

Generalised free extensions

Algebraic normalisation and dependent types

Nathan Corbyn

University of Oxford

ICFP'22 SRC


Background

Background

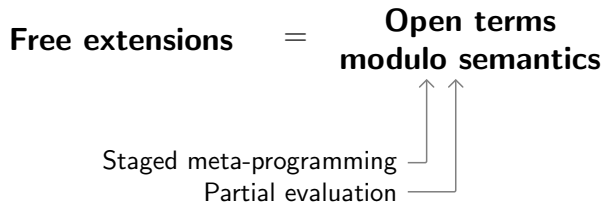
Free extensions = **Open terms
modulo semantics**

Background

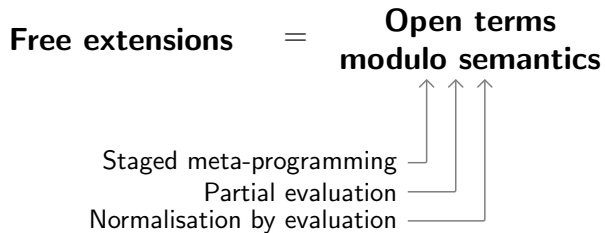
Free extensions = **Open terms
modulo semantics**

Staged meta-programming 

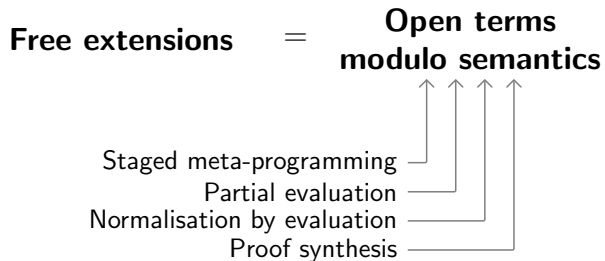
Background



Background



Background

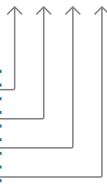


Background

Free extensions = **Open terms modulo semantics**

Key applications

Staged meta-programming
Partial evaluation
Normalisation by evaluation
Proof synthesis



Contributions

Free extensions
of algebras

Contributions

Free extensions
of algebras

e.g., monoids, groups, rings

Contributions



Contributions



Contributions

Free extensions
of algebras
e.g., monoids, groups, rings



Free extensions ^{New}
of *generalised algebras*
dependently-typed operators

The state of play

Generalised algebra

Categories

Monoidal categories

Cartesian categories

CCCs

Applications

Semantics

Linear type systems

First order PLs

Purely functional PLs

Free extension

✓ (Agda)

Todo

Todo

Todo

EXAMPLE: Normalising in a commutative monoid

$$(\mathbb{N}, +)$$

EXAMPLE: Normalising in a commutative monoid

$(\mathbb{N}, +)$ extended by

EXAMPLE: Normalising in a commutative monoid

$(\mathbb{N}, +)$ extended by $\{x, y, z\}$

EXAMPLE: Normalising in a commutative monoid

$\hookrightarrow (\mathbb{N}, +)$ extended by $\{x, y, z\} \longleftarrow$
Concrete monoid *Free* variables

EXAMPLE: Normalising in a commutative monoid

Syntax: sums built from a binary operator with unit.

$$([x] + [2]) + (([y] + [z]) + ([3] + [x]))$$

EXAMPLE: Normalising in a commutative monoid

Syntax: sums built from a binary operator with unit.

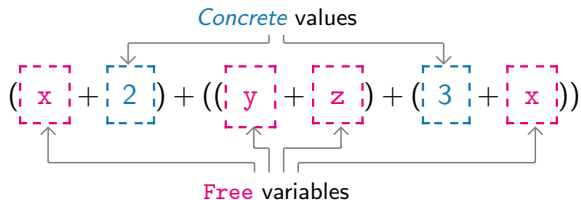
$$((x + 2) + ((y + z) + (3 + x)))$$

Free variables

The diagram illustrates the expression $((x + 2) + ((y + z) + (3 + x)))$ with variables and constants highlighted. The variables x , y , z , and x are enclosed in red dashed boxes, while the constants 2 and 3 are in blue dashed boxes. Arrows from the text "Free variables" point to each of these four variables.

EXAMPLE: Normalising in a commutative monoid

Syntax: sums built from a binary operator with unit.



EXAMPLE: Normalising in a commutative monoid

Normal forms: bags of **free** variables and a *constant*.

EXAMPLE: Normalising in a commutative monoid

Normal forms: bags of **free** variables and a *constant*.

$$x^2$$

EXAMPLE: Normalising in a commutative monoid

Normal forms: bags of **free** variables and a *constant*.

x^2

y^1

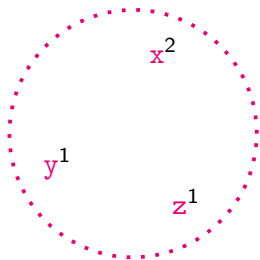
EXAMPLE: Normalising in a commutative monoid

Normal forms: bags of **free** variables and a *constant*.

$$x^2$$
$$y^1$$
$$z^1$$

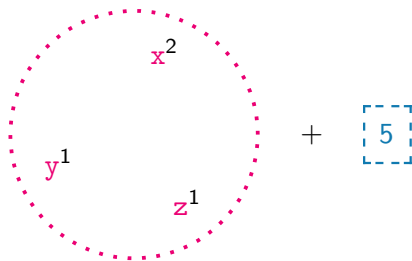
EXAMPLE: Normalising in a commutative monoid

Normal forms: bags of **free** variables and a *constant*.



EXAMPLE: Normalising in a commutative monoid

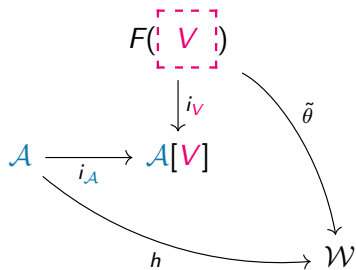
Normal forms: bags of **free** variables and a *constant*.



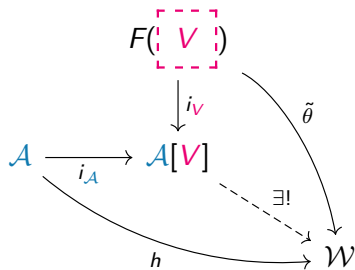
A universal property

$$\begin{array}{ccc} & & F(\boxed{V}) \\ & & \downarrow i_V \\ \mathcal{A} & \xrightarrow{i_{\mathcal{A}}} & \mathcal{A}[V] \end{array}$$

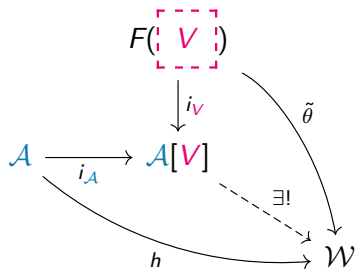
A universal property



A universal property



A universal property



Coproduct with a **free** algebra!

EXAMPLE: Normalising composites in a category

$\text{Vect}_{\mathbb{R}}$

EXAMPLE: Normalising composites in a category

$\text{Vect}_{\mathbb{R}}$ extended by

EXAMPLE: Normalising composites in a category

$$\text{Vect}_{\mathbb{R}} \text{ extended by } \left(\begin{array}{l} \mathbf{X} \mathbf{Y} : \mathcal{O} \\ \mathbf{f} : \mathbb{R}^2 \rightarrow \mathbf{X} \\ \mathbf{g} : \mathbf{X} \rightarrow \mathbf{Y} \\ \mathbf{h} : \mathbf{Y} \rightarrow \mathbb{R}^3 \end{array} \right)$$

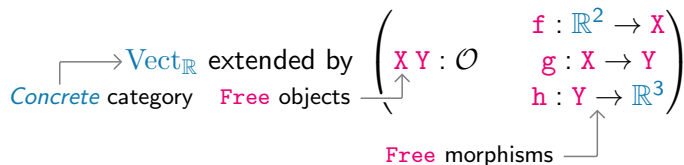
EXAMPLE: Normalising composites in a category

$$\begin{array}{l} \text{Concrete category} \xrightarrow{\quad} \text{Vect}_{\mathbb{R}} \text{ extended by } \left(\begin{array}{l} \mathbf{X} \mathbf{Y} : \mathcal{O} \\ \mathbf{f} : \mathbb{R}^2 \rightarrow \mathbf{X} \\ \mathbf{g} : \mathbf{X} \rightarrow \mathbf{Y} \\ \mathbf{h} : \mathbf{Y} \rightarrow \mathbb{R}^3 \end{array} \right) \end{array}$$

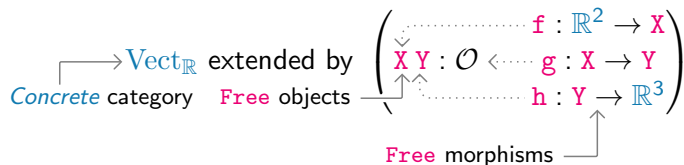
EXAMPLE: Normalising composites in a category

$$\begin{array}{l} \text{Concrete category} \xrightarrow{\quad} \text{Vect}_{\mathbb{R}} \text{ extended by} \\ \text{Free objects} \xrightarrow{\quad} \left(\begin{array}{l} X \ Y : \mathcal{O} \\ f : \mathbb{R}^2 \rightarrow X \\ g : X \rightarrow Y \\ h : Y \rightarrow \mathbb{R}^3 \end{array} \right) \end{array}$$

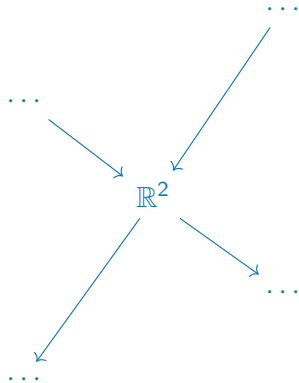
EXAMPLE: Normalising composites in a category



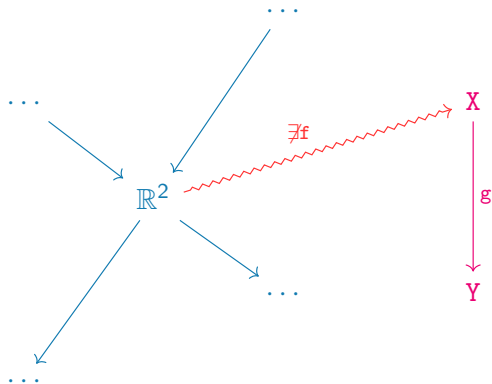
EXAMPLE: Normalising composites in a category



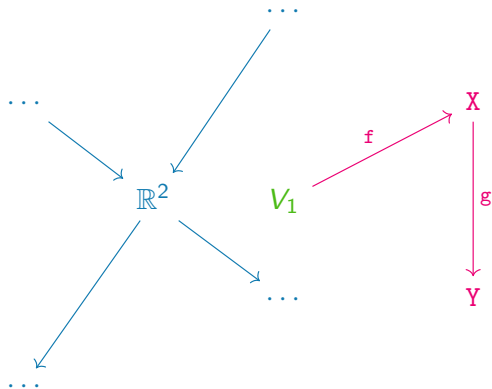
Coproducts are too strong!



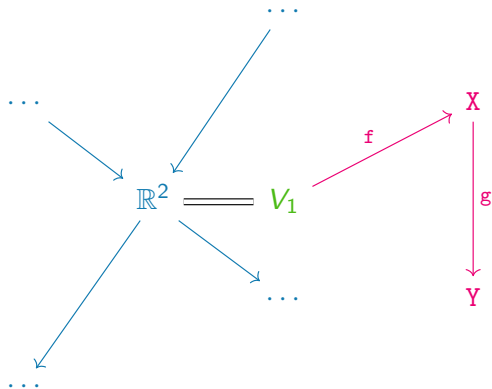
Coproducts are too strong!



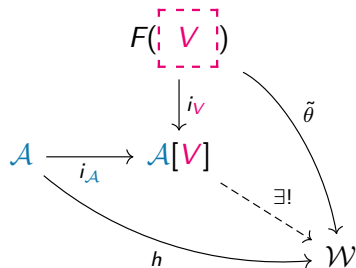
IDEA: Introduce supporting variables



IDEA: Introduce supporting variables *and* quotient

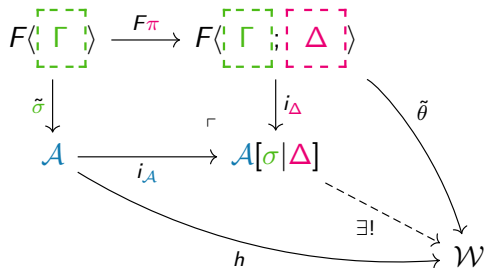


Generalised universal property







Coproduct with a **free** algebra

Generalised universal property



New: Push-out along a weakening!

References

-  Guillaume Allais, Edwin Brady, Nathan Corbyn, Ohad Kammar, and Jeremy Yallop.
Frex: dependently-typed algebraic simplification.
Draft, 2022.
-  John Cartmell.
Generalised algebraic theories and contextual categories.
Annals of Pure and Applied Logic, 32:209–243, 1 1986.
-  Nathan Corbyn.
Proof synthesis with free extensions in intensional type theory.
Technical report, University of Cambridge, 2021.
MEng Dissertation.
-  Jeremy Yallop, Tamara von Glehn, and Ohad Kammar.
Partially-static data as free extension of algebras.
Proc. ACM Program. Lang., 2(ICFP), July 2018.