An Introduction to Corecursive Algebras

Venanzio Capretta

in collaboration with
Tarmo Uustalu (Tallinn) and Varmo Vene (Tartu)

The Foundations Group / Brouwer Institute Seminar
Nijmegen, Tue 4 Dec 2007

Two example of recursive function definition:

$$\mathsf{qs} : \mathsf{List}(\mathbb{N}) \to \mathsf{List}(\mathbb{N})$$
$$\mathsf{qs}\,[] = []$$
$$\mathsf{qs}\,(x :: l) = (\mathsf{qs}\, l_{\leq x}) +\!\!+\; x :: (\mathsf{qs}\, l_{>x})$$

where

$$l_{\leq x} = [y \leftarrow l \mid y \leq x]$$
$$l_{>x} = [y \leftarrow l \mid y > x]$$

$$\mathsf{sp} :: \mathsf{Stream}(\mathbb{N}) \to \mathsf{Stream}(\mathbb{N})$$
$$\mathsf{sp}\,(x :: s) = x :: (\mathsf{sp}\, s_{\mathsf{odd}} \oplus \mathsf{sp}\, s_{\mathsf{even}})$$

where

If $s = y_1 :: y_2 :: y_3 :: y_4 :: \cdots$

$$s_{\mathsf{odd}} = y_1 :: y_3 :: \cdots$$
$$s_{\mathsf{even}} = y_2 :: y_4 :: \cdots$$

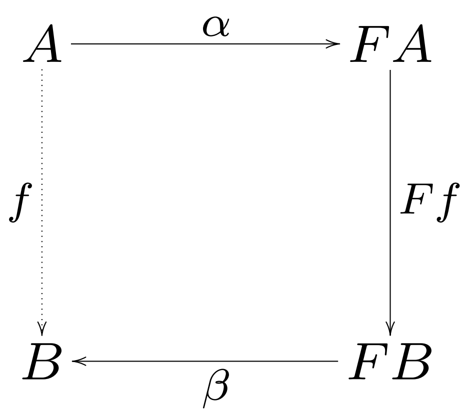
If $s' = z_1 :: z_2 :: z_3 :: z_4 :: \cdots$

$$s \oplus s' = y_1 + z_1 :: y_2 + z_2 :: \cdots$$

qs terminates by induction, but is not structurally recursive.
sp is productive, but not guarded.

Both define total functions, for different reasons.

Both recursive function definitions can be written as the solution of a Recursive Diagram:



F : Set $\longrightarrow$ Set functor
$\alpha : A \longrightarrow FA$ coalgebra
$\beta : FB \longrightarrow B$ algebra

A solution of the recursive diagram: $f = \beta \circ Ff \circ \alpha$.

Find conditions on α or β that ensure existence/uniqueness of f.

Quicksort example: $FX = \mathsf{Unit} + \mathbb{N} \times X \times X$, $A = B = \mathsf{List}(\mathbb{N})$.

$$
\begin{aligned}
&\alpha : \mathsf{List}(\mathbb{N}) \to \mathsf{Unit} + \mathbb{N} \times \mathsf{List}(\mathbb{N}) \times \mathsf{List}(\mathbb{N}) \\
&\alpha\,[] = \mathsf{inl}\,tt \\
&\alpha\,(x :: l) = \mathsf{inr}\,\langle x, l_{\leq x}, l_{>x} \rangle
\end{aligned}
$$

$$
\begin{aligned}
&\beta : \mathsf{Unit} + \mathbb{N} \times \mathsf{List}(\mathbb{N}) \times \mathsf{List}(\mathbb{N}) \to \mathsf{List}(\mathbb{N}) \\
&\beta\,(\mathsf{inl}\,u) = [] \\
&\beta\,(\mathsf{inr}\,\langle x, l_1, l_2 \rangle) = l_1 +\!\!+\, x :: l_2
\end{aligned}
$$

$$
\begin{array}{ccc}
\mathsf{List}(\mathbb{N}) & \xrightarrow{\alpha} & \mathsf{Unit} + \mathbb{N} \times \mathsf{List}(\mathbb{N}) \times \mathsf{List}(\mathbb{N}) \\
{\scriptstyle \mathsf{qs}}\,\downarrow & & \downarrow\,{\scriptstyle \mathsf{id}_{\mathsf{Unit}} + \mathsf{id}_{\mathbb{N}} \times \mathsf{List}(\mathsf{qs}) \times \mathsf{List}(\mathsf{qs})} \\
\mathsf{List}(\mathbb{N}) & \xleftarrow[\beta]{} & \mathsf{Unit} + \mathbb{N} \times \mathsf{List}(\mathbb{N}) \times \mathsf{List}(\mathbb{N})
\end{array}
$$

Stream Example: $FX = \mathbb{N} \times X \times X$, $A = \mathsf{Stream}(\mathbb{N})$, $B = \mathsf{Stream}(\mathbb{N})$.

$$\alpha : \mathsf{Stream}(\mathbb{N}) \to \mathbb{N} \times \mathsf{Stream}(\mathbb{N}) \times \mathsf{Stream}(\mathbb{N})$$
$$\alpha\,(x :: s) = \langle x, s_{\mathsf{odd}}, s_{\mathsf{even}} \rangle$$

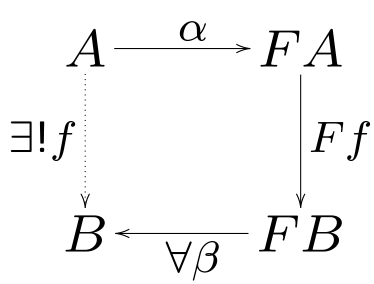
$$\beta : \mathbb{N} \times \mathsf{Stream}(\mathbb{N}) \times \mathsf{Stream}(\mathbb{N}) \to \mathsf{Stream}(\mathbb{N})$$
$$\beta\,\langle x, s_1, s_2 \rangle = x :: s_1 \oplus s_2$$

$$\begin{array}{ccc} \mathsf{Stream}(\mathbb{N}) & \xrightarrow{\alpha} & \mathbb{N} \times \mathsf{Stream}(\mathbb{N}) \times \mathsf{Stream}(\mathbb{N}) \\ {\scriptstyle \mathsf{sp}}\big\downarrow & & \big\downarrow{\scriptstyle \mathsf{id}_{\mathbb{N}} \times \mathsf{sp} \times \mathsf{sp}} \\ \mathsf{Stream}(\mathbb{N}) & \xleftarrow[\beta]{} & \mathbb{N} \times \mathsf{Stream}(\mathbb{N}) \times \mathsf{Stream}(\mathbb{N}) \end{array}$$

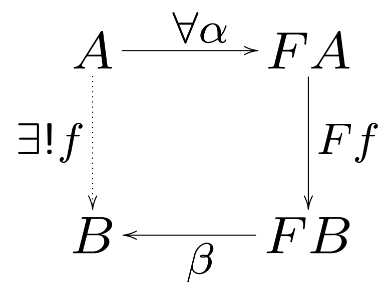
Existence/Unicity has a different origin in the two examples.
For qs it derives from a property of α.
For sp it derives from a property of β.

α is a recursive coalgebra if, for every algebra β, there exists a unique f satisfying the recursive diagram.
(G. Osius, 1974; P. Taylor, 1996; CUV, 2004-6; Adàmek, Lüke, Milius, 2007)
β is a corecursive algebra if, for every coalgebra α, there exists a unique f satisfying the recursive diagram.

Recursive Coalgebra α



CoRecursive Algebra β



In Set, $\mathcal{P}$ powerset functor. $\mathcal{P}(X) =$ subsets of X.
$\mathcal{P}$ doesn't have any fixed points.
No initial algebra. No terminal coalgebra.

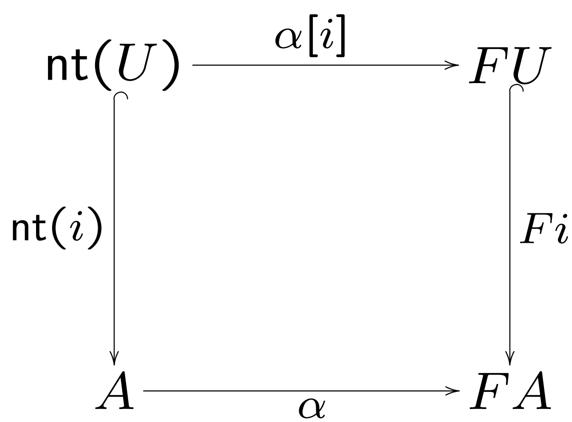
But $\mathcal{P}$ has recursive coalgebras.
$\mathcal{P}$-coalgebras are relations:

$$\begin{array}{ccc} \alpha : A \to \mathcal{P}A & \Longleftrightarrow & R_\alpha \subseteq A \times A \\ x_1 \in \alpha(x_2) & \Longleftrightarrow & x_1 \; R_\alpha \; x_2 \\ \alpha \text{ recursive} & \Longleftrightarrow & R_\alpha \text{ well-founded} \end{array}$$

Proof: Direct or
Special case of a more general result (Taylor, 1996).
Generalize the notion of well-founded relation to any functor and coalgebra.

Let F be a functor and $\alpha : A \to FA$ a coalgebra.

Idea: Generate the *accessible* part of A
by iterating the *next-time operator* nt (Jacobs):
Given a subobject $U \overset{i}{\hookrightarrow} A$, define $\mathsf{nt}(U)$ as the result of the pullback



Intuitively: $\mathsf{nt}(U) = \{x \in A \mid \alpha(x) \in FU\}$.

Induction principle for a well-founded relation R:

If U is a subset of A such that:

for every $x \in A$, if $\forall y \in A, y \, R \, x \to y \in U$, then $x \in U$;

then $U = A$.

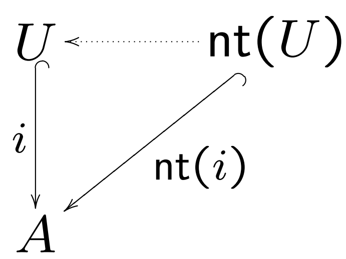
Definition (Taylor):

α is a well-founded coalgebra if:

For every subobject $i : U \hookrightarrow A$ such that:

$\mathsf{nt}(i)$ factors through i;

then $U \cong A$.



We can define the accessible part of A (w.r.t. α) by a fixed point of nt obtained by iterating nt starting with the empty subobject.

$$A_0 = \emptyset \quad A_1 = \mathsf{nt}(A_0) \quad \cdots \quad A_{i+1} = \mathsf{nt}(A_i)$$
$$A_\omega = \varinjlim_{i<\omega} A_i \quad (= \cup_{i<\omega} A_i)$$
$$A_\gamma = \varinjlim_{\delta<\gamma} A_i$$

Accessible part of A (Bove/C., 2001):
$A_\zeta = \mathsf{nt}(A_\zeta)$, if it exists.

α is a inductive coalgebra if
There is an ordinal ζ such that $A_\zeta = A$.

Theorem (Taylor)

Under some conditions on the category and the functor F;

The three conditions:

- α is a recursive coalgebra,
- α is a well-founded coalgebra,
- α is a inductive coalgebra,

are equivalent.

Does the previous characterization hold
also for corecursive algebras?
NO

Dual of the next-time operator.

The dual of a subset is a quotient, $Q = (B/\equiv)$.
Categorically, an epimorphisms $q : B \twoheadrightarrow Q$.

The next-time of Q, $\mathsf{nt}(Q)$ is given by a push-out:

$$\begin{array}{ccc} FB & \xrightarrow{\beta} & B \\ {\scriptstyle Fq}\big\downarrow & & \big\downarrow{\scriptstyle \mathsf{nt}(q)} \\ FQ & \xrightarrow[\beta[q]]{} & \mathsf{nt}(Q) \end{array}$$

Intuitively: $\mathsf{nt}(Q) = (B/\equiv')$ where
$\equiv'$ is the reflexive/transitive closure of:
$\forall y_1, y_2 \in FB, y_1 \equiv^F y_2 \to \beta(y_1) \equiv' \beta(y_2)$.

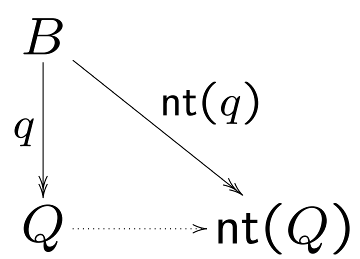
Definition:

β is a discriminating algebra if:

For every quotient $q : B \twoheadrightarrow Q$ such that:

$\mathsf{nt}(q)$ factors through q;

then $Q \cong B$.



Sequence of iterations of nt:

$$Q_0 = \mathsf{Unit} \equiv (B/\equiv_0) \text{ with } \equiv_0 \text{ the total relation}$$
$$Q_{i+1} = \mathsf{nt}(Q_i)$$
$$Q_{\delta<\gamma} = \varprojlim_{\delta<\gamma} Q_\delta$$

Definition:

β is a focusing algebra if

There is an ordinal ζ such that $Q_\zeta = Q$.

Problem:

Transfinite iterations of nt are not necessarily quotients of B.

focusing $\underset{\not\leftarrow}{\longrightarrow}$ discriminating   focusing $\underset{\not\leftarrow}{\longrightarrow}$ corecursive

Counterexample 1

Consider the following algebra for the functor $FX = X \times X$:

$$\begin{array}{l} \beta : 3 \times 3 \to 3 \\ \beta(1,2) = 2 \\ \beta(n,m) = 0 \text{ otherwise} \end{array}$$

β is corecursive

β is neither discriminating nor focusing:

$$Q_0 = \mathsf{Unit} \quad Q_1 = \{1, \bot\} \quad \text{where } \bot = \{0, 2\}$$

Q_1 is already a fixed point.

Counterexample 2

Consider the following algebra for the functor $FX = X \times X$:

$$\beta : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$$
$$\beta(1, m) = m + 2$$
$$\beta(n, m) = 0 \quad \text{otherwise}$$

β is corecursive and discriminating

β is not focusing:

$$Q_0 = \{\bot\} \qquad Q_3 = \{0, 1, 2, 3, 5, \bot\} \qquad Q_\omega = \mathbb{N} \cup \{\bot\}$$
$$Q_1 = \{1, \bot\} \qquad Q_4 = \{0, 1, 2, 3, 4, 5, 7, \bot\}$$
$$Q_2 = \{0, 1, 3, \bot\} \qquad Q_i = \{0, \ldots, 2i - 3, 2i - 1, \bot\}$$

Coda: A recursive diagram that has a unique solution but such that α is not recursive and β is not corecursive.

$$\mathsf{down\_runs} : \mathsf{Stream}(\mathbb{N}) \to \mathsf{Stream}(\mathbb{N})$$
$$\mathsf{down\_runs}\, s = \mathsf{drf}\, \langle 1, s \rangle$$

$$\mathsf{drf} : \mathbb{N} \times \mathsf{Stream}(\mathbb{N}) \to \mathsf{Stream}(\mathbb{N})$$
$$\mathsf{drf}\, \langle n, (x_1 :: x_2 :: s) \rangle = \begin{cases} \mathsf{drf}\, \langle (n+1), (x_2 :: s) \rangle & \text{if } x_1 > x_2 \\ n :: \mathsf{drf}\, \langle 1, (x_2 :: s) \rangle & \text{if } x_1 \le x_2 \end{cases}$$

Diagram: $FX = X + \mathbb{N} \times X$, $A = \mathbb{N} \times \mathsf{Stream}(\mathbb{N})$, $B = \mathsf{Stream}(\mathbb{N})$.

$$\alpha : \mathbb{N} \times \mathsf{Stream}(\mathbb{N}) \to \mathbb{N} \times \mathsf{Stream}(\mathbb{N}) + \mathbb{N} \times \mathbb{N} \times \mathsf{Stream}(\mathbb{N})$$

$$\alpha\,\langle n, x_1 :: x_2 :: s\rangle = \begin{cases} \mathsf{inl}\,\langle (n+1), (x_2 :: s)\rangle & \text{if } x_1 > x_2 \\ \mathsf{inr}\,\langle n, 1, (x_2 :: s)\rangle & \text{if } x_1 \leq x_2 \end{cases}$$

$$\beta : \mathsf{Stream}(\mathbb{N}) + \mathbb{N} \times \mathsf{Stream}(\mathbb{N}) \to \mathsf{Stream}(\mathbb{N})$$

$$\beta\,(\mathsf{inl}\,u) = u$$

$$\beta\,(\mathsf{inr}\,\langle n, u\rangle) = n :: u$$

$$\begin{array}{ccc} \mathbb{N} \times \mathbb{S} & \xrightarrow{\alpha} & \mathbb{N} \times \mathbb{S} + \mathbb{N} \times \mathbb{N} \times \mathbb{S} \\ {\scriptstyle \mathsf{drf}} \downarrow & & \downarrow {\scriptstyle \mathsf{drf} + \mathsf{id}_{\mathbb{N}} \times \mathsf{drf}} \\ \mathbb{S} & \xleftarrow[\beta]{} & \mathbb{S} + \mathbb{N} \times \mathbb{S} \end{array}$$