Coalgebras, the Hennessy-Milner property, and the Adjoint Functor Theorem

Raul Andres Leal[1]

[1]ILLC
Universiteit van Amsterdam

7-10-2008
University of Leicester

Outline

1. Coalgebras: Structures
   - Behavioral Equivalence
   - Final Coalgebras
2. Coalgebras: Languages
3. Structures Vs Languages
4. Pointless Languages

Outline

1. Coalgebras: Structures
    - Behavioral Equivalence
    - Final Coalgebras
2. Coalgebras: Languages
3. Structures Vs Languages
4. Pointless Languages

Coalgebras: Intuition

- Coalgebra = Dual of Algebra.
- Observation Vs Construction.
- Coalgebra = Machines from the point of view of the user.

Example: Battery Chargers

Battery chargers are coalgebraic structures (One button machines).

The are represented by a function

$$\alpha : A \longrightarrow 1 + A$$

Example: Battery Chargers

Battery chargers are coalgebraic structures (One button machines).
The are represented by a function

$$\alpha : A \longrightarrow 1 + A$$

Examples

- One button machines with screen (deterministic transition systems)

$$\alpha : A \longrightarrow L \times A$$

- Kripke frames (non-deterministic transition systems)

$$\alpha : A \longrightarrow \mathcal{P}A$$

- Kripke Models

$$\alpha : A \longrightarrow \mathcal{P}(Q) \times \mathcal{P}(A)$$

- Non-deterministic label transition systems

$$\alpha : A \longrightarrow \mathcal{P}(L \times A)$$

Coalgebraic Structures

Definition

A coalgebra for a functor $T : Set \longrightarrow Set$ is a function

$$\alpha : A \longrightarrow TA$$

Question:

How do we relate coalgebraic structures?

Coalgebraic Structures

Definition

A coalgebra for a functor $T : Set \longrightarrow Set$ is a function

$$\alpha : A \longrightarrow TA$$

Question:

How do we relate coalgebraic structures?

Examples:

Hard Situation

We want to relate two systems

$$\alpha : A \longrightarrow \mathcal{P}(A) \text{ and } \beta : B \longrightarrow \mathcal{P}(B)$$

Easy Situation:

We want to relate two machines

$$\alpha : A \longrightarrow 1 + A \text{ and } \beta : B \longrightarrow 1 + B$$

Examples:

Hard Situation

We want to relate two systems

$$\alpha : A \longrightarrow \mathcal{P}(A) \text{ and } \beta : B \longrightarrow \mathcal{P}(B)$$

Easy Situation:

We want to relate two machines

$$\alpha : A \longrightarrow 1 + A \text{ and } \beta : B \longrightarrow 1 + B$$

Solving the easy situation:

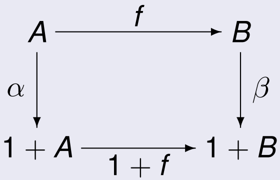
Easy Situation:

To relate two machines $\alpha : A \longrightarrow 1 + A$ and $\beta : B \longrightarrow 1 + B$

- The halting states should be related.
- Related states should have the same “charge”

Solution

The following diagram



commutes.

Coalgebraic Morphisms

Definition

A coalgebraic morphism from α to β, written $f : \alpha \longrightarrow \beta$, is a function $f : A \longrightarrow B$ such that the following diagram

$$\begin{array}{ccc} A & \xrightarrow{f} & B \\ {\scriptstyle\alpha}\downarrow & & \downarrow{\scriptstyle\beta} \\ TA & \xrightarrow[T(f)]{} & TB \end{array}$$

commutes

Solving the hard situation

Hard Situation:

To relate two machines $\alpha : A \longrightarrow \mathcal{P}(A)$ and $\beta : B \longrightarrow \mathcal{P}(B)$

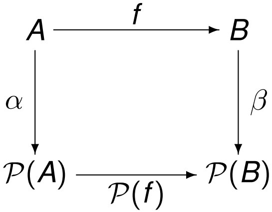
Solution

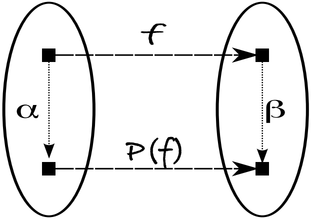
The following diagram

$$\begin{array}{ccc} A & \xrightarrow{\ f\ } & B \\ \downarrow{\scriptstyle \alpha} & & \downarrow{\scriptstyle \beta} \\ \mathcal{P}(A) & \xrightarrow[\mathcal{P}(f)]{} & \mathcal{P}(B) \end{array}$$

commutes.

Reading the Solution





Another Example

Deterministic Label Transition Systems

$$
\begin{array}{ccc}
A & \xrightarrow{\quad f \quad} & B \\
\alpha \downarrow & & \downarrow \beta \\
L \times A & \xrightarrow[(1_L, f)]{} & L \times B
\end{array}
$$

Related states should have the same labels.

Remark

The states s and $f(s)$ always have the same behavior!!

Another Example

Deterministic Label Transition Systems

$$\begin{array}{ccc} A & \xrightarrow{f} & B \\ \downarrow{\scriptstyle \alpha} & & \downarrow{\scriptstyle \beta} \\ L \times A & \xrightarrow[(1_L, f)]{} & L \times B \end{array}$$

Related states should have the same labels.

Remark

The states s and $f(s)$ always have the same behavior!!

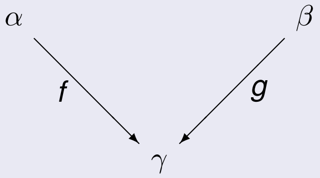
Outline

1. Coalgebras: Structures
   - Behavioral Equivalence
   - Final Coalgebras
2. Coalgebras: Languages
3. Structures Vs Languages
4. Pointless Languages

Behavioral Equivalence of States

Definition

Two states $s \in \alpha$ and $s' \in \beta$ are *behavioral equivalent, written* $s \sim s'$, iff there exists a coalgebra γ and morphisms



such that $f(s) = g(s')$.

The Behavior of a State

Behavior

The behavior of a state is the “evolution” of the state.

Under appropriate circumstances we can give a concrete representation to the observable behavior

The observable behavior of one button machines

A state s can...

- lead to the halt of the machine, or
- lead us to one step closer to the halt of the machine.
- Keep us waiting, i.e. we will never see the machine stop.

A concrete presentation

Consider the set

$$\overline{\mathbb{N}} = \mathbb{N} \cup \infty$$

and the function $\zeta : \overline{\mathbb{N}} \longrightarrow 1 + \overline{\mathbb{N}}$ defined as follows

$$\zeta(0) = *; \quad \zeta(n+1) = n; \quad \zeta(\infty) = \infty$$

The observable behavior of one button machines

A state s can...

- lead to the halt of the machine, or
- lead us to one step closer to the halt of the machine.
- Keep us waiting, i.e. we will never see the machine stop.

A concrete presentation

Consider the set

$$\overline{\mathbb{N}} = \mathbb{N} \cup \infty$$

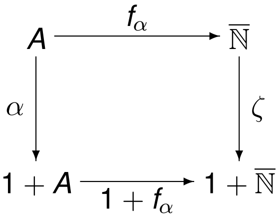
and the function $\zeta : \overline{\mathbb{N}} \longrightarrow 1 + \overline{\mathbb{N}}$ defined as follows

$$\zeta(0) = *; \quad \zeta(n+1) = n; \quad \zeta(\infty) = \infty$$

Why is this cool?

Because..

Given a machine $\alpha : A \longrightarrow 1 + A$ we can define a unique morphism $f_\alpha : \alpha \longrightarrow \zeta$ as follows



$$f_\alpha(a) = \begin{cases} 0 \text{ if } \alpha(a) = * \\ n \text{ if } f_\alpha \alpha(a) = n + 1 \\ \infty \text{ if } f_\alpha \alpha(a) = \infty \end{cases}$$

This is coinduction!!!

Outline

Final Coalgebras

Definition

A final T-coalgebra (Z, ζ) is a terminal object in the category of T-coalgebras, i.e. for every T-coalgebra α there exists a unique morphism

$$f_\alpha : \alpha \longrightarrow \zeta.$$

Examples

- Deterministic transition systems: A final coalgebra is the set of infinite lists over L.

$$\zeta : L^{\mathbb{N}} \longrightarrow L \times L^{\mathbb{N}}$$

- Kripke frames, and Kripke models have no final coalgebra.

Nice properties of final coalgebras

Theorem

If a final coalgebra exists, two states $s \in \alpha$ and $s' \in \beta$ are behavioral equivalent iff they are mapped to the same state in a final coalgebra, i.e.

$$s \sim s' \text{ iff } f_\alpha(s) = f_\beta(s')$$

Important

Final coalgebras code behavioral equivalence semantically.

Nice properties of final coalgebras

Theorem

If a final coalgebra exists, two states $s \in \alpha$ and $s' \in \beta$ are behavioral equivalent iff they are mapped to the same state in a final coalgebra, i.e.

$$s \sim s' \text{ iff } f_\alpha(s) = f_\beta(s')$$

Important

Final coalgebras code behavioral equivalence semantically.

Outline

Abstract coalgebraic languages

Definition

An *abstract coalgebraic language* is a set $\mathcal{L}$ together with a function

$$Th_\alpha : A \longrightarrow \mathcal{P}\mathcal{L}$$

for each coalgebra $\alpha : A \longrightarrow TA$.

Example: Modal Logic for Kripke structures

- We use have basic propositional logic.
- We describe the behavior of a state using two modalities $\Box$, and $\Diamond$. Given a Kripke frame $\alpha : A \longrightarrow \mathcal{P}(A)$

$$a \models \Box\varphi \text{ iff } \alpha(a) \subseteq [\![\varphi]\!]$$

Important fact

If two states are behavioral equivalent, they satisfy the same formulas.

What do we want coalgebraic languages?

- We want to generalize modal logic.
- We want to describe the behavior of a system.
- We want to provide an internal local perspective of dynamic systems.

Expressive languages

Definition

An abstract coalgebraic language is *expressive* iff it completely describes behavioral equivalence, i.e.

$$s \sim s' \text{ iff } Th_\alpha(s) = Th_\beta(s').$$

Important

Expressive languages code behavioral equivalence syntactically

Expressive languages

Definition

An abstract coalgebraic language is *expressive* iff it completely describes behavioral equivalence, i.e.

$$s \sim s' \text{ iff } Th_{\alpha}(s) = Th_{\beta}(s').$$

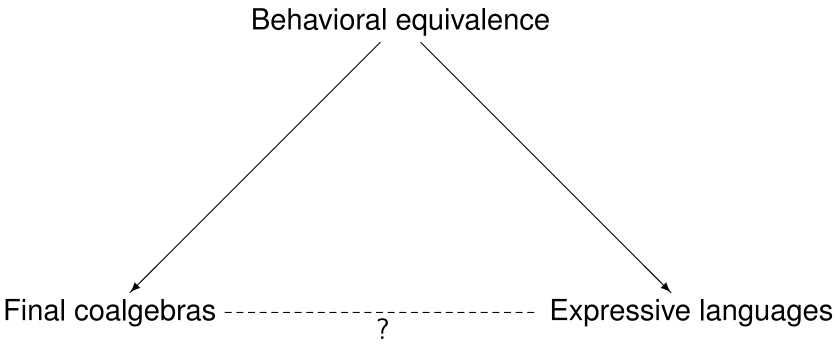
Important

Expressive languages code behavioral equivalence syntactically

Outline

Summary

$$f_\alpha(s) = f_\beta(s') \text{ iff } s \sim s' \text{ iff } Th_\alpha(s) = Th_\beta(s')$$



Our main topic

Theorem (Goldblatt)

For every functor $T : Set \longrightarrow Set$, the existence of a final T-coalgebra is equivalent to the existence of an expressive language with respect to behavioral equivalence.

From final coalgebras to expressive languages

Theorem

If there exists a final coalgebra ζ, there exists an expressive abstract coalgebraic language.

Proof.

Take $\mathcal{L} = Z$ and $Th_\alpha = f_\alpha$. □

From expressive languages to final coalgebras

Theorem

If there exists an expressive language, there exists a final coalgebra.

A point wise definition of final coalgebras

Proof.

1. Take $Z = \{\Phi \subseteq \mathcal{L} \mid (\exists \alpha)(\exists s \in \alpha)(Th_\alpha(s) = \Phi)\}$.
2. Define $\zeta : Z \longrightarrow TZ$ as follows: an element $Th_\alpha(s) = \Phi \in \mathcal{PL}$ is mapped to

$$\zeta(\Phi) = T(Th_\alpha)\alpha(s).$$

3. Prove that ζ is well defined.
4. Prove that $Th_\alpha : \alpha \longrightarrow \zeta$ is the only morphism of coalgebras.

□

A point wise definition of final coalgebras

Proof.

1. Take $Z = \{\Phi \subseteq \mathcal{L} \mid (\exists\alpha)(\exists s \in \alpha)(Th_\alpha(s) = \Phi)\}$.

Those are the states of a final coalgebra

2. Define $\zeta : Z \longrightarrow TZ$ as follows: an element $Th_\alpha(s) = \Phi \in \mathcal{PL}$ is mapped to

$$\zeta(\Phi) = T(Th_\alpha)\alpha(s).$$

3. Prove that ζ is well defined.
4. Prove that $Th_\alpha : \alpha \longrightarrow \zeta$ is the only morphism of coalgebras.

□

A point wise definition of final coalgebras

Proof.

1. Take $Z = \{\Phi \subseteq \mathcal{L} \mid (\exists\alpha)(\exists s \in \alpha)(Th_\alpha(s) = \Phi)\}$.
2. Define $\zeta : Z \longrightarrow TZ$ as follows:

$$\begin{array}{ccc} A & \xrightarrow{Th_\alpha} & Z \\ {\scriptstyle\alpha}\downarrow & & \downarrow{\scriptstyle\zeta} \\ TA & \xrightarrow[T(Th_\alpha)]{} & TZ \end{array}$$

3. Prove that ζ is well defined.
4. Prove that $Th_\alpha : \alpha \longrightarrow \zeta$ is the only morphism of coalgebras.

□

A point wise definition of final coalgebras

Proof.

1. Take $Z = \{\Phi \subseteq \mathcal{L} \mid (\exists\alpha)(\exists s \in \alpha)(Th_\alpha(s) = \Phi)\}$.
2. Define $\zeta : Z \longrightarrow TZ$ as follows: an element $Th_\alpha(s) = \Phi \in \mathcal{PL}$ is mapped to

$$\zeta(\Phi) = T(Th_\alpha)\alpha(s).$$

3. Prove that ζ is well defined.
4. Prove that $Th_\alpha : \alpha \longrightarrow \zeta$ is the only morphism of coalgebras.

□

A point wise definition of final coalgebras

Proof.

1. Take $Z = \{\Phi \subseteq \mathcal{L} \mid (\exists \alpha)(\exists s \in \alpha)(Th_\alpha(s) = \Phi)\}$.
2. Define $\zeta : Z \longrightarrow TZ$ as follows: an element $Th_\alpha(s) = \Phi \in \mathcal{P}\mathcal{L}$ is mapped to
$$\zeta(\Phi) = T(Th_\alpha)\alpha(s).$$
3. Prove that ζ is well defined.
4. Prove that $Th_\alpha : \alpha \longrightarrow \zeta$ is the only morphism of coalgebras.

□

A point wise definition of final coalgebras. We are here!!!

Proof.

1. Take $Z = \{\Phi \subseteq \mathcal{L} \mid (\exists \alpha)(\exists s \in \alpha)(Th_\alpha(s) = \Phi)\}$.
2. Define $\zeta : Z \longrightarrow TZ$ as follows: an element $Th_\alpha(s) = \Phi \in \mathcal{PL}$ is mapped to

$$\zeta(\Phi) = T(Th_\alpha)\alpha(s).$$

3. Prove that ζ is well defined.
4. Prove that $Th_\alpha : \alpha \longrightarrow \zeta$ is the only morphism of coalgebras.

□

The structural map ζ is well defined

Proof.

1. For every morphism $f : \alpha \longrightarrow \beta$ and every state $s \in \alpha$, the equation
$$T(Th_\alpha)\alpha(s) = T(Th_\beta)\beta f(s)$$
holds.
2. For every pair of states $s \in \alpha$ and $s' \in \beta$. If $Th_\alpha(s) = Th_\beta$ then
$$T(Th_\alpha)\alpha(s) = T(Th_\beta)\beta(s')$$

Important

You have to use that the language $\mathcal{L}$ is expressive

□

The structural map ζ is well defined

Proof.

1. For every morphism $f : \alpha \longrightarrow \beta$ and every state $s \in \alpha$, the equation
$$T(Th_\alpha)\alpha(s) = T(Th_\beta)\beta f(s)$$
holds.

You will use that $s \sim s'$ implies $Th_\alpha(s) = Th_\beta(s')$

2. For every pair of states $s \in \alpha$ and $s' \in \beta$. If $Th_\alpha(s) = Th_\beta$ then
$$T(Th_\alpha)\alpha(s) = T(Th_\beta)\beta(s')$$

Important

You have to use that the language $\mathcal{L}$ is expressive

The structural map ζ is well defined

Proof.

1. For every morphism $f : \alpha \longrightarrow \beta$ and every state $s \in \alpha$, the equation
$$T(Th_\alpha)\alpha(s) = T(Th_\beta)\beta f(s)$$
holds.
2. For every pair of states $s \in \alpha$ and $s' \in \beta$. If $Th_\alpha(s) = Th_\beta$ then
$$T(Th_\alpha)\alpha(s) = T(Th_\beta)\beta(s')$$

You will use that $Th_\alpha(s) = Th_\beta(s')$ implies $s \sim s'$

Important

You have to use that the language $\mathcal{L}$ is expressive

The structural map ζ is well defined

Proof.

1. For every morphism $f : \alpha \longrightarrow \beta$ and every state $s \in \alpha$, the equation
$$T(Th_\alpha)\alpha(s) = T(Th_\beta)\beta f(s)$$
holds.
2. For every pair of states $s \in \alpha$ and $s' \in \beta$. If $Th_\alpha(s) = Th_\beta$ then
$$T(Th_\alpha)\alpha(s) = T(Th_\beta)\beta(s')$$

Important

You have to use that the language $\mathcal{L}$ is expressive

□

A point wise definition of final coalgebras. We are here!!!

Proof.

1. Take $Z = \{\Phi \subseteq \mathcal{L} \mid (\exists\alpha)(\exists s \in \alpha)(Th_\alpha(s) = \Phi)\}$.
2. Define $\zeta : Z \longrightarrow TZ$ as follows: an element $Th_\alpha(s) = \Phi \in \mathcal{PL}$ is mapped to

$$\zeta(\Phi) = T(Th_\alpha)\alpha(s).$$

3. Prove that ζ is well defined.
4. Prove that $Th_\alpha : \alpha \longrightarrow \zeta$ is the only morphism of coalgebras.

□

The function Th_α is the only morphism

Proof.

1. Assume there exists a morphism $f : \alpha \longrightarrow \zeta$ and $s \in \alpha$ such that
$$f(s) \neq Th_\alpha(s).$$
2. Then there exists a coalgebra β and $s' \in \beta$ such that $Th_\beta(s') = f(s)$.
3. This implies $s \sim s'$. Since the language is expressive we conclude
$$Th_\beta(s') = Th_\alpha(s).$$
4. A contradiction.

□

The function Th_α is the only morphism

Proof.

1. Assume there exists a morphism $f : \alpha \longrightarrow \zeta$ and $s \in \alpha$ such that
$$f(s) \neq Th_\alpha(s).$$
2. Then there exists a coalgebra β and $s' \in \beta$ such that $Th_\beta(s') = f(s)$.
3. This implies $s \sim s'$. Since the language is expressive we conclude
$$Th_\beta(s') = Th_\alpha(s).$$
4. A contradiction.

□

The function Th_α is the only morphism

Proof.

1. Assume there exists a morphism $f : \alpha \longrightarrow \zeta$ and $s \in \alpha$ such that
$$f(s) \neq Th_\alpha(s).$$
2. Then there exists a coalgebra β and $s' \in \beta$ such that $Th_\beta(s') = f(s)$.
3. This implies $s \sim s'$. Since the language is expressive we conclude
$$Th_\beta(s') = Th_\alpha(s).$$
4. A contradiction.

□

The function Th_α is the only morphism

Proof.

1. Assume there exists a morphism $f : \alpha \longrightarrow \zeta$ and $s \in \alpha$ such that
$$f(s) \neq Th_\alpha(s).$$
2. Then there exists a coalgebra β and $s' \in \beta$ such that $Th_\beta(s') = f(s)$.
3. This implies $s \sim s'$. Since the language is expressive we conclude
$$Th_\beta(s') = Th_\alpha(s).$$
4. A contradiction.

□

The function Th_α is the only morphism

Proof.

1. Assume there exists a morphism $f : \alpha \longrightarrow \zeta$ and $s \in \alpha$ such that
$$f(s) \neq Th_\alpha(s).$$
2. Then there exists a coalgebra β and $s' \in \beta$ such that $Th_\beta(s') = f(s)$.
3. This implies $s \sim s'$. Since the language is expressive we conclude
$$Th_\beta(s') = Th_\alpha(s).$$
4. A contradiction.

□

Moreover....

Theorem

An abstract coalgebraic language $\mathcal{L}$ is expressive iff the set

$$Z = \{\Phi \subseteq \mathcal{L} \mid (\exists \alpha)(\exists s \in \alpha)(Th_\alpha(s) = \Phi)\}$$

admits a final coalgebraic structure ζ (for T) such that the arrow Th_α is the only morphism.

Corollary

An abstract coalgebraic language $\mathcal{L}$ has the Henessy-Milner iff the set

$$Z = \{\Phi \subseteq \mathcal{L} \mid (\exists \alpha)(\exists s \in \alpha)(Th_\alpha(s) = \Phi)\}$$

admits a coalgebraic structure ζ (for T) such that the arrow Th_α is a morphism.

Moreover....

Theorem

An abstract coalgebraic language $\mathcal{L}$ is expressive iff the set

$$Z = \{\Phi \subseteq \mathcal{L} \,|\, (\exists \alpha)(\exists s \in \alpha)(Th_\alpha(s) = \Phi)\}$$

admits a final coalgebraic structure ζ (for T) such that the arrow Th_α is the only morphism.

Corollary

An abstract coalgebraic language $\mathcal{L}$ has the Henessy-Milner iff the set

$$Z = \{\Phi \subseteq \mathcal{L} \,|\, (\exists \alpha)(\exists s \in \alpha)(Th_\alpha(s) = \Phi)\}$$

admits a coalgebraic structure ζ (for T) such that the arrow Th_α is a morphism.

Some extra properties

- The theory map $Th_\zeta : Z \longrightarrow \mathcal{PL}$ is the inclusion.
- **Truth Lemma:** For any formula $\varphi \in \mathcal{L}$ and any set $\Phi \in Z$

$$\varphi \in Th_\zeta(\Phi) \text{ iff } \varphi \in \Phi.$$

Outline

Farewell to *Set*

Our aim

To construct final coalgebras over categories different than *Set*

First issue

What is an expressive language outside *Set*?

Pointless languages I

Definition

An *abstract coalgebraic language* is a set $\mathcal{L}$ together with a function

$$Th_\alpha : A \longrightarrow \mathcal{PL}$$

for each coalgebra $\alpha : A \longrightarrow TA$.

- In our construction we are not using the points (formulas) in $\mathcal{L}$.
- In the “real live” $\mathcal{L}$ has an algebraic structure and...
- in the boolean case, our theory maps are functions

$$Th_\alpha : A \longrightarrow Uf(\mathcal{L}).$$

Pointless languages II

Definition

Given a functor $T : \mathbb{A} \longrightarrow \mathbb{A}$, an *abstract coalgebraic language* for T-coalgebras is an object $\mathcal{L}$ together with a morphism

$$Th_\alpha : A \longrightarrow \mathcal{L}$$

for each coalgebra $\alpha : A \longrightarrow TA$.

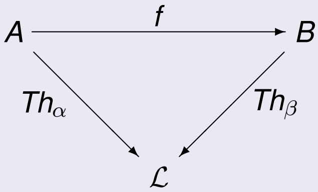
Pointless expressivity I

Expressivity in *Set*

$$s \sim s' \text{ iff } Th_\alpha(s) = Th_\beta(s')$$

From left to right

The following diagram



commutes for every coalgebra morphism f.

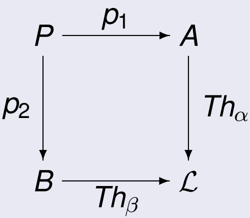
Pointless expressivity II

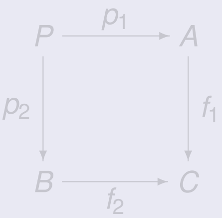
Expressivity in *Set*

$$s \sim s' \text{ iff } Th_\alpha(s) = Th_\beta(s')$$

One reading from right to left

For every pullback there exists a pair of coalgebra morphism f_1, f_2 such that





the diagram on the right commutes.

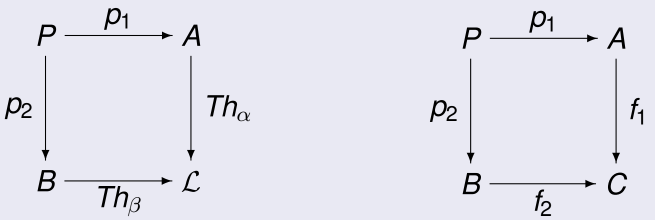
Pointless expressivity II

Expressivity in *Set*

$$s \sim s' \text{ iff } Th_\alpha(s) = Th_\beta(s')$$

One reading from right to left

For every pullback there exists a pair of coalgebra morphism f_1, f_2 such that



the diagram on the right commutes.

From final coalgebras to expressive languages

Theorem

For any functor $T : \mathbb{A} \longrightarrow \mathbb{A}$ over a category with pullbacks; if there exists a final coalgebra ζ, there exists an expressive abstract coalgebraic language.

Proof.

Take $\mathcal{L} = Z$ and $Th_\alpha = f_\alpha$. □

One road to go

The converse of the previous theorem holds if $\mathbb{A}$ is monadic over *Set*

Nostalgia for *Set*

In *Set* the following are equivalent: For an adequate language $\mathcal{L} \ldots$

- $\mathcal{L}$ is expressive.
- The function ζ is well defined.
- The set

  $$Z = \{\Phi \subset \mathcal{L} \mid (\exists \alpha)(\exists a \in \alpha)(Th_{\alpha}(a) = \Phi)\}$$

  admits a coalgebraic structure such that for each coalgebra α the function Th_{α} is a morphism of coalgebras.
- The condition with pullbacks ...
- But there is more...

Nostalgia for *Set*

In *Set* the following are equivalent: For an adequate language $\mathcal{L}$ ...

- $\mathcal{L}$ is expressive.
- The function ζ is well defined.
- The set

  $$Z = \{\Phi \subset \mathcal{L} \mid (\exists \alpha)(\exists a \in \alpha)(Th_\alpha(a) = \Phi)\}$$

  admits a coalgebraic structure such that for each coalgebra α the function Th_α is a morphism of coalgebras.
- The condition with pullbacks ...
- But there is more...

The blind Set theorist

In *Set* the following are equivalent: For an adequate language $\mathcal{L}$ . . .

- The set

$$Z = \{\Phi \subset \mathcal{L} \mid (\exists \alpha)(\exists a \in \alpha)(Th_\alpha(a) = \Phi)\}$$

  admits a coalgebraic structure (for T) such that for each coalgebra α the function Th_α is a morphism of coalgebras.

- For each coalgebra α the set

$$Z_\alpha = \{\Phi \subseteq \mathcal{L} \mid (\exists a \in \alpha)(Th_\alpha(a) = \Phi)\}$$

  admits a coalgebraic structure for T such that the function Th_α is a morphism of coalgebras.

- For each coalgebra (A, α) we can make the quotient with $Ker(Th_\alpha)$ in $Coalg(T)$.

The blind Set theorist

In *Set* the following are equivalent: For an adequate language $\mathcal{L}$ . . .

- The set

$$Z = \{\Phi \subset \mathcal{L} \,|\, (\exists \alpha)(\exists a \in \alpha)(Th_\alpha(a) = \Phi)\}$$

  admits a coalgebraic structure (for T) such that for each coalgebra α the function Th_α is a morphism of coalgebras.
- For each coalgebra α the set

$$Z_\alpha = \{\Phi \subseteq \mathcal{L} \,|\, (\exists a \in \alpha)(Th_\alpha(a) = \Phi)\}$$

  admits a coalgebraic structure for T such that the function Th_α is a morphism of coalgebras.
- For each coalgebra (A, α) we can make the quotient with $Ker(Th_\alpha)$ in $Coalg(T)$.

The blind Set theorist

In *Set* the following are equivalent: For an adequate language $\mathcal{L}$...

- The set

$$Z = \{\Phi \subset \mathcal{L} \mid (\exists \alpha)(\exists a \in \alpha)(Th_\alpha(a) = \Phi)\}$$

  admits a coalgebraic structure (for T) such that for each coalgebra α the function Th_α is a morphism of coalgebras.
- For each coalgebra α the set

$$Z_\alpha = \{\Phi \subseteq \mathcal{L} \mid (\exists a \in \alpha)(Th_\alpha(a) = \Phi)\}$$

  admits a coalgebraic structure for T such that the function Th_α is a morphism of coalgebras.
- For each coalgebra (A, α) we can make the quotient with $Ker(Th_\alpha)$ in $Coalg(T)$.

Unraveling the quotient

The quotient

For each coalgebra α

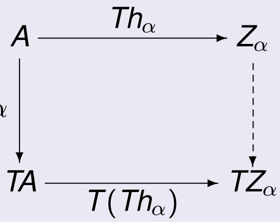
$$\begin{array}{ccc} A & \xrightarrow{\;Th_\alpha\;} & Z_\alpha \\ {\scriptstyle\alpha}\big\downarrow & & \\ TA & \xrightarrow[\;T(Th_\alpha)\;]{} & TZ_\alpha \end{array}$$

we can fill this diagram.

Unraveling the quotient

The quotient

For each coalgebra α



we can fill this diagram.

The categorical Point of View

Two facts:

- We are using a factorization structure.
- We can use adjoints.

The Adjoint Functor Theorem

Theorem

If $\mathbb{C}$ is a cocomplete category, then $\mathbb{C}$ has a terminal object if and only if it has a set S of objects which is weakly final, i.e. For every $c \in \mathbb{C}$ there exists an arrow $c \longrightarrow s$.

Corollary

For any functor $T : \mathbb{A} \longrightarrow \mathbb{A}$ over a decent category with factorization structures the existence of an expressive object implies the existence of a final coalgebra.

The End.