Comodules over relative comonads for streams and infinite matrices

Benedikt Ahrens

joint work with Régis Spadotti

Institut de Recherche en Informatique de Toulouse
Université Paul Sabatier

CMCS 2014

Goal

- Category-theoretic semantics of **co**inductive data types in intensional Martin-Löf type theory (IMLTT)

$\rightsquigarrow$ Develop a notion of "coalgebra" for the signature of a codata type

- Incorporate canonical cosubstitution

Outline

1. Syntax: inductives and substitution
2. Homogeneous cosyntax: streams
3. Heterogeneous cosyntax: infinite triangular matrices

Outline

1. Syntax: inductives and substitution
2. Homogeneous cosyntax: streams
3. Heterogeneous cosyntax: infinite triangular matrices

Heterogeneous data types

Motivation: **binding syntax** in MLTT

$$\mathsf{Lc} : \mathsf{Type} \to \mathsf{Type}$$

$$\frac{x : X}{\mathsf{var}(x) : \mathsf{Lc}(X)} \qquad \frac{s, t : \mathsf{Lc}X}{\mathsf{app}(s, t) : \mathsf{Lc}X} \qquad \frac{t : \mathsf{Lc}(X + 1)}{\mathsf{abs}(t) : \mathsf{Lc}X}$$

Heterogeneity of abs:

recursive argument with bigger parameter $X + 1$

Substitution:

$$\mathsf{subst}_{X,Y} : \big(X \to \mathsf{Lc}Y\big) \to \mathsf{Lc}X \to \mathsf{Lc}Y$$

Avoiding capture:

$$\mathsf{shift}_{X,Y} : \big(X \to \mathsf{Lc}Y\big) \to X + 1 \to \mathsf{Lc}(Y + 1)$$

Initial semantics for binding syntax

Initial semantics for lambda calculus: Fiore, Plotkin & Turi '99

- characterizes not only data type but also **substitution**
- reformulated using **monads** by Hirschowitz & Maggesi '07

Basis for this reformulation:

Lemma (Substitution is monadic: Altenkirch & Reus '99)

$(\mathsf{Lc}, \mathsf{var}, \mathsf{subst})$ *forms a monad (in Kleisli form)*

Initial semantics for λ-calculus using monads

Definition (Algebra for signature of Lc, H & M '07)

- a monad $(T, \mathsf{unit}, \mathsf{bind})$ on Type
- two **morphisms of modules over** T,

$$App : T \times T \to T$$
$$Abs : T(\_ + 1) \to T$$

Initial semantics for λ-calculus using monads

Definition (Algebra for signature of Lc, H & M '07)

- a monad $(T, \mathsf{unit}, \mathsf{bind})$ on Type
- two **morphisms of modules over** T,

$$App : T \times T \to T$$
$$Abs : T(\_ + 1) \to T$$

"Module morphism" expresses **commutativity with** bind:

$$\begin{aligned} \mathsf{bind}\ f \circ App &= App \circ (\mathsf{bind}\ f)^2 \\ \mathsf{bind}\ f \circ Abs &= Abs \circ \mathsf{bind}\ (\mathsf{shift}\ f) \end{aligned}$$

Initial semantics for λ-calculus using monads

Definition (Algebra for signature of Lc, H & M '07)

- a monad $(T, \mathsf{unit}, \mathsf{bind})$ on Type
- two **morphisms of modules over** T,

$$App : T \times T \to T$$
$$Abs : T(\_ + 1) \to T$$

"Module morphism" expresses **commutativity with** bind:

$$\begin{aligned} \mathsf{bind}\ f \circ App &= App \circ (\mathsf{bind}\ f)^2 \\ \mathsf{bind}\ f \circ Abs &= Abs \circ \mathsf{bind}\ (\mathsf{shift}\ f) \end{aligned}$$

Lemma (Initial semantics for Lc, H & M '07)

$(\mathsf{Lc}, \mathsf{app}, \mathsf{abs})$ *is the initial algebra, where* $\mathsf{Lc} = (\mathsf{Lc}, \mathsf{var}, \mathsf{subst})$

Goal: characterize **co**data types with **co**substitution

Goal

dualize techniques of H & M to characterize

- **co**data types in **intensional ML type theory** with
- **co**substitution

as **terminal** object

In this talk

- streams
- infinite triangular matrices

Outline

Streams over a base type

- Streams = infinite lists over some base type A

$$\underbrace{a_0}_{\text{head}} \mid \underbrace{a_1 \quad a_2 \quad \cdots}_{\text{tail}}$$

- Specified by destructors

$$\frac{s : \mathsf{Stream} A}{\mathsf{head}_A(s) : A} \qquad \frac{s : \mathsf{Stream} A}{\mathsf{tail}_A(s) : \mathsf{Stream} A}$$

- "Sameness" = bisimilarity in IMLTT

$$\frac{s \sim s'}{\mathsf{head}(s) = \mathsf{head}(s')} \qquad \frac{s \sim s'}{\mathsf{tail}(s) \sim \mathsf{tail}(s')}$$

Structure on streams

- $(\mathsf{Stream}A, \sim)$ is a *setoid*,

$$\mathsf{Stream} : \mathsf{Type} \to \mathsf{Setoid}$$

- Canonical cosubstitution

$$\mathsf{cosubst}_{A,B} : (\mathsf{Stream}A \to B) \to \mathsf{Stream}A \to \mathsf{Stream}B$$

is compatible with bisimilarity:

$$\mathsf{cosubst}_{A,B} : \mathsf{Setoid}(\mathsf{Stream}A, \mathsf{eq}B) \to \mathsf{Setoid}(\mathsf{Stream}A, \mathsf{Stream}B)$$

with

$$\mathsf{eq} : \mathsf{Type} \to \mathsf{Setoid} \qquad \mathsf{eq} \dashv \mathsf{forget}$$

Structure on streams

Lemma

(Stream, head, cosubst) *is a* ***comonad relative to*** eq : Type $\to$ Setoid.

Definition (Relative (co)monad, Alten., Chapm. & Uust. '10)

- underlying functor is **not** necessarily **endo**
- needs "mediating" functor (above: eq)

About the destructor tail

Morphisms of modules over monads

characterize commutativity of substitution with constructors

$$\mathsf{app} : \mathsf{Lc} \times \mathsf{Lc} \to \mathsf{Lc}$$

$$\mathsf{subst}\ f \circ \mathsf{app} = \mathsf{app} \circ (\mathsf{subst}\ f)^2$$

Morphisms of **co**modules over **relative** **co**monads

characterize commutativity of **co**substitution with destructors

$$\mathsf{tail} : \mathsf{Stream} \to \mathsf{Stream}$$

$$\mathsf{tail} \circ \mathsf{cosubst}\ f = \mathsf{cosubst}\ f \circ \mathsf{tail}$$

Terminal semantics for Stream

Definition (Category of coalgebras)

A coalgebra for the signature of Stream is given by a pair (S, t):

- a comonad S relative to $\mathsf{eq} : \mathsf{Type} \to \mathsf{Setoid}$
- a morphism of comodules over S

$$t : S \to S$$

Morphisms: ...

Lemma

$(\mathsf{Stream}, \mathsf{tail})$ *is the terminal object in the above category.*

Outline

1. Syntax: inductives and substitution
2. Homogeneous cosyntax: streams
3. Heterogeneous cosyntax: infinite triangular matrices

An example of cosyntax: infinite triangular matrices

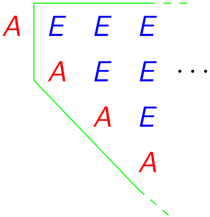
Tri: the codata type of infinite triangular matrices

- omit redundant information below the diagonal
- have a **variable** type A of diagonal elements
  - e.g. invertible elements
- a fixed type E of elements for rest of matrix
- usage: Pascal matrices (binomial coefficients), mathematical physics (infinite-dim. problems)

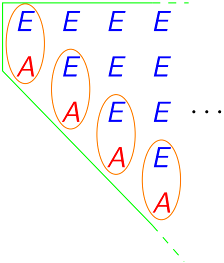
$$\begin{array}{cccc} A & E & E & E \\ & A & E & E \; \cdots \\ & & A & E \\ & & & A \end{array}$$

Matrices through trapezia: the destructors of Tri

$$\dfrac{t : \mathsf{Tri}A}{\mathsf{top}_A(t) : A}$$

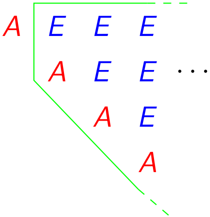


$$\dfrac{t : \mathsf{Tri}A}{\mathsf{rest}_A(t) : \mathsf{Tri}(E \times A)}$$

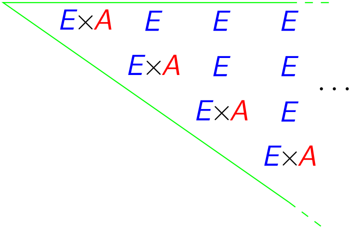


Matrices through trapezia: the destructors of Tri

$$\frac{t : \mathsf{Tri}A}{\mathsf{top}_A(t) : A}$$

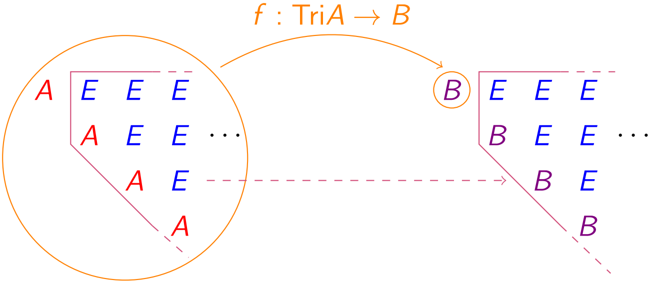


$$\frac{t : \mathsf{Tri}A}{\mathsf{rest}_A(t) : \mathsf{Tri}(E \times A)}$$



Redecoration

$$\mathsf{redec}_{A,B} : (\mathsf{Tri}A \to B) \to (\mathsf{Tri}A \to \mathsf{Tri}B)$$



$$\mathsf{top} \circ \mathsf{redec}\ f := f \quad \text{and}$$

$$\mathsf{rest} \circ \mathsf{redec}\ f := \mathsf{redec}\ (\mathsf{lift}\ f) \circ \mathsf{rest}$$

with $\mathsf{lift}\ f : \mathsf{Tri}(E \times A) \to E \times B$

Tri is a weak constructive comonad

Sameness = bisimilarity

Bisimilarity ∼ coinductively defined via destructors

$$\frac{t \sim t'}{\mathsf{top}(t) = \mathsf{top}(t')} \qquad \frac{t \sim t'}{\mathsf{rest}(t) \sim \mathsf{rest}(t')}$$

Lemma (Matthes and Picard '11)

$(\mathsf{Tri} : \mathsf{Type} \to \mathsf{Type}, \mathsf{top}, \mathsf{redec})$ *forms a "weak constructive comonad".*

⇝ "weak constructive" refers to compatibility conditions with bisimilarity

Tri is a relative comonad

Alternatively, $\mathsf{Tri}A$ is a setoid rather than a (plain) type

$$\begin{aligned} \mathsf{top}_A &: \mathsf{Setoid}(\mathsf{Tri}A, \mathsf{eq}A) \\ \mathsf{redec}_{A,B} &: \mathsf{Setoid}(\mathsf{Tri}A, \mathsf{eq}B) \to \mathsf{Setoid}(\mathsf{Tri}A, \mathsf{Tri}B) \end{aligned}$$

with $\mathsf{eq} : \mathsf{Type} \to \mathsf{Setoid}$

Tri is a relative comonad

Alternatively, $\mathsf{Tri}A$ is a setoid rather than a (plain) type

$$\mathsf{top}_A : \mathsf{Setoid}(\mathsf{Tri}A, \mathsf{eq}A)$$
$$\mathsf{redec}_{A,B} : \mathsf{Setoid}(\mathsf{Tri}A, \mathsf{eq}B) \to \mathsf{Setoid}(\mathsf{Tri}A, \mathsf{Tri}B)$$

with $\mathsf{eq} : \mathsf{Type} \to \mathsf{Setoid}$

Lemma (Reformulation of Matthes and Picard '11)

$(\mathsf{Tri} : \mathsf{Type} \to \mathsf{Setoid}, \mathsf{top}, \mathsf{redec})$ *forms a* ***comonad relative to*** $\mathsf{eq} : \mathsf{Type} \to \mathsf{Setoid}$.

(Co)modules over (relative) (co)monads

Morphisms of modules over monads

characterize commutativity of substitution with constructors

$$\mathsf{abs} : \mathsf{Lc}(\_ + 1) \to \mathsf{Lc}$$

$$\mathsf{subst}\ f \circ \mathsf{abs} = \mathsf{abs} \circ \mathsf{subst}\ (\mathsf{shift}\ f)$$

Morphisms of **co**modules over **relative co**monads

characterize commutativity of **co**substitution with destructors

$$\mathsf{rest} : \mathsf{Tri} \to \mathsf{Tri}(E \times \_)$$

$$\mathsf{rest} \circ \mathsf{redec}\ f = \mathsf{redec}\ (\mathsf{lift}\ f) \circ \mathsf{rest}$$

Coalgebras for the signature of Tri

Definition (Category of coalgebras)

A coalgebra for the signature of Tri is given by a pair (T, r):

- a comonad T relative to $\mathsf{eq} : \mathsf{Type} \to \mathsf{Setoid}$
- a morphism of comodules over T

$$r : T \to T(E \times \_)$$

Morphisms: ...

Lemma

$(\mathsf{Tri}, \mathsf{rest})$ *is the terminal object in the above category.*

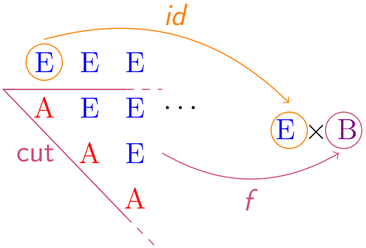
That's **almost** how it works ...

Technical difficulty: definition of lift

Definition of

$$\mathsf{lift}_A : (\mathsf{Tri}A \to B) \to \mathsf{Tri}(E \times A) \to E \times B$$

requires auxiliary function $\mathsf{cut}_A : \mathsf{Tri}(E \times A) \to \mathsf{Tri}A$



A specified cut for any coalgebra

- we were not able to define cut categorically
- fix: every coalgebra (T, r) comes with a **specified**

$$c_A : T(E \times A) \to TA$$

  and equations characterizing c
- $c^{\mathsf{Tri}} := \mathsf{cut}$ for Tri uniquely determined by these equations

Lemma (for real this time)

$(\mathsf{Tri}, \mathsf{cut}, \mathsf{rest})$ *is terminal in the category of coalgebras* (T, c, r).

Formalization

Mechanization in the proof assistant `Coq`

- helped to find a mistake we made (already made by Abel, Matthes & Uustalu '05)
- reuses code by Matthes & Picard '11
- 3000 lines of code, among which 1500 by MP '11
- fully constructive and axiom-free
- tedious: coinduction in `Coq` cumbersome

Summary

- Coinductive type + bisimilarity as setoid in IMLTT
- Stream and Tri are **relative** comonads
- Develop comodules over relative comonads
- Terminal coalgebra semantics for Stream and Tri
- Bisimilarity and redecoration are part of universal object
- Tri not as straightforward as the λ-calculus because of cut
- Mechanization in Coq

Summary

- Coinductive type + bisimilarity as setoid in IMLTT
- Stream and Tri are **relative** comonads
- Develop comodules over relative comonads
- Terminal coalgebra semantics for Stream and Tri
- Bisimilarity and redecoration are part of universal object
- Tri not as straightforward as the λ-calculus because of cut
- Mechanization in Coq

Thanks for your attention

Some references

- Altenkirch, Chapman & Uustalu: *Monads need not be endofunctors*
- Hirschowitz & Maggesi: *Modules over Monads and Linearity*
- Matthes & Picard: *Verification of Redecoration for Infinite Triangular Matrices using Coinduction*
- preprint about this work on the arXiv

TikZ pictures used with permission from Matthes and Picard