Stream processors and comodels

CALCO 2021

Richard Garner · Centre of Australian Category Theory

The type of streams of elements of a type A is

$$A^{\mathbb{N}} = \{(a_0, a_1, a_2, \dots)\}.$$

This has a well known characterisation as a final coalgebra:

$$A^{\mathbb{N}} = \nu X.\, A \times X \quad \text{via} \quad \begin{aligned} A^{\mathbb{N}} &\longrightarrow A \times A^{\mathbb{N}} \\ \vec{a} &\longmapsto (a_0, \partial\vec{a}) \end{aligned}$$

$$\partial\vec{a} = (a_1, a_2, a_3, \dots)$$

ie. cts for Baire topology

An A-B-stream processor is a map $f : A^{\mathbb{N}} \longrightarrow B^{\mathbb{N}}$ which is productive: each B-token of $f(\vec{a})$ determined by a finite initial segment of $\vec{a}$.

Q: can we characterise stream processors as a final coalgebra?

A1: "Yes, sort of" (Hancock, Pattinson, Ghani 2009).

An A-B-stream processing state machine is a set S of states, equipped with a coalgebra structure:

$$\gamma : S \longrightarrow T_A(B \times S)$$

A-ary branching trees with leaves in $B \times S$

Each state of (S, γ) encodes some productive $A^{\mathbb{N}} \to B^{\mathbb{N}}$.

Eg: "add 1": $2^{\mathbb{N}} \to 2^{\mathbb{N}}$ encoded by state c of $S = \{c, z\}$ with

$$\gamma :\quad c \longmapsto \begin{tikzcd}[column sep=small] (1,z) & & (0,c) \\ & \bullet \arrow[ul, dash, "0"] \arrow[ur, dash, "1"'] & \end{tikzcd} \qquad z \longmapsto \begin{tikzcd}[column sep=small] (0,z) & & (1,z) \\ & \bullet \arrow[ul, dash, "0"] \arrow[ur, dash, "1"'] & \end{tikzcd}$$

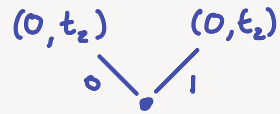
1 1 0 1 0 0 1 . . . $\longmapsto$ 0 0 1 1 0 0 1 . . .

The set of A-B-stream processors is the terminal A-B-stream processing state machine $I_{AB} := \nu X.\, T_A(B \times X)$

intensional

But ... different states of I_{AB} encode the same fn $A^{\mathbb{N}} \to B^{\mathbb{N}}$!

Eg: $f : 2^{\mathbb{N}} \longrightarrow 2^{\mathbb{N}}$ with $f(\vec{a}) = (0,0,0,\ldots)$ encoded by both

$t_1 \longmapsto (0, t_1)$ and $t_2 \longmapsto$ 

in I_{AB}.

A2: "Yes, sort of" (me, today).

An A-ary magma is a set X t/w $\xi : X^A \longrightarrow X$ (s.t. nothing!)

More generally, have A-ary magmas in any caty w/ products ...

and A-ary comagmas in catys w/ coproducts. $X \to \overbrace{X + \cdots + X}^{A}$

Theorem The set of productive functions $A^{\mathbb{N}} \to B^{\mathbb{N}}$ underlies the terminal B-ary comagma in the caty of A-ary magmas

only sort of a terminal coalgebra

Notions of computation

Moggi: notions of computation $\leftrightsquigarrow$ strong monads $\mathbb{T}$

for us: on $\underset{\sim}{\mathrm{Set}}$

computations returning a value in A $\leftrightsquigarrow$ elements of $T(A)$

sequencing by monadic binding $\leftrightsquigarrow$ composition in Kleisli caty $\mathrm{K}\ell(\mathbb{T})$

Eg: state, stack, non-determinism, probabilistic, input/output, ...

Plotkin–Power: notions of computation $\leftrightsquigarrow$ **presentations of strong monads**

by operations and equations

Eg: monad $\mathbb{T}_A$ for input from an alphabet A is generated by the operation $\mathrm{read} \in T_A(A)$ and no equations.

(Yields computations like $\mathrm{read}(\lambda n.\, \mathrm{read}(\lambda m.\, n+m)) \in T_{\mathbb{N}}(\mathbb{N})$).

Models & comodels

Given a monad $\mathbb{T}$ on $\underset{\sim}{\mathrm{Set}}$, can define $\mathbb{T}$-models in any caty $\mathcal{C}$ with products: involves an object $X \in \mathcal{C}$ t/w

$$[\![\sigma]\!] : X^A \longrightarrow X$$

we assume a presentation

for each generating operation $\sigma \in T(A)$, satisfying equations.

Eg: Each $T(A)$ is a (free) $\mathbb{T}$-model in $\underset{\sim}{\mathrm{Set}}$; actually, every $\mathbb{T}$-model in $\underset{\sim}{\mathrm{Set}}$ is a quotient of a free one. ← computations up to ...

Eg: A $\mathbb{T}_A$-model in $\mathcal{C}$ is $X \in \mathcal{C}$ t/w $\xi : X^A \longrightarrow X$ ← A-ary magma!

A $\mathbb{T}$-comodel in $\mathcal{C}$ is a $\mathbb{T}$-model in $\mathcal{C}^{op}$.

Eg: A $\mathbb{T}_A$-comodel in $\mathcal{C}$ is $X \in \mathcal{C}$ t/w $\xi : X \longrightarrow \underbrace{X + \cdots + X}_{A}$ ← A-ary comagma!

Comodels in $\mathbf{Set}$

Power-Shkaravska; Plotkin-Power; Uustalu:

$\mathbf{Set}$-comodels of $\mathbb{T}$ $\leftrightsquigarrow$ environments to evaluate $\mathbb{T}$-computations

Eg: a $\mathbb{T}_A$-comodel in $\mathbf{Set}$ is a state machine for answering requests for A-tokens:

current state   next token read   next state

set of states

$$(S, [\![\mathrm{read}]\!] : S \longrightarrow A \times S)$$

This induces a co-interpretation for each computation $t \in T_A(B)$:

initial state   final state

$$[\![t]\!] : S \longrightarrow B \times S$$

B-value

Eg: $[\![\mathrm{read}]\!] : \begin{array}{l} a \longmapsto (b, 11) \\ b \longmapsto (c, 4) \\ c \longmapsto (b, 5) \end{array} \rightsquigarrow [\![\mathrm{read}(\lambda n.\, \mathrm{read}(\lambda m.\, n+m))]\!] : \begin{array}{l} a \longmapsto (c, 15) \\ b \longmapsto (b, 9) \\ c \longmapsto (c, 9) \end{array}$

Note: the final $\mathbb{T}_A$-comodel in $\mathbf{Set}$ is the set $A^{\mathbb{N}}$ of A-streams.

Residual comodels

Katsumata, Rivas, Uustalu; Ahman, Bauer:

$\mathbb{T}, \mathbb{R}$ are monads

An $\mathbb{R}$-residual $\mathbb{T}$-comodel is a $\mathbb{T}$-comodel in $\mathrm{Kl}(\mathbb{R})$.

$\mathbb{R}$-residual $\mathbb{T}$-comodels $\leftrightsquigarrow$ environments to translate $\mathbb{T}$-computations into $\mathbb{R}$-computations

An $\mathbb{R}$-residual $\mathbb{T}$-comodel involves a set S of states and

$$[\![\sigma]\!] : S \longrightarrow R(A \times S)$$

initial state

computation giving final state & return value

for each generating $\sigma \in T(A)$ plus equations.

Eg: a $\mathbb{T}_A$-residual $\mathbb{T}_B$-comodel is a set S of states t/w

$$\gamma : S \longrightarrow T_A(B \times S)$$

i.e. an A-B-stream processing state machine.

And the final residual comodel is I_{AB}.

The extent of an intensional stream processor

Abstractly: $\mathbb{R}$-comodel M in $\underset{\sim}{\mathrm{Set}}$ $\leftrightsquigarrow$

$$\begin{tikzcd} \underset{\sim}{\mathrm{Set}} \arrow[r, "(-)\times M"] \arrow[d] & \underset{\sim}{\mathrm{Set}} \\ \mathrm{K}\ell(\mathbb{R}) \arrow[ur, dashed, "{[\![-]\!]_M}"'] & \end{tikzcd}$$

Linton, Uustalu

$\mathbb{R}$-residual $\mathbb{T}$-comodel S $\leftrightsquigarrow$

$$\begin{tikzcd} \underset{\sim}{\mathrm{Set}} \arrow[r, "(-)\times S"] \arrow[d] & \underset{\sim}{\mathrm{Set}} \arrow[d] \\ \mathrm{K}\ell(\mathbb{T}) \arrow[r, dashed, "{[\![-]\!]_S}"'] & \mathrm{K}\ell(\mathbb{R}) \end{tikzcd}$$

Compose to get tensor product $\mathbb{T}$-comodel $S \cdot M$ with state-set $S \times M$.

The extent function of an A-B-stream processing state machine S is the unique $\mathbb{T}_B$-comodel map

final comodels

productive fn associated to s

tensor product

$$\begin{aligned} S \cdot A^{\mathbb{N}} &\longrightarrow B^{\mathbb{N}} \\ (s, \vec{a}) &\longmapsto f_s(\vec{a}) \end{aligned}$$

Bimodels

Freyd (1966!); Tall, Wraith; Bergman, Hausknecht

A $\mathbb{T}$-$\mathbb{R}$-bimodel is a $\mathbb{T}$-comodel in $\mathbb{R}$-$\mathrm{Mod}(\mathrm{Set})$.

This involves a set S with $\mathbb{R}$-model structure, plus

$$[\![\sigma]\!] : S \longrightarrow \underbrace{S + \cdots + S}_{A}$$

coprod. in $\mathbb{R}$-Mod

and equations

If S is a free $\mathbb{R}$-model, this is just a $\mathbb{R}$-residual $\mathbb{T}$-comodel.

Actually, any bimodel is a quotient of a residual comodel by some bisimulation:

A bisimulation on a $\mathbb{R}$-residual $\mathbb{T}$-comodel S is an $\mathbb{R}$-congruence $\sim$ on S s.t. each $[\![\sigma]\!] : S \longrightarrow R(A \times S)$ sends $\sim$-congruent terms to $\sim'$-congruent ones.

Stream processors as a final bimodel

Theorem The set of productive functions $A^{\mathbb{N}} \to B^{\mathbb{N}}$ underlies the final Π_A-Π_B-bimodel.

quotient of final residual comodel I_{AB} by largest bisimulation

Proof Have an adjunction

induced by topological comodel $A^{\mathbb{N}}$

$$\begin{tikzcd} \Pi_A\text{-}\mathrm{Mod}(\mathrm{Set}) \arrow[r, phantom, "\top"] \arrow[r, bend right, "(-)\otimes A^{\mathbb{N}}"'] & \mathrm{Top} \arrow[l, bend right, "\mathrm{Top}(A^{\mathbb{N}},-)"'] \end{tikzcd}.$$

By a fan theorem argument, the right adjoint preserves coproducts.

So induce an adjunction

$$\begin{tikzcd} \Pi_B\text{-}\mathrm{Comod}(\Pi_A\text{-}\mathrm{Mod}(\mathrm{Set})) \arrow[r, phantom, "\top"] \arrow[r, bend right, "(-)\otimes A^{\mathbb{N}}"'] & \Pi_B\text{-}\mathrm{Comod}(\mathrm{Top}) \arrow[l, bend right, "\mathrm{Top}(A^{\mathbb{N}},-)"'] \end{tikzcd}$$

whose right adjoint sends the final comodel $B^{\mathbb{N}}$ to a final bimodel $\mathrm{Top}(A^{\mathbb{N}}, B^{\mathbb{N}})$. $\square$