Polynomial Modelling of Abstract Syntax

Marcelo Fiore
University of Cambridge

Topos Institute Workshop on Polynomial Functors
16. III. 2022

Abstract

The abstract syntax of a formal language is the essential syntactical structure reflecting the semantic import of the language phrases. Abstract syntax has both synthetic and analytic aspects: the former concerns the constructors needed to form phrases, the latter the destructors needed to take them apart. The categorical algebraic point of view regards abstract syntax as an initial algebra: the structure map is synthetic syntax, its inverse is analytic syntax. Initiality provides compositional semantics (as the unique homomorphism to a model) by structural recursion, with an associated principle of structural induction.

Specifications of language phrases are typically given syntactically by means of signatures or, more generally, typing rules. In this talk, I will explain my thesis: (i) that these are notation for polynomial diagrams, and (ii) that abstract syntax arises from the associated polynomial endofunctors by free constructions. I will do so by considering a variety of language features of increasing complexity: mono and multi sorted algebraic term structure, simple and polymorphic type structure (with variable-binding and parametrised-metavariable term structure), and cartesian and/or linear context structure. The mathematical development naturally leads to the consideration of two kinds of polynomial functors: the traditional one between slice categories, arising from locally cartesian closed structure, and another one between presheaf categories, arising from essential geometric morphisms. The former polynomial functors (and their initial algebras) have a type-theoretic rendering as indexed containers (and general trees) that is directly implementable in dependently-typed proof assistants. This is not so for the latter polynomial functors and I will present an approach to bridging this gap via adjoint modalities.

Complementary Material

M.Fiore, G.Plotkin and D.Turi. Abstract syntax and variable binding. In LICS 1999.

M.Fiore. Second-order and dependently-sorted abstract syntax. In LICS 2008.

M.Fiore. Algebraic Type Theory. Note 2008.

M.Fiore and O. Mahmoud. Second-order algebraic theories. In MFCS 2010.

M. Fiore and C.-K. Hur. Second-order equational logic. In CSL 2010.

M.Hamana and M.Fiore. A Foundation for GADTs and Inductive Families: Dependent Polynomial Functor Approach. In WGP 2011.

M.Fiore. Discrete Generalised Polynomial Functors. In ICALP 2012.

M.Fiore and M.Hamana. Multiversal polymorphic algebraic theories: Syntax, semantics, translations, and equational logic. In LICS 2013.

M. Fiore and O. Mahmoud. Functorial semantics of second-order algebraic theories. Arxiv 2014.

M.Fiore. Algebraic Simple Type Theory. Note 2017.

N.Arkor and M.Fiore. Algebraic models of simple type theories: A polynomial approach. In LICS 2020.

M.Fiore and D.Szamozvancev. Formal metatheory of second-order abstract syntax. In POPL 2022.

M.Fiore. Mathematical and Computational Metatheory of Second-Order Algebraic Theories. 2022.
Slides: https://www.math.uwo.ca/faculty/kapulkin/seminars/hottestfiles/Fiore-2022-02-17-HoTTEST.pdf
Talk: https://www.youtube.com/watch?v=nP2J9SJ9DVM

Modelling

the activity of using mathematical descriptions of a system to make calculations or predictions.

Syntax

The structure of statements or elements in a language

EXAMPLES

mathematical
logical
type-theoretic
programming

Syntax

[Birkhoff]

Algebraic Languages

sorts	V	R
operators	$0 : V$	$\cdot : R, V \to V$
	$+ : V, V \to V$	

Syntax

Algebraic Languages

sorts V $\quad$ R

operators $0 : V$ $\quad$ $\cdot : R, V \to V$

$+ : V, V \to V$

Typing rules

$$\frac{}{0 : V}$$

$$\frac{t_1 : V \quad t_2 : V}{t_1 + t_2 : V}$$

$$\frac{s : R \quad t : V}{s \cdot t : V}$$

Syntax

Algebraic Languages

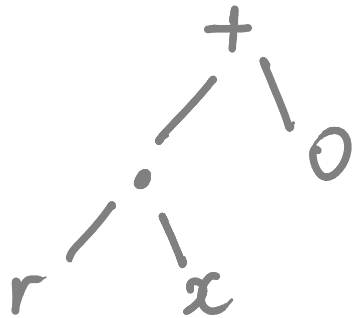
sorts $\quad V \qquad\qquad R$

operators $\quad 0 : V \qquad\qquad \cdot : R, V \to V$

$\qquad\qquad + : V, V \to V$

(1) terms

$$r : R,\ x : V \vdash r \cdot x + 0 : V$$



Syntax

Algebraic Languages

sorts $\quad V \qquad\qquad R$

operators $\quad 0 : V \qquad\qquad \cdot : R, V \to V$

$\qquad\qquad\quad + : V, V \to V$

(1) terms

(2) equational/rewriting theories

(3) algebraic models

(4) sound and complete reasoning

(5) free constructions.

Binding operators

[Frege]

- calculus

$$\dfrac{\dfrac{y : V,\ x : V \vdash x+y : V \qquad y : V \vdash a : V}{y : V \vdash \partial(x.\,x+y)|_0 : V \qquad \vdash b : V}}{\vdash \partial(y.\ \partial(x.\ x+y)|\,a)|b : V}$$

Binding operators

- calculus

$$\dfrac{\dfrac{y : V, x : V \vdash x + y : V \qquad y : V \vdash a : V}{y : V \vdash \partial(x. x+y)|_a : V} \qquad \vdash b : V}{\vdash \partial(y. \partial(x. x+y)|_a)|_b : V}$$

$$\equiv \partial(v. \partial(u. u+v)|_a)|_b$$

$$\equiv \partial(x. \partial(y. y+x)|_a)|_b$$

Binding operators

- calculus

$$\dfrac{\dfrac{y:V,\, x:V \vdash x+y : V}{y:V \vdash \partial(x.\,x+y)|_0 : V} \qquad y:V \vdash a:V \qquad \vdash b:V}{\vdash \partial(y.\ \partial(x.\ x+y)|a)|b \ : V}$$

partial derivative operator

$$\partial : (V)V,\, V \to V$$

- Logic / type theory

sort

$*$

operators

$$\Rightarrow \, : \, *, * \to *$$

$$\forall \, : \, (*) * \to *$$

terms

$$\vdash \forall(\alpha . \alpha) : *$$

$$\beta : * \vdash \forall(\alpha . \alpha \Rightarrow \beta) : *$$

- programming / logic

sort

$$*$$

operators

$$\Rightarrow \; : \; *, * \to *$$

$$\forall \; : \; (*) * \to *$$

terms

$$\frac{\alpha : * ; \; x : \alpha \vdash x : \alpha}{\alpha ; * ; \; - \vdash \underline{\mathrm{fun}}(x . x) : \alpha \Rightarrow \alpha}$$

- programming / logic

sort

$*$

operators

$\Rightarrow : *, * \to *$

$\forall : (*) * \to *$

terms

$$\dfrac{\dfrac{\alpha : *;\ x : \alpha \vdash x : \alpha}{\alpha ; *; - \vdash \underline{\text{fun}}(x.x) : \alpha \Rightarrow \alpha}}{-;- \vdash \Lambda(\alpha.\ \underline{\text{fun}}(x.x)) : \forall(\alpha.\ \alpha \Rightarrow \alpha)}$$

- programming / logic

sort $\quad *$

type operators

$$\Rightarrow \;:\; *, * \to *$$

$$\forall \;:\; (*)* \to *$$

term operators

$$A : *,\ B : * \ \triangleright\ \mathrm{fun}_{A,B} : (A)B \to A \Rightarrow B$$

$$A : (*)* \ \triangleright\ \Lambda_A : (\alpha . A[\alpha]) \to \forall(\alpha . A[\alpha])$$

Thesis

(i) Syntax specifications
= notation for polynomial diagrams

(ii) Abstract syntax
= free constructions w.r.t the associated polynomial functors

Abstract Syntax

[McCarthy]

Essential structure reflecting the semantic import of language phrases.

Desiderata:
- synthetic and analytic syntax
- models
- compositional interpretations
- structural recursive definitions
- structural inductive reasoning principles

Abstract Syntax

Essential structure reflecting the semantic import of language phrases.

Desiderata:
- synthetic and analytic syntax
- models
- compositional interpretations
- structural recursive definitions
- structural inductive reasoning principles

▶ Initial algebras (free constructions) [ADJ]

$$\begin{tikzcd} P(T) \arrow[r, "PI"] \arrow[d, "\tau", "\cong"'] & P(A) \arrow[d, "\alpha"] \\ T \arrow[r, dashed, "\exists! I"] & A \end{tikzcd}$$

$$I = \alpha \circ P(I) \circ \tau^{-1}$$

Algebraic Languages

(signatures $O \to S^* \times S$)

sorts V, R

operators $v_i : V \ (i \in I)$ $\qquad r_j : R \ (j \in J)$

$+ : V, V \to V$ $\qquad \cdot : R, V \to V$

Algebraic Languages

sorts $\quad V, R$

operators $\quad v_i : V \ (i \in I) \qquad r_j : R \ (j \in J)$

$$+ : V, V \to V \qquad \cdot : R, V \to V$$

polynomial diagram [Moerdijk & Palmgren]

$$\begin{tikzcd}
\{+_1, +_2, \cdot_1, \cdot_2\} \arrow[r, "\text{arguments}"] \arrow[d, "\text{source sorts}"'] & \{v_i, +, r_j, \cdot\} \arrow[d, "\text{target sorts}"] \\
\{V, R\} & \{V, R\}
\end{tikzcd}$$

Algebraic Languages

sorts V, R

operators $v_i : V \ (i \in I)$ $\quad$ $r_j : R \ (j \in J)$

$+ : V, V \to V$ $\quad$ $\cdot : R, V \to V$

polynomial diagram

$$
\begin{tikzcd}
2+2 \arrow[rr] \arrow[d] & & I+1+J+1 \arrow[d] \\
2 & & 2
\end{tikzcd}
$$

polynomial endofunctor on $\underline{Set}/2$

$$P(X,Y) = (I + X \times X + Y \times X, J)$$

Binding Operators

signatures

$O \to (S^* \times S)^* \times S^* \times S$

(1) $\forall : (*)* \to *$

- One needs to take contexts seriously:

$$\underbrace{\alpha_1 : *, \ldots, \alpha_n : *}_{\text{in } \mathbb{F}[*]} \vdash A : *$$

free cocartesian category on $*$

$\rightsquigarrow$ $\underline{\underline{\mathrm{Set}}}^{\mathbb{F}[*]}$

[Fiore & Plotkin & Turi]

typing rule

$$\frac{\Delta, \alpha : * \vdash A : *}{\Delta \vdash \forall(\alpha . A) : *}$$

typing rule

$$\frac{\Delta, \alpha : * \vdash A : *}{\Delta \vdash \forall(\alpha . A) : *}$$

polynomial diagram

$V = y(*)$

$$\begin{tikzcd} & V \arrow[dl] \arrow[r] & 1 \arrow[dr] & \\ 1 & & & 1 \end{tikzcd}$$

in $\underline{\underline{Set}}^{\mathbb{F}[*]}$

P-algebra

$$X(-+*) \to X(-)$$

typing rule

$$\frac{\Delta, \alpha : * \vdash A : *}{\Delta \vdash \forall(\alpha . A) : *}$$

polynomial diagram

$V = y(*)$

$$\begin{tikzcd} & V \arrow[dl] \arrow[r] & 1 \arrow[dr] & \\ 1 & & & 1 \end{tikzcd}$$

in $\underline{\underline{Set}}^{\mathbb{F}[*]}$

polynomial endofunctor on $\underline{\underline{Set}}^{\mathbb{F}[*]}$

$$P(X) = X^{V} \cong X(-+*)$$

[Fiore & Plotkin & Turi]

(2) $\Rightarrow : *, * \to *$

$$D_{\Rightarrow} = \begin{tikzcd} & 2 \arrow[r] \arrow[dl] & 1 \arrow[dr] & \\ 1 & & & 1 \end{tikzcd}$$

model $\begin{tikzcd} T^2 \arrow[d, "\Rightarrow"] \\ T \end{tikzcd}$ in $\underline{\underline{Set}}$

(2) $\Rightarrow : *, * \to *$ model $\begin{array}{c} T^2 \\ \downarrow \Rightarrow \\ T \end{array}$ in $\underline{\underline{Set}}$

$$\alpha : *, \beta : * \triangleright \mathit{fun}_{\alpha,\beta} : (\alpha)\beta \to \alpha \Rightarrow \beta$$

$$\frac{\Gamma, x : \alpha \vdash e : \beta}{\Gamma \vdash \underline{\mathit{fun}}(x.e) : \alpha \Rightarrow \beta}$$

(2) $\Rightarrow : *, * \to *$ model $\begin{array}{c} T^2 \\ \downarrow \Rightarrow \\ T \end{array}$ in $\underline{\underline{Set}}$

$$\alpha : *, \beta : * \triangleright \mathrm{fun}_{\alpha,\beta} : (\alpha)\beta \to \alpha \Rightarrow \beta$$

$$\frac{\Gamma, x : \alpha \vdash e : \beta}{\Gamma \vdash \underline{\mathrm{fun}}(x.e) : \alpha \Rightarrow \beta}$$

polynomial diagram

[Fiore]

$$V = \sum_{\alpha \in T} y(\alpha)$$

$$\begin{tikzcd} T & V \times T \arrow[l, "\pi_2"'] \arrow[r, "\nu \times T"] & T \times T \arrow[r, "\Rightarrow"] & T \end{tikzcd}$$

in $\underline{\underline{Set}}^{\mathbb{F}[T]}$

(2) $\alpha : *,\ \beta : * \blacktriangleright \mathrm{fun}_{\alpha,\beta} : (\alpha)\beta \to \alpha \Rightarrow \beta$

polynomial diagram

$$V = \sum_{\alpha \in T} y(\alpha) \qquad \begin{tikzcd} & V \times T \arrow[r, "\nu \times T"] \arrow[dl, "\pi_2"'] & T \times T \arrow[dr, "\Rightarrow"] & \\ T & & & T \end{tikzcd}$$

in $\underline{\underline{Set}}^{\#[T]}$

polynomial endofunctor on $Set^{\#[T]}/T$

$$P\begin{pmatrix} X \\ \downarrow \\ T \end{pmatrix} = \begin{tikzcd} \sum_{\alpha,\beta \in T} X_\beta^{y(\alpha)} \arrow[d] \\ T \times T \arrow[d, "\Rightarrow"] \\ T \end{tikzcd}$$

$$\begin{tikzcd} \sum_{\alpha,\beta \in T} X_\beta^{y_\alpha} \arrow[r] \arrow[d] \arrow[dr, phantom, "P\text{-alg}"] & X \arrow[d] \\ T \times T \arrow[r, "\Rightarrow"'] & T \end{tikzcd}$$

From slices to presheaves

$$P \circlearrowright \underline{\underline{\mathrm{Set}}}^{\mathbb{F}[T]}/T \simeq \left(\underline{\underline{\mathrm{Set}}}^{\mathbb{F}[T]}\right)^T \cong \underline{\underline{\mathrm{Set}}}^{\mathbb{F}[T]\times T} \circlearrowleft P'$$

From slices to presheaves

$$P \circlearrowright \underline{\mathbf{Set}}^{\mathbb{F}[T]}/T \simeq \left(\underline{\mathbf{Set}}^{\mathbb{F}[T]}\right)^{T} \cong \underline{\mathbf{Set}}^{\mathbb{F}[T]\times T} \circlearrowleft P'$$

$$P'(X)(\Gamma,\tau) = \sum_{\substack{\alpha,\beta\in T \\ \alpha\Rightarrow\beta=\tau}} X(\Gamma+\alpha,\beta)$$

$\rightsquigarrow$ a generalised polynomial endofunctor on presheaves

[Fiore, Spivak]

Generalised polynomial functors

$$\begin{tikzcd} \mathbb{A} \arrow[d, "f"'] \\ \mathbb{B} \end{tikzcd} \qquad \begin{tikzcd} \underline{\underline{\mathrm{Set}}}^{\mathbb{A}} \arrow[d, bend right, "f_!"'] \arrow[d, bend left, "f_*"] \\ \underline{\underline{\mathrm{Set}}}^{\mathbb{B}} \arrow[u, "f^*" description] \end{tikzcd} \qquad f_! \dashv f^* \dashv f_*$$

essential geometric morphism

diagram

$$\begin{tikzcd} & \mathbb{E} \arrow[r, "p"] \arrow[dl, "s"'] & \mathbb{B} \arrow[dr, "t"] & \\ \mathbb{I} & & & \mathbb{J} \end{tikzcd}$$

in $\underline{\underline{\mathrm{Cat}}}$

functor

$$\underline{\underline{\mathrm{Set}}}^{\mathbb{I}} \xrightarrow{s^*} \underline{\underline{\mathrm{Set}}}^{\mathbb{E}} \xrightarrow{p_*} \underline{\underline{\mathrm{Set}}}^{\mathbb{B}} \xrightarrow{t_!} \underline{\underline{\mathrm{Set}}}^{\mathbb{J}}$$

typing rule

$$\frac{\Gamma, x : \alpha \vdash e : \beta}{\Gamma \vdash \underline{\mathrm{fun}}\,(x.e) : \alpha \Rightarrow \beta}$$

polynomial diagram [Fiore]

$$\begin{tikzcd}
\mathbb{F}[T] \times T \times T \arrow[r, equal] \arrow[d, "+ \times \mathrm{id}"'] & \mathbb{F}[T] \times T \times T \arrow[d, "\mathrm{id} \times \Rightarrow"] \\
\mathbb{F}[T] \times T & \mathbb{F}[T] \times T
\end{tikzcd}$$

typing rule

$$\frac{\Gamma, x:\alpha \vdash e:\beta}{\Gamma \vdash \underline{\text{fun}}(x.e) : \alpha \Rightarrow \beta}$$

polynomial diagram

$$\begin{tikzcd}
\mathbb{F}[T]\times T\times T \arrow[r, equal] \arrow[d, "+\times id"'] & \mathbb{F}[T]\times T\times T \arrow[d, "id\times\Rightarrow"] \\
\mathbb{F}[T]\times T & \mathbb{F}[T]\times T
\end{tikzcd}$$

polynomial

$$P(X)(\Gamma,\tau) = \sum_{\substack{\alpha,\beta\in T \\ \alpha\Rightarrow\beta = T}} X(\Gamma+\alpha,\beta)$$

Typing rules induce finitely discrete polynomial diagrams

$$
\begin{tikzcd}
 & H \cdot \mathbb{J} \arrow[r, "\nabla"] \arrow[dl] & \mathbb{J} \arrow[dr] & \\
\mathbb{A} & & & \mathbb{A}
\end{tikzcd}
\qquad (H \text{ finite set})
$$

in $\underline{\underline{\mathrm{Cat}}}$

Polymorphic Syntax

$$\frac{\Delta \vdash A : * \qquad \Delta \vdash B : *}{\Delta \vdash A \Rightarrow B : *}$$

$$\frac{\Delta, \alpha : * \vdash A *}{\Delta \vdash \forall(\alpha . A) : *}$$

Polymorphic Syntax

$$\frac{\Delta \vdash A : * \qquad \Delta \vdash B : *}{\Delta \vdash A \Rightarrow B : *} \qquad \frac{\Delta, \alpha : * \vdash A \, *}{\Delta \vdash \forall(\alpha . A) : *}$$

$$\begin{tikzcd}
2 \cdot \mathbb{F}[*] \arrow[r] \arrow[d, "{[id, id]}"'] & \mathbb{F}[*] \arrow[d, "id"] \\
\mathbb{F}[*] & \mathbb{F}[*]
\end{tikzcd}
\qquad
\begin{tikzcd}
1 \cdot \mathbb{F}[*] \arrow[r] \arrow[d, "(-)+*"'] & \mathbb{F}[*] \arrow[d, "id"] \\
\mathbb{F}[*] & \mathbb{F}[*]
\end{tikzcd}$$

$$P(X) = X^2 + X^{\vee} \quad \text{on} \quad \underline{\underline{\mathrm{Set}}}^{\mathbb{F}[*]}$$

$$\frac{\Delta ; \Gamma, x : A \vdash e : B}{\Delta ; \Gamma \vdash fun(x.e) : A \Rightarrow B} \qquad \frac{\Delta, \alpha : * ; \Gamma \vdash t : A}{\Delta ; \Gamma \vdash \Lambda(\alpha.t) : \forall(\alpha.A)}$$

model

$$\begin{tikzcd}
T^2 \arrow[dr, "\Rightarrow"'] & & T^V \arrow[dl, "\forall"] \\
& T &
\end{tikzcd}$$

in $\underline{\underline{Set}}^{\mathbb{F}[*]}$

$$\frac{\Delta;\Gamma, x:A \vdash e:B}{\Delta;\Gamma \vdash \mathrm{fun}(x.e) : A \Rightarrow B} \qquad \frac{\Delta, \alpha:*;\Gamma \vdash t:A}{\Delta;\Gamma \vdash \Lambda(\alpha.t) : \forall(\alpha.A)}$$

model

$$\begin{tikzcd} T^2 \arrow[dr, "\Rightarrow"'] & & T^V \arrow[dl, "\forall"] \\ & T & \end{tikzcd}$$

in $\underline{\underline{\mathbf{Set}}}^{\mathbb{F}[*]}$

$$\mathbb{B} = \mathcal{G}r\left(\mathbb{F}[*] \to \underline{\underline{\mathbf{Cat}}} : \Delta \mapsto \mathbb{F}[T\Delta] \times T\Delta\right)$$

[Hamana]

$$\frac{\Delta;\ \Gamma,\ x : A \vdash e : B}{\Delta;\ \Gamma \vdash \mathrm{fun}(x.e) : A \Rightarrow B}$$

$$\begin{tikzcd}
gr(\Delta \mapsto \mathbb{F}[T\Delta] \times T\Delta \times T\Delta) \arrow[d, "+ \times id"'] & = & gr(\Delta \mapsto \mathbb{F}[T\Delta] \times T\Delta \times T\Delta) \arrow[d, "id \times \Rightarrow"] \\
G & & G
\end{tikzcd}$$

$$\frac{\Delta, \alpha : * ; \Gamma \vdash t : A}{\Delta ; \Gamma \vdash \Lambda(\alpha.t) : \forall(\alpha.A)}$$

$$\begin{array}{ccc}
\mathrm{gr}(\Delta \mapsto \mathbb{F}[T\Delta] \times T(\Delta + *)) & = & \mathrm{gr}(\Delta \mapsto \mathbb{F}[T\Delta] \times T(\Delta + *)) \\
\downarrow & & \downarrow \; \mathrm{id} \times \forall \\
\mathbb{G} & & \mathbb{G}
\end{array}$$

$\Delta, \Gamma \in \mathbb{F}[T\Delta], A \in T(\Delta + *)$

$\mapsto$

$\Delta + *, \Gamma[1] \in \mathbb{F}[T(\Delta + *)], A$

$\leadsto$ algebraic syntax and models of polymorphic languages

[Hamana, Fiore & Hamana]

Summary

- Syntax specifications are polynomial diagrams
- Abstract syntax arises from free constructions on associated polynomial functors

Developments

- Theory of substitution
- Theory of metavariables
- Algebraic models and compositional semantics
- Structural recursion
- Equational theories and logic
- Provably-correct implementation for reasoning and computation
- Higher-dimensional structure

APPENDIX

From presheaves to indexed families via Adjoint Modalities

monad $\Diamond \dashv \Box$ comonad

$$
\begin{tikzcd}
\mathbb{C}_0 \arrow[r, hook] \arrow[d] & \underline{\mathbf{Set}}^{\mathbb{C}_0} \arrow[loop left] \arrow[loop right] \arrow[d, bend right=50, "\blacklozenge"'] \arrow[d, bend left=50, "\blacksquare"] \\
\mathbb{C} \arrow[r, hook] & \underline{\mathbf{Set}}^{\mathbb{C}^{op}} \arrow[u] \arrow[u, phantom, shift right=4, "\dashv"] \arrow[u, phantom, shift left=4, "\dashv"]
\end{tikzcd}
$$

families

presheaves

[Fiore & Szamozvancev]

$\cong \quad \Diamond\text{-}\underline{\mathrm{Alg}}$ (actions) $\qquad \cong \quad \Box\text{-}\underline{\mathrm{coAlg}}$ (coactions)

$$\blacklozenge F(\Gamma) = \sum_{\Delta} F(\Delta) \times \mathbb{C}(\Gamma, \Delta) \qquad \blacksquare F(\Gamma) = \prod_{\Delta} \mathbb{C}(\Delta, \Gamma) \Rightarrow F\Delta$$

Initial-Algebra Lifting Theorem

$$\begin{tikzcd}
\mathcal{S}\text{-coAlg} \arrow[r, "F_\lambda"] \arrow[d] & \mathcal{S}\text{-coAlg} \arrow[d] \\
\mathcal{E} \arrow[r, "F"'] & \mathcal{E}
\end{tikzcd}$$

Initial F-algebras lift to initial F_λ-algebras

$$\begin{tikzcd}
FA \arrow[rr, "\alpha \text{ initial}"] \arrow[d] & & A \arrow[d, "\exists!"', "\text{coalgebra structure}"] \\
FSA \arrow[r, "\lambda"'] & SFA \arrow[r, "S\alpha"'] & SA
\end{tikzcd}$$