Observation, state and behaviour (an introduction to coalgebra)

Luís S. Barbosa

HASLab - INESC TEC
Universidade do Minho
Braga, Portugal

27 February, 2013

The questions

How to specify and reason about dynamic, reactive, state-based systems?

- persistence, i.e., internal state and state transitions
- continued interaction along the whole computational process
- potential infinite behaviour
- observability through well-defined interfaces to ensure flow of data

How to do it in a generic way?

Behaviour & Interaction

[R. Milner, 1997]
Thus software, from being a prescription for how to do something — in Turing's terms a "list of instructions" — becomes much more akin to a description of behaviour, not only programmed on a computer, but occurring by hap or design inside or outside it.

[B. Jacobs, 2005]
The subject of Computer Science is not information processing or symbol manipulation, but generated behaviour.

Behaviour & Interaction

Behavioural abstractions aims at

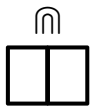
- representing state-based systems
- dealing with objects, processes, services whose semantics is inherently observational
- handling infinite types
- specifying finitely otherwise infinitely axiomatizable abstract data types
- ...

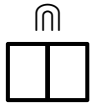
Antecipating

B^* – finite sequences

$$[\text{nil}, \text{cons}] : \mathbf{1} + B \times L \longrightarrow L$$

In general:

a tool box: 

an assembly process:  artifact $\xrightarrow{a}$ artifact

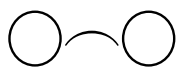
- abstract data structures as (initial) algebras
- emphasis is on construction

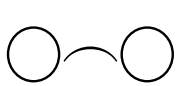
Antecipating

B^ω – streams

$$\langle \mathsf{at}, \mathsf{m} \rangle : U \longrightarrow B \times U$$

In general:

a lens: 

an observation structure: universe $\xrightarrow{c}$  universe

- abstract behavioural structures as (final) coalgebras
- emphasis is on observation

Antecipating

- The lens describes the shape (or signature) of legal observations, whose collection corresponds to the system's generated behaviour.

- The observation structure describes the system's one-step dynamics; It's a sort of behaviour generating machine.

Antecipating

Coalgebra as the mathematics of computational dynamics

Basic References:

- Universal coalgebra: A theory of systems, J. Rutten, *Theor. Comp. Sci.*, 249(1), 2000 (previous CWI Rep, 1996).
- A tutorial on (co)algebras and (co)induction, B. Jacobs and J. Rutten, *EATCS Bulletin*, 62, 1997.
- Lectures on semantics : The initial algebra and final coalgebra perspectives, P. Aczel, *Lect. for 1995 Marktoberdorf School*, Springer, 1997.
- An introduction to coalgebra, J. Adamek, *Theory and Applications of Categories*, 14(8), 2005.
- Elements of the general theory of coalgebras, H. P- Gumm, Lutacs'99 Lect. Notes, 1999.

A parenthesis for the functional programmer

(...

A parenthesis for the functional programmer

There are several ways of glueing functions
... each one leading to a different way of aggregating information:

Pipelining: leading to function space B^A (dependency)

$$A \xrightarrow{\quad f \quad} B \xrightarrow{\quad g \quad} C$$

Conjunction: leading to product $A \times B$ (spatial aggregation)

$$C \xrightarrow{\quad \langle f,g \rangle \quad} A \times B$$

where $\langle f, g \rangle (c) = (f\ c, g\ c)$

A parenthesis for the functional programmer

Disjunction: leading to coproduct (or disjoint union) $A + B$ (choice)

$$A + B = \{1\} \times A \cup \{2\} \times B \xrightarrow{[f,g]} C$$

where $$[f, g](x) = (x = (1, a)) \to f\ a$$
$$(x = (2, b)) \to g\ b$$

Constants & points:

empty	$() : \emptyset \longrightarrow A$
collapse	$! : A \longrightarrow \mathbf{1}$
points	$\underline{a} : \mathbf{1} \longrightarrow A$

A parenthesis for the functional programmer

The underlying 'semantic universe' assumes an elementary

- space of types and typed arrows ...
- with the structure of a (partial) monoid
- ... taken in the sequel as sets and set-theoretical functions

upon which combinators are defined by universal arrows

- associated to the product, sum and exponential constructions
- which behave ... as they should (formally, form a ccc)

End of parenthesis

...)

- Introduction
- Motivating example: Automata
- Going generic: Coalgebras
- Application example: Transducers

Automata

state space	U
transition function	$\mathsf{m} : U \longrightarrow U$
attribute (or label)	$\mathsf{at} : U \longrightarrow B$

i.e.,

$$p \;=\; \langle \mathsf{at}, \mathsf{m} \rangle : U \longrightarrow B \times U$$

Notation:

$$u \longrightarrow_p u' \quad \Leftrightarrow \quad \mathsf{m}\, u = u'$$
$$u \downarrow_p b \quad \Leftrightarrow \quad \mathsf{at}\, u = b$$

Automata

The behaviour of p at (from) a state $u \in U$ is revealed by successive observations (experiments):

$$\begin{array}{rcl} [\![p]\!]\, u & = & [\mathsf{at}\ u, \mathsf{at}\ (\mathsf{m}\ u), \mathsf{at}\ (\mathsf{m}\ (\mathsf{m}\ u)), \ldots] \\ [\![p]\!] & = & \mathsf{cons} \cdot \langle \mathsf{at}, [\![p]\!] \cdot \mathsf{m} \rangle \end{array}$$

which means that

Automata behaviours are elements of B^ω (*i.e.*, streams)

Automata

Example: A twist automata

state space	$U = \mathbb{N} \times \mathbb{N}$
transition function	$\mathsf{m}\,(n, n') \;=\; (n', n)$
attribute	$\mathsf{at}\,(n, n') \;=\; n$

i.e.,

$$\mathsf{twist} \;=\; \langle \pi_1, \mathsf{s} \rangle$$

Automata

Example: A stream automata

state space	$U = B^\omega$
transition function	$\mathsf{m}\, s = \mathsf{tl}\, s$
attribute	$\mathsf{at}\, s = \mathsf{hd}\, s$

i.e.,

$$\omega = \langle \mathsf{hd}, \mathsf{tl} \rangle$$

Automata behaviours form themselves an automata

Automata morphisms

A morphism

$$h : p \longrightarrow q$$

where

$$\begin{aligned} p &= \langle \mathrm{at}, \mathrm{m} \rangle : U \longrightarrow B \times U \\ q &= \langle \mathrm{at}', \mathrm{m}' \rangle : V \longrightarrow B \times V \end{aligned}$$

is a function $h : U \longrightarrow V$ such that

$$\begin{array}{ccc} U & \xrightarrow{p} & B \times U \\ {\scriptstyle h}\downarrow & & \downarrow{\scriptstyle \mathrm{id} \times h} \\ V & \xrightarrow{q} & B \times V \end{array}$$

i.e.,

$$\mathrm{at} = \mathrm{at}' \cdot h \quad \text{and} \quad h \cdot \mathrm{m} = \mathrm{m}' \cdot h$$

Behaviour as a morphism

Th: Behaviour $[\![p]\!]$ is an automata morphism from p to ω

because

$$
\begin{aligned}
& \mathsf{at} = \mathsf{hd} \cdot \mathsf{cons} \cdot \langle \mathsf{at}, [\![p]\!] \cdot \mathsf{m} \rangle \\
= & \quad \{ \ \mathsf{hd} \cdot \mathsf{cons} = \pi_1 \ \} \\
& \mathsf{at} = \pi_1 \cdot \langle \mathsf{at}, [\![p]\!] \cdot \mathsf{m} \rangle \\
= & \quad \{ \ \times \text{ cancellation} \ \} \\
& \mathsf{at} = \mathsf{at}
\end{aligned}
$$

and

$$
\begin{aligned}
& [\![p]\!] \cdot \mathsf{m} = \mathsf{tl} \cdot \mathsf{cons} \cdot \langle \mathsf{at}, [\![p]\!] \cdot \mathsf{m} \rangle \\
= & \quad \{ \ \mathsf{tl} \cdot \mathsf{cons} = \pi_2 \ \} \\
& [\![p]\!] \cdot \mathsf{m} = \pi_2 \cdot \langle \mathsf{at}, [\![p]\!] \cdot \mathsf{m} \rangle \\
= & \quad \{ \ \times \text{ cancellation} \ \} \\
& [\![p]\!] \cdot \mathsf{m} = [\![p]\!] \cdot \mathsf{m}
\end{aligned}
$$

Question

How to reason about automata behaviours?

Induction & Coinduction

Reasoning about B^*

$$\mathsf{len}(\mathsf{map}\, f\ l) \;=\; \mathsf{len}\, l$$

where functions are defined inductively by their effect on B^*
constructors

$$\begin{aligned} \mathsf{len}\,[] &= 0 \\ \mathsf{len}(h : t) &= 1 + \mathsf{len}\, t \end{aligned}$$

$$\begin{aligned} \mathsf{map}\, f\,[] &= [] \\ \mathsf{map}\, f(h : t) &= f(h) : \mathsf{map}\, f\ t \end{aligned}$$

These equations can be regarded as HASKELL definitions

Induction & Coinduction

Proof (by structural induction).

Base case is trivial. Then,

$$
\begin{aligned}
& \mathsf{len}(\mathsf{map}\, f\, (h : t)) \\
= & \qquad \{ \ \mathsf{map}\, f \ \text{definition} \ \} \\
& \mathsf{len}(f\, (h) : \mathsf{map}\, f\ t) \\
= & \qquad \{ \ \mathsf{len} \ \text{definition} \ \} \\
& 1 + \mathsf{len}(\mathsf{map}\, f\ t) \\
= & \qquad \{ \ \text{induction hypothesis} \ \} \\
& 1 + \mathsf{len}\ t \\
= & \qquad \{ \ \mathsf{len} \ \text{definition} \ \} \\
& \mathsf{len}(h : t)
\end{aligned}
$$

Induction & Coinduction

Inductive reasoning requires that, by repeatedly unfolding the definition, arguments become smaller, *i.e.*, closer to the elementary constructors

... but what happens if this unfolding process does not terminate?

Induction & Coinduction

Consider

$$\begin{aligned} \mathsf{map}\, f\, (h : t) &= (f\, h) : \mathsf{map}\, f\, t \\ \mathsf{gen}\, f\, x &= x : \mathsf{gen}\, f\, (f\, x) \end{aligned}$$

- definition unfolding does not terminate but ...
- ... reveals longer and longer prefixes of the result: every element in the result gets uniquely determined along this process

Strategy

To reason about circular definitions over infinite structures, our attention shifts from argument's structural shrinking to the progressive construction of the result which becomes richer in informational contents.

Induction & Coinduction

Reasoning about B^ω: the global view

Stream equality

$$\langle \forall\ n\ :\ n \geq 0 :\ s\,n\ =\ t\,n \rangle$$

can be established by induction over n
However, it

- requires a (workable) formula for arguments $s\,n$, $t\,n$, often not available
- does not scale easily to other behaviour types

Induction & Coinduction

Reasoning about B^ω: the local view

Two streams s and r are observationally the same if

- they have identical head observations: hd s = hd r,
- and their tails — tl s and tl r — support a similar verification.

Relation $R : B^\omega \longrightarrow B^\omega$ is a (stream) bisimulation iff

$$\langle x, y\rangle \in R \;\Rightarrow\; \text{hd } x = \text{hd } y \;\wedge\; \langle \text{tl } x, \text{tl } y\rangle \in R$$

(i.e., R is closed under the computational dynamics)

Induction & Coinduction

Coinduction as a proof principle:

- a systematic way of strengthening the statement to prove: from equality $s = r$ to a larger set R which contains pair $\langle s, r \rangle$
- ensuring that such a set is a bisimulation, *i.e.*, the closure of the original set under taking derivatives
- moreover, as a proof principle, it generalises from streams to a large class of behaviour types

Induction & Coinduction

$$\mathsf{map}_f \cdot \mathsf{gen}_f \cdot f$$

Check that R below is a bisimulation

$$R \;=\; \{\langle \mathsf{map}\, f\, (\mathsf{gen}\, f\, x)\,,\, \mathsf{gen}\, f\, (f\, x)\rangle |\; x \in \ldots, f \in \ldots\}$$

- $\mathsf{hd}\ (\mathsf{map}\, f\, (\mathsf{gen}\, f\, x)) \;=\; f\, x \;=\; \mathsf{hd}\ (\mathsf{gen}\, f\, (f\, x))$
- $\mathsf{tl}\ (\mathsf{map}\, f\, (\mathsf{gen}\, f\, x)) \;=\; \mathsf{map}\, f\ \ \mathsf{tl}\ \ (\mathsf{gen}\, f\, x)$ and $\mathsf{tl}\ (\mathsf{gen}\, f\, (f\, x)) \;=\; \mathsf{gen}\, f\, (f\, f\, x)$. Thus,

$$\langle \mathsf{tl}\ (\mathsf{map}\, f\, (\mathsf{gen}\, f\, x)), \mathsf{tl}\ (\mathsf{gen}\, f\, (f\, x))\rangle \in R$$

Remark:
In general, however, much larger relations have to be considered and the construction of bisimulations is not trivial

Coinduction calculationally

Existence and uniqueness of $[\![(p)]\!]$ can be captured by the following universal property:

$$k = [\![(p)]\!] \quad \Leftrightarrow \quad \omega \cdot k = (\mathrm{id} \times k) \cdot p$$

- Existence $\Leftrightarrow$ definition principle (co-recursion)
- Uniqueness $\Leftrightarrow$ proof principle (co-induction)

From which:

$$\begin{array}{rrcl} \text{cancellation} & \omega \cdot [\![(p)]\!] & = & (\mathrm{id} \times [\![(p)]\!]) \cdot p \\ \text{reflection} & [\![(\omega)]\!] & = & \mathrm{id}_\omega \\ \text{fusion} & [\![(p)]\!] \cdot h & = & [\![(q)]\!] \quad \text{if} \quad p \cdot h = (\mathrm{id} \times h) \cdot q \end{array}$$

An universal property

Example: fusion law

$$
\begin{array}{ll}
 & [\![p]\!] \cdot h \;=\; [\![q]\!] \\
\Leftrightarrow & \qquad \{ \text{ universal law } \} \\
 & \omega \cdot [\![p]\!] \cdot h \;=\; (\mathrm{id} \times ([\![p]\!] \cdot h)) \cdot q \\
\Leftrightarrow & \qquad \{ \text{ cancellation law and functoriality } \} \\
 & (\mathrm{id} \times [\![p]\!]) \cdot p \cdot h \;=\; (\mathrm{id} \times [\![p]\!]) \cdot (\mathrm{id} \times h) \cdot q \\
\Leftarrow & \qquad \{ \text{ function equality } \} \\
 & p \cdot h \;=\; (\mathrm{id} \times h) \cdot q
\end{array}
$$

An universal property

... from which the following (main) result is a direct corollary:

Th: morphisms preserve behaviour: $[\![(p)]\!] = [\![(q)]\!] \cdot h$

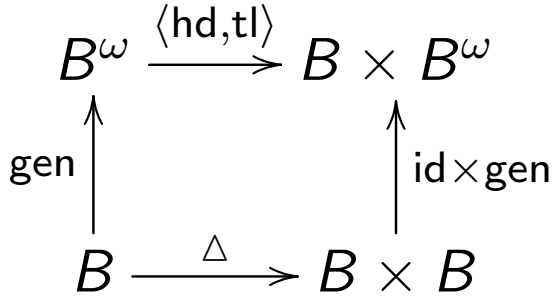
Definition by coinduction

- by a specification genetic inheritance
- by an explicit specification of behaviour under all observers
- by a recursive expression

depending on context and purpose

Definition by coinduction

by genetic inheritance



$$\mathsf{gen} \;=\; [\![(\triangle)]\!]$$

$\triangle$ carries the 'genetic inheritance' of the generating process

Definition by coinduction

by specification of behaviour under observers

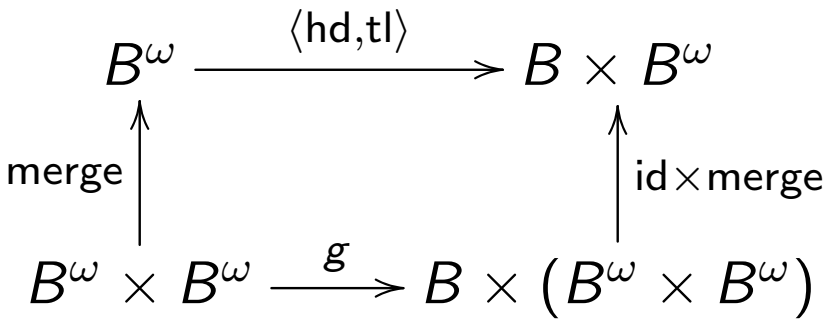
... equations come by diagram unfolding:

$$
\begin{aligned}
& (\mathsf{id} \times \mathsf{gen}) \cdot \triangle = \langle \mathsf{hd}, \mathsf{tl} \rangle \cdot \mathsf{gen} \\
= & \quad \{ \ \triangle \text{ definition} \ \} \\
& (\mathsf{id} \times \mathsf{gen}) \cdot \langle \mathsf{id}, \mathsf{id} \rangle = \langle \mathsf{hd}, \mathsf{tl} \rangle \cdot \mathsf{gen} \\
= & \quad \{ \ \times \text{ absorption and fusion} \ \} \\
& \langle \mathsf{id}, \mathsf{gen} \rangle = \langle \mathsf{hd} \cdot \mathsf{gen}, \mathsf{tl} \cdot \mathsf{gen} \rangle \\
= & \quad \{ \ \text{structural equality} \ \} \\
& \mathsf{hd} \cdot \mathsf{gen} = \mathsf{id} \ \wedge \ \mathsf{tl} \cdot \mathsf{gen} = \mathsf{gen} \\
= & \quad \{ \ \text{going pointwise} \ \} \\
& \mathsf{hd}\ (\mathsf{gen}\ a) = a \ \wedge \ \mathsf{tl}\ (\mathsf{gen}\ a) = \mathsf{gen}\ a
\end{aligned}
$$

coinductive definition = behaviour under all the observers

Definition by coinduction

Merge



$$g = \langle \mathsf{hd} \cdot \pi_1, \mathsf{s} \cdot (\mathsf{tl} \times \mathsf{id}) \rangle$$

Definition by coinduction

Unfolding the diagram and going pointwise, we get an explicit definition of stream merge:

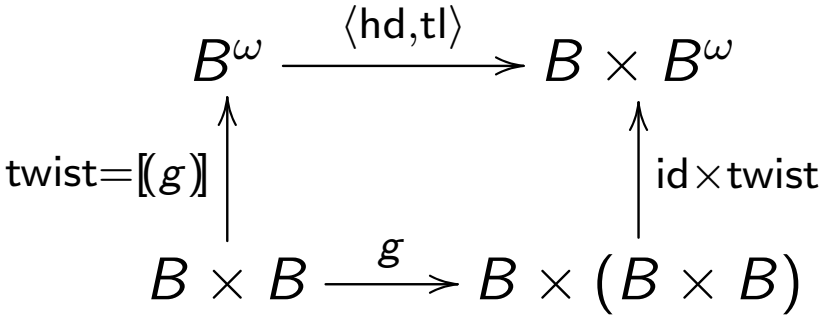
$$\begin{aligned} \mathsf{hd}\ \mathsf{merge}\,(s,t) &= \mathsf{hd}\ s \\ \mathsf{tl}\ \mathsf{merge}\,(s,t) &= \mathsf{merge}\,(t, \mathsf{tl}\ s) \end{aligned}$$

by recursive expressions

$$\mathsf{merge}\,(x : s, t) = x : \mathsf{merge}\,(t, s)$$

Definition by coinduction

Twist



$$g = \langle \pi_1, \mathsf{s} \rangle$$

Proof by coinduction

Lemma: $\text{merge}\,(a^\omega, b^\omega) \;=\; (ab)^\omega$

i.e.

$$\text{merge} \cdot (\text{gen} \times \text{gen}) \;=\; \text{twist}$$

Proof by coinduction

$$
\begin{aligned}
& \mathsf{merge} \cdot (\mathsf{gen} \times \mathsf{gen}) \;=\; \mathsf{twist} \\
= & \quad \{ \text{ merge definition } \} \\
& [\![\langle \mathsf{hd} \cdot \pi_1, \mathsf{s} \cdot (\mathsf{tl} \times \mathsf{id}) \rangle]\!] \cdot (\mathsf{gen} \times \mathsf{gen}) \;=\; [\![\langle \pi_1, \mathsf{s} \rangle]\!] \\
\Leftarrow & \quad \{ \text{ coinduction fusion } \} \\
& \langle \mathsf{hd} \cdot \pi_1, \mathsf{s} \cdot (\mathsf{tl} \times \mathsf{id}) \rangle \cdot (\mathsf{gen} \times \mathsf{gen}) \;=\; \mathsf{id} \times (\mathsf{gen} \times \mathsf{gen}) \cdot \langle \pi_1, \mathsf{s} \rangle \\
= & \quad \{ \text{ } \times \text{ absorption and reflection } \} \\
& \langle \mathsf{hd} \cdot \mathsf{gen} \cdot \pi_1, \mathsf{s} \cdot ((\mathsf{tl} \cdot \mathsf{gen}) \times \mathsf{gen}) \rangle \;=\; \mathsf{id} \times (\mathsf{gen} \times \mathsf{gen}) \cdot \langle \pi_1, \mathsf{s} \rangle \\
= & \quad \{ \text{ } \mathsf{tl} \cdot \mathsf{gen} = \mathsf{gen} \text{ and } \mathsf{hd} \cdot \mathsf{gen} = \mathsf{id} \text{ } \} \\
& \langle \pi_1, \mathsf{s} \cdot (\mathsf{gen} \times \mathsf{gen}) \rangle \;=\; \mathsf{id} \times (\mathsf{gen} \times \mathsf{gen}) \cdot \langle \pi_1, \mathsf{s} \rangle
\end{aligned}
$$

Proof by coinduction

$$\langle \pi_1, \mathsf{s} \cdot (\mathsf{gen} \times \mathsf{gen}) \rangle = \mathsf{id} \times (\mathsf{gen} \times \mathsf{gen}) \cdot \langle \pi_1, \mathsf{s} \rangle$$

$=$ { $\times$ absorption }

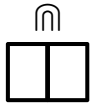
$$\langle \pi_1, \mathsf{s} \cdot (\mathsf{gen} \times \mathsf{gen}) \rangle = \langle \pi_1, (\mathsf{gen} \times \mathsf{gen}) \cdot \mathsf{s} \rangle$$

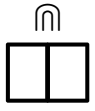
$=$ { s is natural, *i.e.*, $(f \times g) \cdot \mathsf{s} = \mathsf{s} \cdot (g \times f)$ }

$$\langle \pi_1, \mathsf{s} \cdot (\mathsf{gen} \times \mathsf{gen}) \rangle = \langle \pi_1, \mathsf{s} \cdot (\mathsf{gen} \times \mathsf{gen}) \rangle$$

- Introduction
- Motivating example: Automata
- Going generic: Coalgebras
- Application example: Transducers

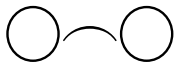
Algebras

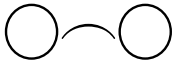
a tool box: 

an assembly process:  artifact $\xrightarrow{a}$ artifact

- algebras describe assembly processes
- and abstract data types as (initial) algebras (term algebras)
- emphasis is on construction

Coalgebras

a lens: 

an observation structure: universe $\xrightarrow{c}$  universe

- coalgebras describe observation structures (*i.e.*, transition systems)
- and abstract behaviour types as (final) coalgebras
- emphasis is on observation

Typical lens

- 'opaque'

$$U = \mathbf{1}$$

- black & white

$$U = \mathbf{2}$$

- colouring

$$U = O$$

... in each case the colour set acts as a space classifier

Typical lens

- partiality

$$\bigcirc\!\!\frown\!\!\bigcirc\, U \;=\; U + \mathbf{1}$$

- visible attributes

$$\bigcirc\!\!\frown\!\!\bigcirc\, U \;=\; O \times U$$

- external stimulus

$$\bigcirc\!\!\frown\!\!\bigcirc\, U \;=\; U^{I}$$

- non determinism

$$\bigcirc\!\!\frown\!\!\bigcirc\, U \;=\; \mathcal{P}U$$

Question

Which lens shall we seek?

- The main criteria is to choose functors for which the final coalgebra does exist
- Such is the case of the all polynomial functors as well as finite powerset functor

Coalgebras

A coalgebra for a functor T is any function from a set U (its carrier) to $\mathsf{T}U$:

$$\alpha : U \longrightarrow \mathsf{T}U$$

For any functor T, if its space of behaviours can be made a T-coalgebra itself

$$\omega_{\mathsf{T}} : \nu_{\mathsf{T}} \longrightarrow \mathsf{T}\nu_{\mathsf{T}}$$

this is the final coalgebra: from any other T-coalgebra p there is a unique morphism $[\![p]\!]$ making the following diagram to commute:

$$\begin{array}{ccc} \nu_{\mathsf{T}} & \xrightarrow{\omega_{\mathsf{T}}} & \mathsf{T}\nu_{\mathsf{T}} \\ {\scriptstyle [\![p]\!]}\uparrow & & \uparrow{\scriptstyle \mathsf{T}[\![p]\!]} \\ U & \xrightarrow{p} & \mathsf{T}U \end{array}$$

Coalgebras

This universal property is equivalently captured by the following law:

$$k = [\![p]\!] \quad \Leftrightarrow \quad \omega_{\mathsf{T}} \cdot k = \mathsf{T}\ k \cdot p$$

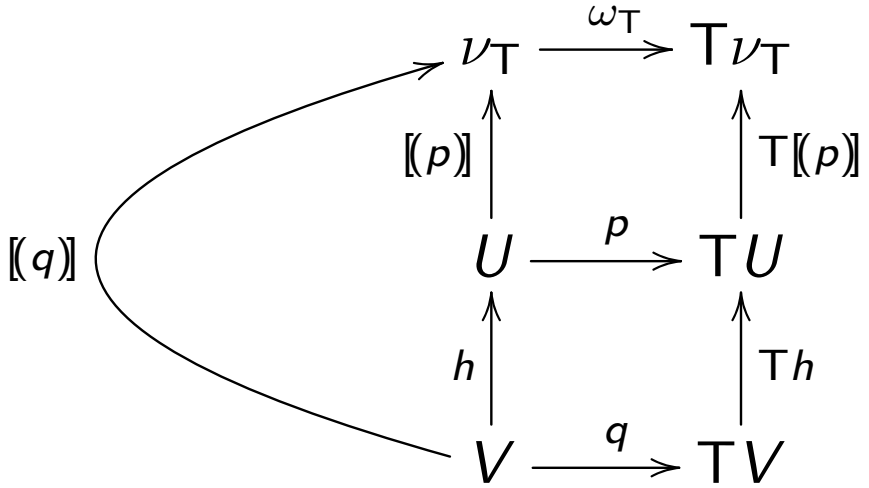
- Existence $\Leftrightarrow$ definition principle (co-recursion)
- Uniqueness $\Leftrightarrow$ proof principle (co-induction)

From which:

$$\begin{array}{rrcl} \text{cancellation} & \omega_{\mathsf{T}} \cdot [\![p]\!] & = & \mathsf{T}\ [\![p]\!] \cdot p \\ \text{reflection} & [\![\omega_{\mathsf{T}}]\!] & = & \mathrm{id}_{\nu_{\mathsf{T}}} \\ \text{fusion} & [\![p]\!] \cdot h & = & [\![q]\!] \quad \text{if} \quad p \cdot h = \mathsf{T}\ h \cdot q \end{array}$$

Coalgebras

Example: fusion law



Coalgebras

Example: fusion law

$$
\begin{array}{ll}
 & [\![p]\!] \cdot h \;=\; [\![q]\!] \\
\Leftrightarrow & \qquad \{ \text{ universal law } \} \\
 & \omega \cdot [\![p]\!] \cdot h \;=\; \mathsf{T}([\![p]\!] \cdot h) \cdot q \\
\Leftrightarrow & \qquad \{ \text{ cancellation law and } T \text{ functor } \} \\
 & \mathsf{T}[\![p]\!] \cdot p \cdot h \;=\; \mathsf{T}[\![p]\!] \cdot \mathsf{T}h \cdot q \\
\Leftarrow & \qquad \{ \text{ function equality } \} \\
 & p \cdot h \;=\; \mathsf{T}h \cdot q
\end{array}
$$

Coalgebras

From which one may generalise the fundamental result
(proved previously for automata)

Th: morphisms preserve behaviour: $[\![(q)]\!] = [\![(p)]\!] \cdot h$

Lambek's Lemma

The dynamics of the final coalgebra is an isomorphism

proof idea:

- Assume the existence of an inverse α_{T} to $\omega_{\mathsf{T}} : \nu_{\mathsf{T}} \longrightarrow \mathsf{T}\nu_{\mathsf{T}}$. Then, $\alpha_{\mathsf{T}} \cdot \omega_{\mathsf{T}} = \mathsf{id}_{\nu_{\mathsf{T}}}$ and $\omega_{\mathsf{T}} \cdot \alpha_{\mathsf{T}} = \mathsf{id}_{\mathsf{T}\nu_{\mathsf{T}}}$
- Take one of this requirements and use it to conjecture a definition for α_{T} (or an implementation ...)
  Note the use of the reflection law to introduce an anamorphism in the calculation, instead of eliminating one
- Then check the validity of this conjecture by verifying with it the other requirement

Proof by coinduction

$$\alpha_{\mathsf{T}} \cdot \omega_{\mathsf{T}} = \mathsf{id}_{\nu_{\mathsf{T}}}$$

$\Leftrightarrow$ { reflection law }

$$\alpha_{\mathsf{T}} \cdot \omega_{\mathsf{T}} = [\![(\omega_{\mathsf{T}})]\!]$$

$\Leftrightarrow$ { universal law }

$$\omega_{\mathsf{T}} \cdot \alpha_{\mathsf{T}} \cdot \omega_{\mathsf{T}} = \mathsf{T}(\alpha_{\mathsf{T}} \cdot \omega_{\mathsf{T}}) \cdot \omega_{\mathsf{T}}$$

$\Leftrightarrow$ { as a functor T preserves composition }

$$\omega_{\mathsf{T}} \cdot \alpha_{\mathsf{T}} \cdot \omega_{\mathsf{T}} = \mathsf{T}\alpha_{\mathsf{T}} \cdot \mathsf{T}\omega_{\mathsf{T}} \cdot \omega_{\mathsf{T}}$$

$\Leftrightarrow$ { cancel ω_{T} from both sides & universal law }

$$\alpha_{\mathsf{T}} = [\![(\mathsf{T}\omega_{\mathsf{T}})]\!]$$

Proof by coinduction

$$
\begin{aligned}
& \omega_T \cdot \alpha_T \\
= & \quad \{ \text{ replace } \alpha_T \text{ by the derived conjecture } \} \\
& \omega_T \cdot [\!(T\omega_T)\!] \\
= & \quad \{ \text{ } [\!(T\omega_T)\!] \text{ is a morphism } \} \\
& T[\!(T\omega_T)\!] \cdot T\omega_T \\
= & \quad \{ \text{ as a functor T preserves composition } \} \\
& T([\!(T\omega_T)\!] \cdot \omega_T) \\
= & \quad \{ \text{ just proved } \} \\
& T\ \mathrm{id}_{\nu_T} \\
= & \quad \{ \text{ as a functor T preserves identities } \} \\
& \mathrm{id}_{(T\mathrm{id}_{\nu_T})}
\end{aligned}
$$

- Introduction
- Motivating example: Automata
- Going generic: Coalgebras
- Application example: Transducers

Moore transducers

state space	U
transition function	$\overline{\mathrm{nx}} : U \longrightarrow U^A$
attribute (or label)	$\mathrm{at} : U \longrightarrow B$

i.e.,

$$p \;=\; \langle \overline{\mathrm{nx}}, \mathrm{at} \rangle : U \longrightarrow U^A \times B$$

Notation:

$$u \xrightarrow{a}_p u' \quad \Leftrightarrow \quad \overline{\mathrm{nx}}\, u\, a = u'$$

$$u \downarrow_p b \quad \Leftrightarrow \quad \mathrm{at}\, u = b$$

Moore transducers

The behaviour of p at (from) a state $u \in U$ is revealed by successive observations (experiments) triggered on input of different values $a \in A$:

$$[\![p]\!]\ u \quad = \quad [\text{at } u,\ \text{at } (\overline{\text{nx}}\ u\ a_0),\ \text{at } (\overline{\text{nx}}\ (\overline{\text{nx}}\ u\ a_0)\ a_1), \ldots]$$

$$\begin{aligned} [\![p]\!]\, u\, \underline{\text{nil}} &= \text{at } u \\ [\![p]\!]\, u\, (a : t) &= [\![p]\!]\, (\overline{\text{nx}}\ u\ a)\ t \end{aligned}$$

which means that

> Moore behaviours are elements of B^{A^*}
> (depicted as rooted trees whose branches are labelled by sequences of inputs and leaves by B values)

Moore morphisms

A morphism

$$h : p \longrightarrow q$$

where

$$\begin{aligned} p &= \langle \overline{\mathsf{nx}}, \mathsf{at} \rangle : U \longrightarrow U^A \times B \\ q &= \langle \overline{\mathsf{nx}}', \mathsf{at}' \rangle : V \longrightarrow V^A \times B \end{aligned}$$

is a function $h : U \longrightarrow V$ such that

$$\begin{array}{ccc} U & \xrightarrow{p} & U^A \times B \\ {\scriptstyle h}\downarrow & & \downarrow{\scriptstyle h^A \times \mathsf{id}} \\ V & \xrightarrow{q} & V^A \times B \end{array}$$

To avoid the explicit use of exponentials, the diagram can be decomposed into:

Moore morphisms

$$\begin{array}{ccc} U & \xrightarrow{\text{at}} & B \\ {\scriptstyle h}\downarrow & & \downarrow{\scriptstyle \text{id}} \\ V & \xrightarrow{\text{at}'} & B \end{array}$$

and

$$\begin{array}{ccc} U \times A & \xrightarrow{\text{nx}} & U \\ {\scriptstyle h\times\text{id}}\downarrow & & \downarrow{\scriptstyle h} \\ V \times A & \xrightarrow{\text{nx}'} & V \end{array}$$

corresponding to

$$\begin{aligned} \text{at}' \cdot h &= \text{at} \\ \text{nx}' \cdot (h \times \text{id}) &= h \cdot \text{nx} \end{aligned}$$

Moore morphisms

Clearly, morphisms preserve attributes and transitions

$$
\begin{array}{ll}
 & u \xrightarrow{a}_p u' \quad \text{and} \quad u \downarrow_p b \\
\Leftrightarrow & \quad \{ \text{ definition } \} \\
 & \mathsf{nx}(u, a) = u' \quad \text{and} \quad \mathsf{at}\, u = b \\
\Rightarrow & \quad \{ \text{ Liebniz } \} \\
 & h\, \mathsf{nx}(u, a) = h\, u' \quad \text{and} \quad \mathsf{at}\, u = b \\
\Leftrightarrow & \quad \{ \, h \text{ is a morphism } \} \\
 & \mathsf{nx}'(h\, u, a) = h\, u' \quad \text{and} \quad \mathsf{at}'\, h\, u = b \\
\Leftrightarrow & \quad \{ \text{ definition } \} \\
 & h\, u \xrightarrow{a}_q h\, u' \quad \text{and} \quad h\, u \downarrow_q b
\end{array}
$$

The final Moore transducer

Moore behaviours organise themselves into a final Moore machine over B^{A^*}

$$\omega = \langle \overline{\mathsf{nx}}_\omega, \mathsf{at}_\omega \rangle : B^{A^*} \longrightarrow (B^{A^*})^A \times B$$

where

$$\begin{array}{rcll} \mathsf{at}_\omega\, f & = & f\,\mathsf{nil} & \text{ie, the value before any input} \\ \overline{\mathsf{nx}}_\omega\, f\, a & = & \lambda s\,.\;\; f\,(a : s) & \text{every input determines its evolution} \end{array}$$

The final Moore transducer

Th: Coalgebra ω is the final coalgebra for $\mathsf{T}\,X = X^A \times B$

because

1. For any $p = \langle \overline{\mathsf{nx}}, \mathsf{at} \rangle$, $[\![p]\!]$ is a Moore morphism $[\![p]\!] : p \longrightarrow \omega$

$$
\begin{array}{ll}
 & \mathsf{at}_\omega \cdot [\![p]\!] \;=\; \mathsf{at} \\
\Leftrightarrow & \qquad \{ \text{ introduction of variables } \} \\
 & \mathsf{at}_\omega([\![p]\!]\, u) \;=\; \mathsf{at}\, u \\
\Leftrightarrow & \qquad \{ \text{ definition of } \mathsf{at}_\omega \ \} \\
 & ([\![p]\!]\, u)\,\mathsf{nil} \;=\; \mathsf{at}\, u \\
\Leftrightarrow & \qquad \{ \text{ definition of } [\![p]\!] \ \} \\
 & \textsc{True}
\end{array}
$$

The final Moore transducer

$$\mathsf{nx}_\omega \cdot (\llbracket p \rrbracket \times \mathsf{id}) \;=\; \llbracket p \rrbracket \cdot \mathsf{nx}$$

$\Leftrightarrow$ { introduction of variables and application }

$$\mathsf{nx}_\omega(\llbracket p \rrbracket\, u, a) \;=\; \llbracket p \rrbracket\, \mathsf{nx}\,(u, a)$$

$\Leftrightarrow$ { definition of nx_ω }

$$\lambda s\,.\;\; (\llbracket p \rrbracket\, u)\,(a : s) \;=\; \llbracket p \rrbracket\, \mathsf{nx}\,(u, a)$$

$\Leftrightarrow$ { introduction of variables and application }

$$(\llbracket p \rrbracket\, u)\,(a : t) \;=\; (\llbracket p \rrbracket\, \mathsf{nx}\,(u, a))\,t$$

$\Leftrightarrow$ { definition of $\llbracket p \rrbracket$ }

$$\textsc{True}$$

The final Moore transducer

2. ... and is unique

Exercise. Prove uniqueness (by induction on A^*)

Instances of Moore transducers

$$Queue = \langle \overline{\mathsf{nx}}, \mathsf{at} \rangle : E^* \longrightarrow (E^*)^{E+\mathbf{1}} \times ((E + \mathbf{1}) \times \mathbf{2})$$

with

$$\begin{aligned} \mathsf{at} &= \langle \mathsf{top}, \mathsf{isempty?} \rangle \\ \text{where} \quad \mathsf{top}\ s &= (s = \underline{\mathsf{nil}} \rightarrow \iota_2\ *,\ \iota_1(\mathsf{last}\ s)\ ) \\ \mathsf{isempty?}\ s &= s = \underline{\mathsf{nil}} \end{aligned}$$

$$\begin{aligned} \mathsf{nx} &= [\mathsf{enq}, \mathsf{deq}] \cdot \mathsf{dl} \\ \text{where} \quad \mathsf{enq}\ (s, e) &= e : s \\ \mathsf{deq}\ (s, *) &= (s = \underline{\mathsf{nil}} \rightarrow s,\ (\mathsf{blast}\ s)) \end{aligned}$$

Instances of Moore transducers

Make $B = \mathbf{2}$ in $\mathsf{T}\,X = X^A \times B$.
The carrier (or state space) of the corresponding final coalgebra is

$$\mathbf{2}^{A^*} \;\cong\; \mathcal{P}A^*$$

and its dynamics is $\langle \overline{\mathsf{nx}}_\omega, \mathsf{at}_\omega \rangle : \mathcal{P}A^* \longrightarrow (\mathcal{P}A^*)^A \times \mathbf{2}$
where

$$\begin{array}{lcl} \mathsf{at}_\omega\, L & = & \underline{\mathsf{nil}} \in L \\ \overline{\mathsf{nx}}_\omega\, L & = & \lambda\, a\,.\ \{(a : s) |\ s \in L\} \end{array}$$

Exercise. ... what are we talking about?
Exercise. Make $A = \mathbf{1}$ in $\mathsf{T}\,X = X^A \times B$. What comes up?

Mealy transducers

state space	U
reactive transition function	$\overline{\mathsf{ac}} : U \longrightarrow (U \times B)^A$

Notation:

$$u \xrightarrow{a/b}_p u' \quad \Leftrightarrow \quad \overline{\mathsf{ac}}\, u\, a = (u', b)$$

Mealy transducers

The behaviour of p at a state $u \in U$ is revealed by successive observations (experiments) triggered on input of different values $a \in A$:

$$[\![(p)]\!]\ u \quad = \quad [\pi_2(\overline{\mathsf{ac}}\ u\ a_0), \pi_2(\overline{\mathsf{ac}}\ (\pi_1(\overline{\mathsf{ac}}\ u\ a_0))\ a_1, ...]$$

$$\begin{aligned} [\![(p)]\!]u\,[a] &= \pi_2(\mathsf{ac}\ u\ a) \\ [\![(p)]\!]u\,(a : t) &= [\![(p)]\!]\,(\pi_1(\mathsf{ac}\ u\ a))\ t \end{aligned}$$

which means that

> Mealy behaviours are elements of B^{A^+}

Mealy transducers

Mealy behaviours can alternatively be regarded as

causal functions from A^ω to B^ω

A causal function f over streams is such that, for all $s, t \in A^\omega$ and $n \in \mathbb{N}$,

$$\langle \forall\ k\ :\ k \leq n :\ s\,k\ =\ t\,k \rangle\ \Rightarrow\ (f\,s\,n\ =\ f\,t\,n)$$

i.e, the n-th element of $f\,s$ depends only on the first n elements of input stream s

... upon which the final Mealy automata can be defined:

The final Mealy transducer

Mealy behaviours organise themselves into a final Mealy automata over $\Gamma = \{f : A^\omega \longrightarrow B^\omega \mid f \text{ is causal}\}$

$$\overline{\omega} : \Gamma \longrightarrow (\Gamma \times B)^A$$

where

$$\overline{\omega}\, f\, a \quad = \quad \langle \lambda\, s\, .\, \mathrm{tl}\ f(a : s), \mathrm{hd}\, f(a : r) \rangle$$

which means that

- the next state acts as f after a has been seen
- the output $\mathrm{hd}\, f(a : r)$ depends only on f and a; therefore, the tail r of the input stream is irrelevant.

Non-determinism

Further behavioural effects can be introduced in the basic machines discussed so far by 'sophisticating' the corresponding signature functor. For example,

- non-determinism is captured by the powerset functor $\mathcal{P}$

Automata	$\mathsf{T}X = B \times X$	$\mathsf{T}X = \mathcal{P}(B \times X)$
Moore transducer	$\mathsf{T}X = X^A \times B$	$\mathsf{T}X = \mathcal{P}(X)^A \times B$
Mealy transducer	$\mathsf{T}X = (X \times B)^A$	$\mathsf{T}X = \mathcal{P}(X \times B)^A$