

# **High-level signatures and initial semantics**

Ambroise Lafont

joint work with Benedikt Ahrens, André Hirschowitz, Marco Maggesi

CSL 2018

# Overview

**Topic:** specification and construction of untyped syntaxes with variables and a well-behaved substitution (e.g. lambda calculus).

## **Our work:**

1. general notion of **signature** based on **monads** and **modules**.
  - *Caveat:* Not all of them do **generate a syntax**
  - special case: classical **binding signatures**
2. our main result: any **quotient** of algebraic signatures generates a syntax.

**This talk:** explain the words in bold

# Operations covered by our result

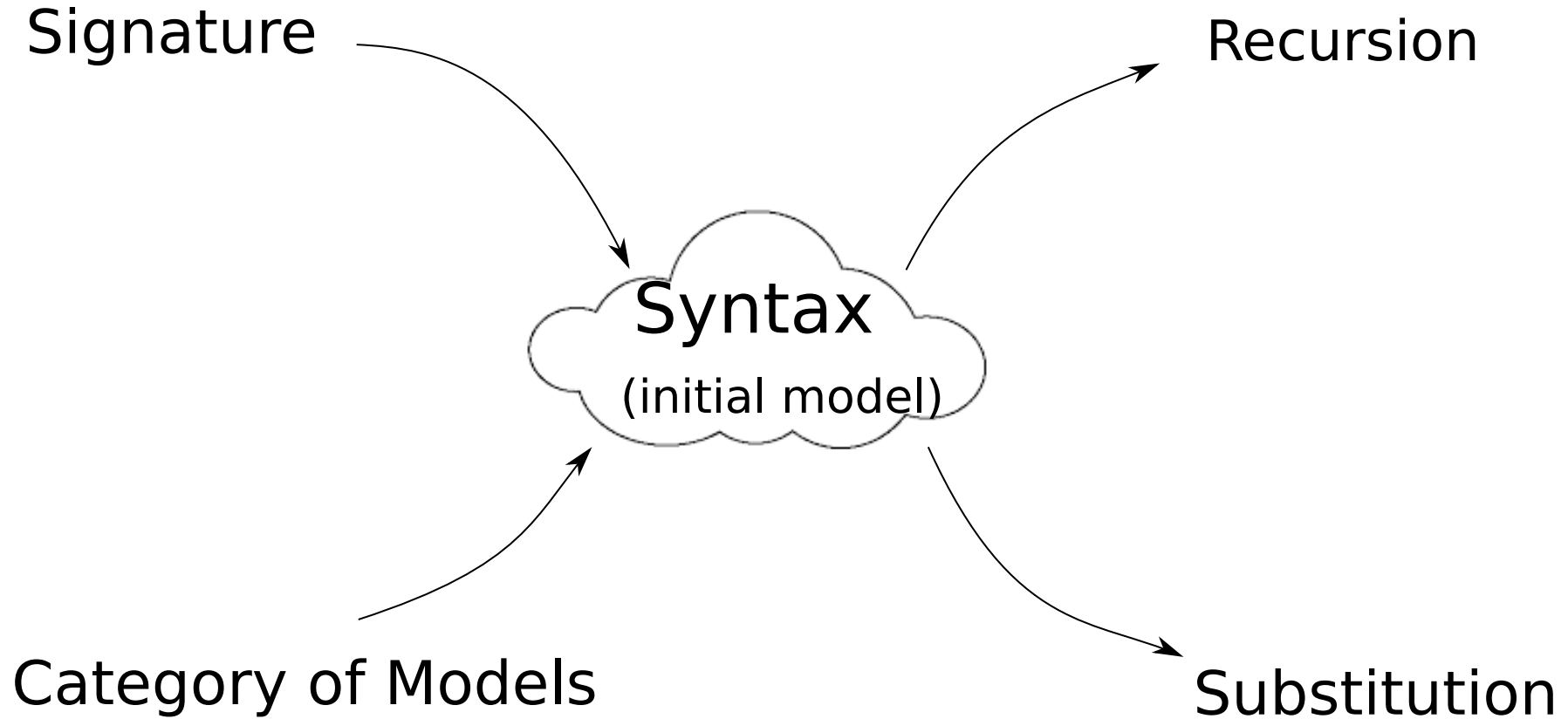
## Some examples:

- Symmetric operations

$$m : T \times T \rightarrow T \quad \text{s.t.} \quad m(t, u) = m(u, t)$$

- Explicit substitution
- Coherent fixed point operation
- Syntactic closure operator

# What is a syntax?



**generates a syntax** = existence of the initial model

# Table of contents

## **1. Binding signatures and their models**

- Categorical formulation of term languages
- Initial semantics for binding signatures
- Categorical formulation of substitution

## 2. Signatures and models based on monads and modules

## 3. Our main result

# Categorical formulation of a term language

**Example:** syntax with a binary operation, a constant, and variables

$$\begin{array}{ll} \text{expr} ::= x & \text{(variable)} \\ \quad | t_1 \star t_2 & \text{(binary operation)} \\ \quad | 0 & \text{(constant)} \end{array}$$

The syntax can be considered as the endofunctor  $B$  (on Set):

$$B : X \mapsto \{\text{expressions over } X\}$$

For example:

$$\begin{aligned} B(\emptyset) &= \{0, 0 \star 0, \dots\} \\ B(\{x, y\}) &= \{0, 0 \star 0, \dots, x, y, x \star y, \dots\} \end{aligned}$$

# Categorical formulation of a term language

Then we have:

$$\star : B \times B \rightrightarrows B$$

$$0 : 1 \rightrightarrows B$$

$$\text{var} : \text{Id}_{\text{Set}} \rightrightarrows B$$

Putting all together:

$$B \times B + 1 + \text{Id}_{\text{Set}} \xrightarrow{\cdot} B$$

i.e.  $B$  is an algebra for the endofunctor  $F \mapsto F \times F + 1 + \text{Id}_{\text{Set}}$  on the category  $\text{End}_{\text{Set}}$ .

Actually,  $B$  can be **defined** to be the initial algebra.

# Binding Signatures

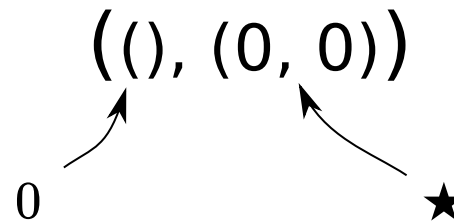
## Definition

**Binding signature** = a family of lists of natural numbers.

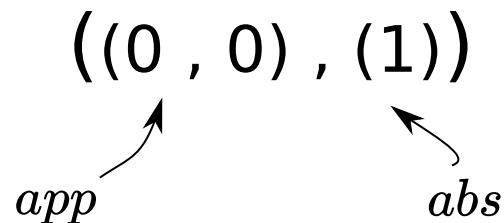
Each list specifies one operation in the syntax:

- length of the list = number of arguments of the operation
- natural number in the list = number of bound variables in the corresponding argument

**Syntax with 0, ★:**



**Lambda calculus:**





# Initial semantics for binding signatures

In the same spirit as in the first example  $(0, \star)$ , any binding signature gives rise to an endofunctor  $\Sigma$  on the category  $\mathbf{End}_{\mathbf{Set}}$ .

A notion of model:  $(\Sigma + \mathbf{Id}_{\mathbf{Set}})$ -algebra

The initial  $(\Sigma + \mathbf{Id}_{\mathbf{Set}})$ -algebra of a binding signature  $\Sigma$  always exists.

Does this initial algebra come with a well-behaved substitution?

# Classical results on initial semantics

The endofunctor  $\Sigma$  induced by a binding signature comes with a *strength* which allows [FPT] to refine the notion of model:

**$\Sigma$ -monoid:**

$\Sigma + \text{Id}_{\text{Set}}$ -algebra **equipped with a well-behaved substitution.**

**$\Sigma$ -monoid morphisms:**

algebra morphisms commuting with substitution.

**Theorem [FPT]:**

The initial  $\Sigma + \text{Id}_{\text{Set}}$ -algebra of a binding signature comes with a well-behaved substitution that makes it initial in the category of  **$\Sigma$ -monoids**.

This suggests defining signatures to be endofunctors on  $\text{End}_{\text{Set}}$  *with strength* (as in [Matthes-Uustalu 2004]).

# Table of contents

1. Binding signatures and their models

## **2. Signatures and models based on monads and modules**

- Our categorical formulation of substitution
- Our take on signatures, models and syntax
- Our take on binding signatures

3. Our main result

# The Big Picture of signatures and models

Binding signatures  $\hookrightarrow$  Endofunctors with strength  $\hookrightarrow$  Our signatures

A **signature**  $\Sigma$  is a functorial assignment:

$$R \mapsto \Sigma(R)$$

A **model of**  $\Sigma$  is a pair:  $(R, \rho : \Sigma(R) \rightarrow R)$

monad  $:=$  endofunctor with substitution

module over a monad  $:=$  endofunctor with substitution

module morphism  $:=$  natural transformation preserving substitution

# The Big Picture of signatures and models

Binding signatures  $\hookrightarrow$  Endofunctors with strength  $\hookrightarrow$  Our signatures

A **signature**  $\Sigma$  is a functorial assignment:

monad



$$R \mapsto \Sigma(R)$$

A **model of**  $\Sigma$  is a pair:

$$(R, \rho : \Sigma(R) \rightarrow R)$$

monad := endofunctor with substitution

module over a monad := endofunctor with substitution

module morphism := natural transformation preserving substitution

# The Big Picture of signatures and models

Binding signatures  $\hookrightarrow$  Endofunctors with strength  $\hookrightarrow$  Our signatures

A **signature**  $\Sigma$  is a functorial assignment:

$$\begin{array}{ccc} \text{monad} & & \text{module over } R \\ & \searrow & \swarrow \\ & R \mapsto \Sigma(R) & \end{array}$$

A **model of**  $\Sigma$  is a pair:  $(R, \rho : \Sigma(R) \rightarrow R)$

monad := endofunctor with substitution

module over a monad := endofunctor with substitution

module morphism := natural transformation preserving substitution

# The Big Picture of signatures and models

Binding signatures  $\hookrightarrow$  Endofunctors with strength  $\hookrightarrow$  Our signatures

A **signature**  $\Sigma$  is a functorial assignment:

$$\begin{array}{ccc} \text{monad} & & \text{module over } R \\ & \searrow & \swarrow \\ & R \mapsto \Sigma(R) & \end{array}$$

A **model of**  $\Sigma$  is a pair:

$$(R, \rho : \Sigma(R) \rightarrow R)$$

monad  $\nearrow$

monad := endofunctor with substitution

module over a monad := endofunctor with substitution

module morphism := natural transformation preserving substitution

# The Big Picture of signatures and models

Binding signatures  $\hookrightarrow$  Endofunctors with strength  $\hookrightarrow$  Our signatures

A **signature**  $\Sigma$  is a functorial assignment:

$$\begin{array}{ccc} \text{monad} & & \text{module over } R \\ & \searrow & \swarrow \\ & R \mapsto \Sigma(R) & \end{array}$$

A **model of**  $\Sigma$  is a pair:

$$\begin{array}{ccc} & (R, \rho : \Sigma(R) \rightarrow R) & \\ \nearrow & & \nwarrow \\ \text{monad} & & \text{module morphism} \end{array}$$

monad := endofunctor with substitution

module over a monad := endofunctor with substitution

module morphism := natural transformation preserving substitution



# Substitution and monads

## Reminder:

- $B(X)$  = expressions built out of  $0$ ,  $\star$  and variables taken in  $X$
- Variables induce a natural transformation  $\eta : \text{Id}_{\text{Set}} \rightarrow B$

**substitution**  $\text{bind} : B(X) \rightarrow (X \rightarrow B(Y)) \rightarrow B(Y)$  subject to satisfy some equations.

A triple  $(B, \eta, \text{bind})$  is called a **monad**.

A **monad morphism** between two monads  $R$  and  $S$  is a family of maps  $(f_X : R(X) \rightarrow S(X))_X$  preserving variables and substitution.

# Preview: Operations are module morphisms


## ★ commutes with substitution

$$(t \star u)[x \mapsto v_x] = t[x \mapsto v_x] \star u[x \mapsto v_x]$$

In the right hand side, substitution acts on a pair of expressions.

## Categorical formulation

$B \times B$  supports  $B$ -substitution   $B \times B$  is a **module over**  $B$

★ commutes with substitution  ★ :  $B \times B \rightarrow B$  is a **module morphism**

# Building blocks for binding signatures

Essential constructions of **modules over a monad  $R$** :

- $R$  itself
- $M \times N$  for any modules  $M$  and  $N$  (in particular,  $R \times R$ )
- The **derivative of a module  $M$**  is the module  $M'$  defined by  $M'(X) = M(X + \{\bullet\})$ .

The derivative is used to model an operation binding a variable (Cf next slide).

# Syntactic operations are module morphisms

A **module morphism** between two modules  $M$  and  $N$  on the same monad  $R$  is a family of maps  $(f_x:M(X) \rightarrow N(X))_X$  commuting with substitution.

$$id_M : M \rightarrow M$$

the family of identity maps  $(id_{M(X)}:M(X) \rightarrow M(X))_X$  for any module  $M$

$$\star : B \times B \rightarrow B$$

$$app : L \times L \rightarrow L$$

the application operation of the lambda calculus monad  $L$ .

$$abs : L' \rightarrow L$$

Indeed, in  $\lambda x.t$ , the term  $t$  depends on an additional free variable  $x$ :

If  $t \in L(Y + \{x\}) = L'(\mathbf{Y})$ , then  $abs(t) = \lambda x.t \in L(Y)$

# The Big Picture again

A **signature**  $\Sigma$  is a functorial assignment:

$$\begin{array}{ccc} \text{monad} & & \text{module over } R \\ & \searrow & \swarrow \\ & R \mapsto \Sigma(R) & \end{array}$$

A **model of**  $\Sigma$  is a pair:

$$\begin{array}{ccc} & (R, \rho : \Sigma(R) \rightarrow R) & \\ \nearrow \text{monad} & & \nwarrow \text{module morphism} \end{array}$$

A **model morphism**  $m : (R, \rho) \rightarrow (S, \sigma)$  is a monad morphism commuting with the module morphism:

$$\begin{array}{ccc} \Sigma(R) & \xrightarrow{\rho} & R \\ \Sigma(m) \downarrow & & \downarrow m \\ \Sigma(S) & \xrightarrow{\sigma} & S \end{array}$$

# Syntax

## Definition

Given a signature  $\Sigma$ , its **syntax** is an initial object in its category of models.

**Question:** Does the syntax exist for every signature?

**Answer:** No.

**Counter-example:** the signature  $R \mapsto \mathcal{P} \circ R$



powerset endofunctor on Set

# Examples of signatures generating syntax

- $R \mapsto 1 + R \times R$

**Model:** ( $R$  equipped with module morphisms  $1 \rightarrow R$   
and  $R \times R \rightarrow R$ ).

The syntax is our previous  $(0, \star)$  language.

- $R \mapsto R \times R + R'$

Models are monads  $R$  equipped with two module morphisms:

$R \times R \rightarrow R$  and  $R' \rightarrow R$ .

The syntax is lambda calculus.

# Algebraic signatures

More generally, the syntax exists for any signature induced by a disjoint sum of products of finite derivatives of the monad ( $R \mapsto R' \times R'' \times R''' + R \times R'' \times R''' \times R + \dots$ ).

We call such a signature an **algebraic signature**. They correspond to binding signatures through the inclusion:

Binding signatures  $\hookrightarrow$  Endofunctors with strength  $\hookrightarrow$  Our signatures

**Our main result:** Quotients of algebraic signatures generate a syntax.



# Table of contents

1. Binding signatures and their models
2. Signatures and models based on monads and modules
- 3. Our main result**
  - Definition of presentable signatures
  - Generated syntax for presentable signatures
  - Examples of presentable signatures

# Quotient of a signature

## Quotient of a set:

A quotient of a set  $X$  is a set  $Y$  together with a surjection  $p : X \rightarrow Y$ .

$$x \sim x' \iff p(x) = p(x')$$

## Quotient of a signature:

A quotient of a signature  $\Sigma$  consists of:

- a signature  $\Psi$
- a (natural) family of surjective module morphisms  $(f_R : \Sigma(R) \rightarrow \Psi(R))_R$

$$R \mapsto \begin{array}{c} \Sigma(R) \\ \downarrow f_R \\ \Psi(R) \end{array}$$

# Syntax for presentable signatures

## Definition

A **presentable signature** is a quotient of an algebraic signature.

## Theorem

Any presentable signature generates a syntax.

**Question:** Are there interesting examples of presentable signatures?

**Answer:**

- Symmetric operations
- Explicit substitution
- Coherent fixed point operation
- ...

# Example 1: Symmetric operations

**Binary commutative operation  $+$ :**

$$t + u = u + t$$

As a quotient of an algebraic signature:

$$R \mapsto \begin{array}{c} R \times R \\ \downarrow \\ R \times R / \{(x, y) \sim (y, x)\} \end{array}$$

This generalizes to **n-ary permutation invariant operations**.

# Example 2: Explicit substitution

An operation  $\_ \langle x_i \mapsto t_i \rangle$  satisfying coherence equations:

- invariance under **permutation**

$$F(x, y) \langle x \mapsto t, y \mapsto u \rangle = F(y, x) \langle x \mapsto u, y \mapsto t \rangle$$

- invariance under **weakening**

$$F(x) \langle x \mapsto t, y \mapsto u \rangle = F(x) \langle x \mapsto u \rangle$$

- invariance under **contraction**

$$F(x, y) \langle x, y \mapsto t \rangle = F(x, x) \langle x \mapsto t \rangle$$

# Example 2: Explicit substitution

Signature of explicit substitution as a quotient of the algebraic signature  $\Sigma$ :

$$\Sigma(R) := R' \times R + R'' \times R \times R + R''' \times R \times R \times R + \dots$$

$t\langle x \mapsto u \rangle$

$t\langle x \mapsto u, y \mapsto v \rangle$

$t\langle x \mapsto u, y \mapsto v, z \mapsto w \rangle$

$\Sigma(R)$   
 $\downarrow$   
 $\Sigma(R) / \sim$

$R \mapsto$

- **permutation:**  $t\langle x \mapsto u, y \mapsto v \rangle \sim t[x \rightleftharpoons y]\langle x \mapsto v, y \mapsto u \rangle$
- **weakening:**  $t\langle x \mapsto u \rangle \sim t\langle x \mapsto u, y \mapsto v \rangle$
- **contraction:**  $t\langle x \mapsto u, y \mapsto u \rangle \sim t[y := x]\langle x \mapsto u \rangle$

# Conclusion

## **Summary of the talk:**

- presented a notion of signature and models
- identified a class of signatures that generate a syntax
  - encompasses the classical binding signatures
  - encompasses operations satisfying some equations

## **Future work:**

- add equations (e.g. lambda calculus modulo beta/eta equivalence);
- extend our framework to simply typed syntaxes.

# Conclusion

## **Summary of the talk:**

- presented a notion of signature and models
- identified a class of signatures that generate a syntax
  - encompasses the classical binding signatures
  - encompasses operations satisfying some equations

## **Future work:**

- add equations (e.g. lambda calculus modulo beta/eta equivalence);
- extend our framework to simply typed syntaxes.

Thank you!