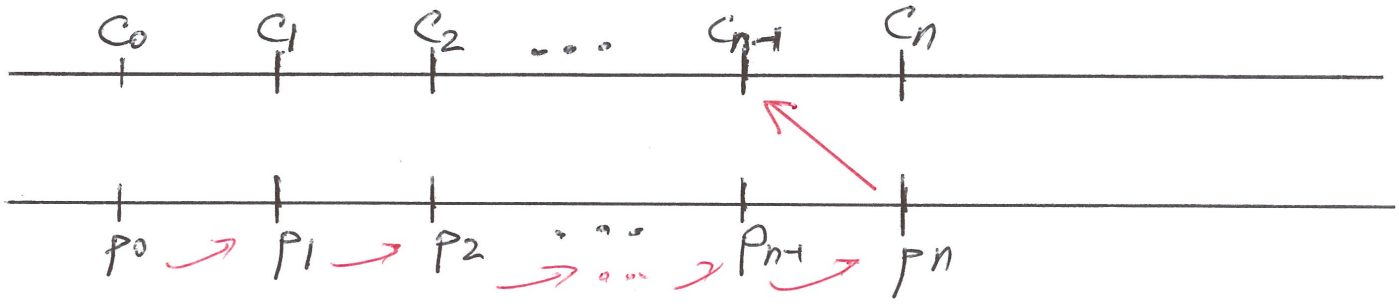
$$\lambda n.\lambda f z.(n\ \mathsf{supp}_{fz}\ (\lambda k.z))\ \mathsf{id}$$

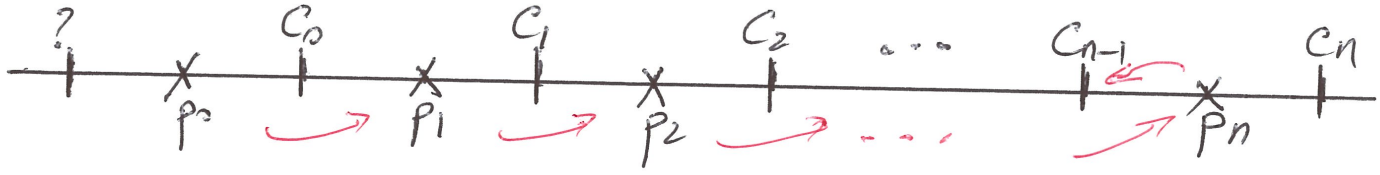
or, in the desugared, normal form

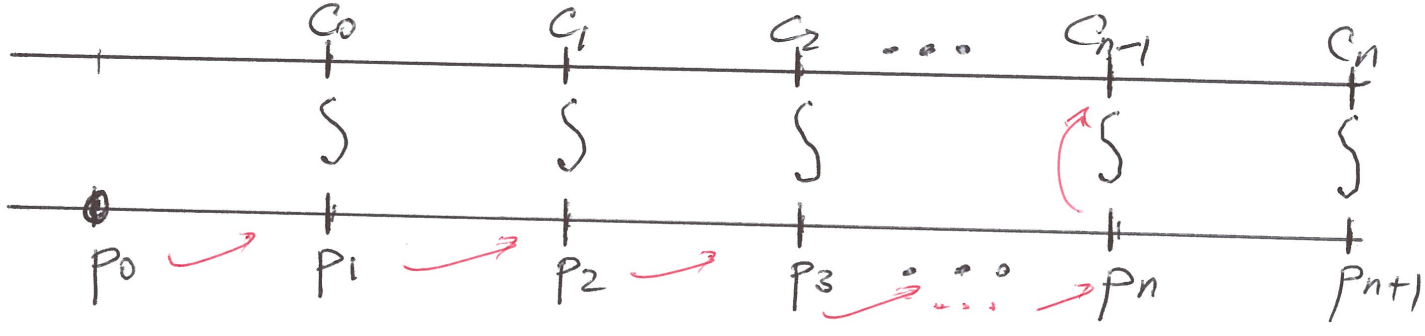
$$\lambda n f z.n\ (\lambda p k.k\ (p\ f))\ (\lambda k.z)\ (\lambda y.y)$$

Outline

Diagram


C0
C1
C2
Cn-1
Cn
P0
P1
P2
Pn-1
Pn


?
C0
C1
C2
Cn-1
Cn
P0
P1
P2
Pn


C0
C1
C2
Cn-1
Cn
P0
P1
P2
P3
Pn
Pn+1

Searching for -1

$$\mathsf{p}_{(n+1)} \equiv \mathsf{c}_n$$

Thus $\mathsf{p}_{(n+1)}$ are c_n themselves. What is p_0, a.k.a c_{-1}? The predecessor is

$$\mathsf{pred} :\doteq \lambda n.(n\ \mathsf{succ}'\ \mathsf{c}_{-1}) \qquad \text{where}$$

$$\mathsf{succ}' :\doteq \lambda n.\begin{cases} \mathsf{succ}\ n & \text{if } n \text{ is a Church numeral} \\ \mathsf{c}_0 & \text{if } n \text{ is } \mathsf{c}_{-1} \end{cases}$$

Now to find c_{-1} than can be distinguished from c_n

Searching for -1

$$\mathsf{p}_{(n+1)} \equiv \mathsf{c}_n$$

Thus $\mathsf{p}_{(n+1)}$ are c_n themselves. What is p_0, a.k.a c_{-1}? The predecessor is

$$\mathsf{pred} :\doteq \lambda n.(n\ \mathsf{succ}'\ \mathsf{c}_{-1}) \qquad \text{where}$$

$$\mathsf{succ}' :\doteq \lambda n.\begin{cases} \mathsf{succ}\ n & \text{if } n \text{ is a Church numeral} \\ \mathsf{c}_0 & \text{if } n \text{ is } \mathsf{c}_{-1} \end{cases}$$

Now to find c_{-1} than can be distinguished from c_n

$$\mathsf{c}_n\ \mathsf{id} \rightsquigarrow^* \mathsf{id}$$

Searching for -1

$$\mathsf{p}_{(n+1)} \equiv \mathsf{c}_n$$

Thus $\mathsf{p}_{(n+1)}$ are c_n themselves. What is p_0, a.k.a c_{-1}? The predecessor is

$$\mathsf{pred} :\doteq \lambda n.(n\ \mathsf{succ}'\ \mathsf{c}_{-1}) \qquad \text{where}$$

$$\mathsf{succ}' :\doteq \lambda n.\begin{cases} \mathsf{succ}\ n & \text{if } n \text{ is a Church numeral} \\ \mathsf{c}_0 & \text{if } n \text{ is } \mathsf{c}_{-1} \end{cases}$$

Now to find c_{-1} than can be distinguished from c_n

$$\mathsf{c}_n\ \mathsf{id} \rightsquigarrow^* \mathsf{id}$$

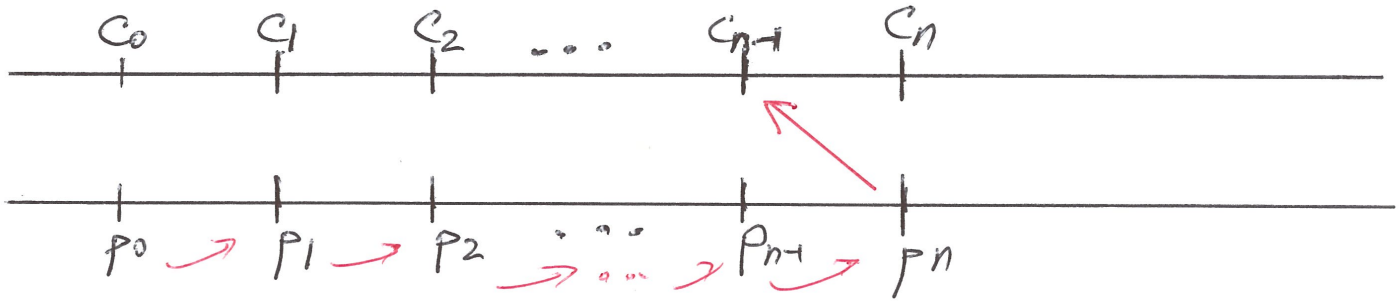
$$\mathsf{c}_{-1} := \lambda f x.\mathsf{c}_0 \qquad\qquad \mathsf{succ}' :\doteq \lambda n.n\ \ \mathsf{id}\ (\mathsf{succ}\ n)$$

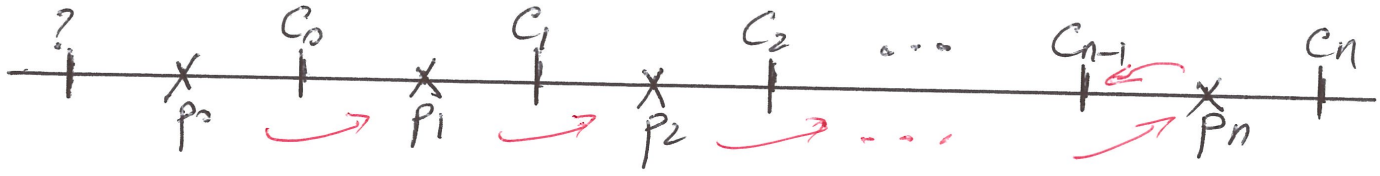
Thus the predecessor is

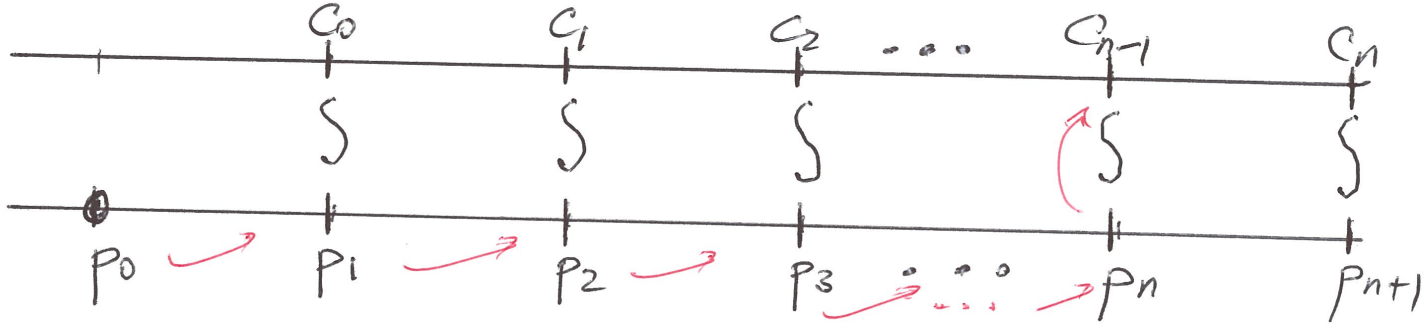
$$\lambda n.n\ (\lambda p.p\ (\lambda x.x)(\lambda f x.f\ (p\ f\ x)))\ (\lambda f x s z.z)$$

(found in the Summer of 1992)

Diagram


C0
C1
C2
Cn-1
Cn
P0
P1
P2
Pn-1
Pn


?
C0
C1
C2
Cn-1
Cn
P0
P1
P2
Pn


C0
C1
C2
Cn-1
Cn
P0
P1
P2
P3
Pn
Pn+1

My favorite predecessor

$$\mathsf{c}_n \doteq \mathsf{c}_n\ \mathsf{succ}\ \mathsf{c}_0 \qquad (*)$$

My favorite predecessor

$$\mathsf{c}_n \doteq \mathsf{c}_n\ \mathsf{succ}\ \mathsf{c}_0 \qquad (*)$$

$$\mathsf{c}_{n+1} \doteq \mathsf{c}_n\ \mathsf{succ}\ \mathsf{c}_1$$

My favorite predecessor

$$\mathsf{c}_n \doteq \mathsf{c}_n \ \mathsf{succ}\ \mathsf{c}_0 \qquad (*)$$

$$\mathsf{c}_{n+1} \doteq \mathsf{c}_n \ \mathsf{succ}\ \mathsf{c}_1$$

$$\mathsf{succ}^\circ :\doteq \lambda n.n \ \ \mathsf{succ}\ \mathsf{c}_1$$

is a successor on Church numerals

My favorite predecessor

$$\mathsf{c}_n \doteq \mathsf{c}_n \ \mathsf{succ}\ \mathsf{c}_0 \qquad (*)$$

$$\mathsf{c}_{n+1} \doteq \mathsf{c}_n \ \mathsf{succ}\ \mathsf{c}_1$$

$$\mathsf{succ}^\circ :\doteq \lambda n.n \ \ \mathsf{succ}\ \mathsf{c}_1$$

is a successor on Church numerals

$$\mathsf{succ}^\circ\ (\lambda f x.\mathsf{c}_0) \doteq \mathsf{c}_0$$

My favorite predecessor

$$\mathsf{c}_n \doteq \mathsf{c}_n\ \mathsf{succ}\ \mathsf{c}_0 \qquad (*)$$

$$\mathsf{c}_{n+1} \doteq \mathsf{c}_n\ \mathsf{succ}\ \mathsf{c}_1$$

$$\mathsf{succ}^\circ :\doteq \lambda n.n\ \ \mathsf{succ}\ \mathsf{c}_1$$

is a successor on Church numerals

$$\mathsf{succ}^\circ\ (\lambda f x.\mathsf{c}_0) \doteq \mathsf{c}_0$$

The predecessor is

$$\mathsf{pred} := \lambda n.n\ \ \mathsf{succ}^\circ\ (\lambda f x.\mathsf{c}_0)$$

Or, in the desugared, normal form (size 25):

$$\lambda n.n\ (\lambda p.p\ (\lambda c f x.f(c f x))(\lambda x.x))\ (\lambda f x s z.z)$$

My favorite predecessor

$$\mathsf{c}_n \doteq \mathsf{c}_n\ \mathsf{succ}\ \mathsf{c}_0 \qquad (*)$$

$$\mathsf{c}_{n+1} \doteq \mathsf{c}_n\ \mathsf{succ}\ \mathsf{c}_1$$

$$\mathsf{succ}^\circ :\doteq \lambda n.n\ \ \mathsf{succ}\ \mathsf{c}_1$$

is a successor on Church numerals

$$\mathsf{succ}^\circ\ (\lambda f x.\mathsf{c}_0) \doteq \mathsf{c}_0$$

The predecessor is

$$\mathsf{pred} := \lambda n.n\ \ \mathsf{succ}^\circ\ (\lambda f x.\mathsf{c}_0)$$

Or, in the desugared, normal form (size 25):

$$\lambda n.n\ (\lambda p.p\ (\lambda c f x.f(c f x))(\lambda x.x))\ (\lambda f x s z.z)$$

succ° is a meta-circular successor:
raise the level, extend the domain

Outline

Resolving my old puzzle

$$\mathsf{c}_n := \quad \lambda f z.\, f^{(n)}\, z$$

$$\begin{aligned} \mathsf{p}_{nz} &:= \quad \mathsf{Some}_z^{(n)}\ \mathsf{None}_z \qquad \text{where} \\ \mathsf{None}_z &:= \quad \lambda k.\, z \\ \mathsf{Some}_z &:= \quad \lambda a.\lambda k.\, k\ a \end{aligned}$$

Pattern-matching: un-application

$$\mathsf{p}_{nz}\ \mathsf{id} = \begin{cases} \mathsf{p}_{(n-1)z} & \text{if } n > 0 \\ z & \text{otherwise} \end{cases}$$

Reification	$\mathsf{c}_n \mapsto \mathsf{p}_{nz}$	$\mathsf{reif}_z := \lambda n.\, n\ \mathsf{Some}_z\ \mathsf{None}_z$
Reflection	$\mathsf{p}_{nz} \mapsto f^{(n)}\, z$	$\mathsf{refl}_f := \mathsf{fix}\ \lambda s.\lambda p.\, p\ (\lambda q.\, f\ (s\ q))$

Outline

Looking back

- Church numerals c_n (Koan) with constant c_0 and unary operation succ
- Numerals $\mathtt{p}_n$, built from constant $\mathtt{p}_0$ and unary operation supp
- $\mathsf{X\ option}$ data type, with constant None and unary operation Some

Looking back

- Church numerals c_n (Koan) with constant c_0 and unary operation succ
  F-algebra $F(X) := 1 + X$
- Numerals $\mathtt{p}_n$, built from constant $\mathtt{p}_0$ and unary operation supp
  F-algebra $F(X) := 1 + X$
- $\mathsf{X\ option}$ data type, with constant None and unary operation Some
  F-algebra/data type $F(X) := 1 + X$

Looking back

- Church numerals c_n (Koan) with constant c_0 and unary operation succ
  Initial F-algebra $F(X) := 1 + X$
- Numerals $\mathtt{p}_n$, built from constant $\mathtt{p}_0$ and unary operation supp
  F-algebra $F(X) := 1 + X$
- $\mathsf{X\ option}$ data type, with constant None and unary operation Some
  F-algebra/data type $F(X) := 1 + X$

Looking back

- Church numerals c_n (Koan) with constant c_0 and unary operation succ
  Initial F-algebra $F(X) := 1 + X$
- Numerals p_n, built from constant p_0 and unary operation supp
  F-algebra $F(X) := 1 + X$
- $\mathsf{X\ option}$ data type, with constant None and unary operation Some
  F-algebra/data type $F(X) := 1 + X$

$$\mathsf{p}_n \ \doteq \mathsf{c}_n\ \mathsf{supp}\ \mathsf{p}_0 \qquad n > 0 \qquad (**)$$

is the unique homomorphism

Looking back

- Church numerals c_n (Koan) with constant c_0 and unary operation succ
  Initial F-algebra $F(X) := 1 + X$
- Numerals p_n, built from constant p_0 and unary operation supp
  F-algebra $F(X) := 1 + X$
- $\mathsf{X\ option}$ data type, with constant None and unary operation Some
  F-algebra/data type $F(X) := 1 + X$

$$\mathsf{p}_n \ \dot{:=}\ \mathsf{c}_n\ \mathsf{supp}\ \mathsf{p}_0 \qquad n > 0 \qquad (**)$$

is the unique homomorphism

$$\mathsf{c}_n \doteq \mathsf{c}_n\ \mathsf{succ}\ \mathsf{c}_0 \qquad (*)$$

the unique homomorphism from an initial algebra to itself must be the identity

Generalizing

General approach, for any algebraic data type:

- write down the F functor
- apply either Böhm-Berarducci or Scott-Mogensen encoding
- *mechanically* obtain an efficient predecessor/extractor

Outline

What's else in the paper

- Sample generalization: left and right branch of a binary tree
- Outline of correctness proofs
  (algebraic/equational proofs)

What's else in the paper

- Sample generalization: left and right branch of a binary tree
- Outline of correctness proofs
  (algebraic/equational proofs)

  If f, h, and g are such that $\mathsf{h} \circ \mathsf{f} \doteq \mathsf{g} \circ \mathsf{h}$ then

$$\mathsf{h} \circ (\mathsf{c}_n\ \mathsf{f}) \doteq (\mathsf{c}_n\ \mathsf{g}) \circ \mathsf{h} \qquad \forall n \geq 0$$

Conclusions

Our reality may be very much like theirs. All this might just be an elaborate simulation running inside a little device sitting on someone's table.
StarTrek TNG, Episode 6x12, "Ship in a Bottle"