Algebraic simple type theory

A polynomial approach

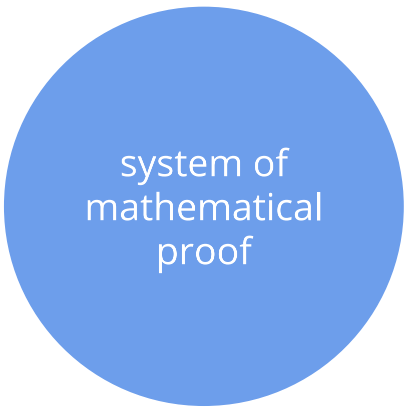
Nathanael Arkor & Marcelo Fiore

Department of Computer Science and Technology
University of Cambridge

Category Theory 2019

What is a type theory?

What is a type theory?



What is a type theory?

system of mathematical proof

specification for a programming language

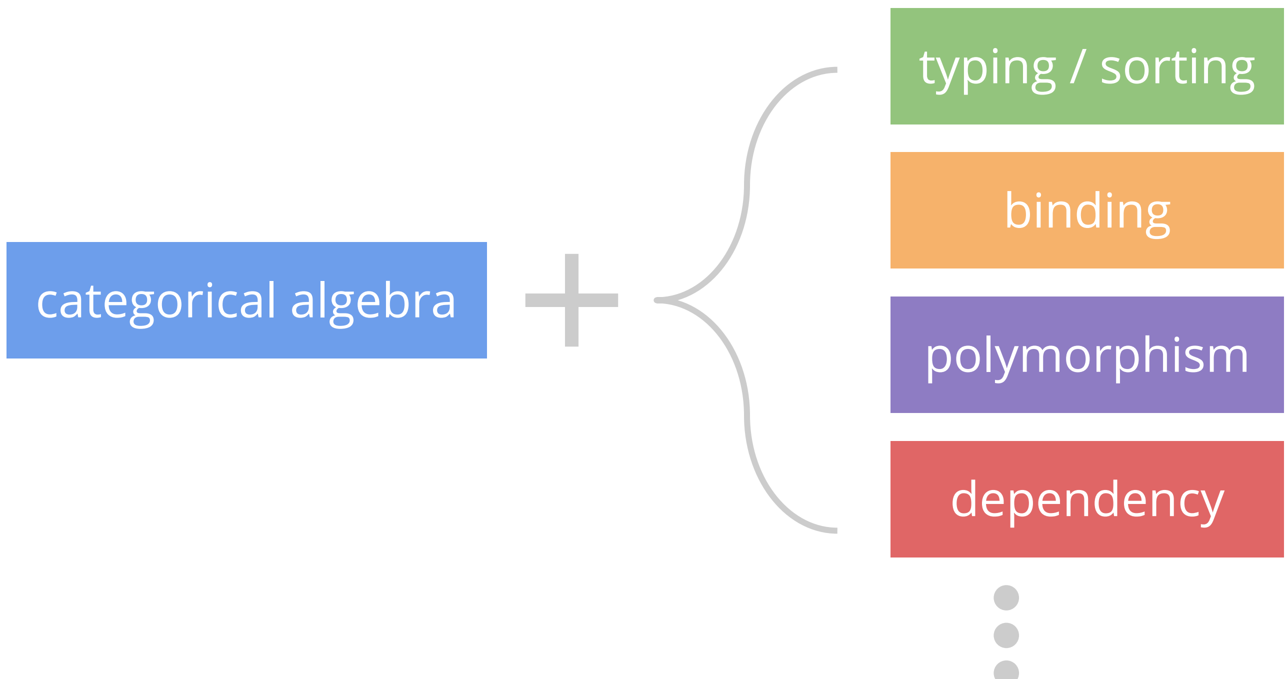
What is a type theory?

system of mathematical proof

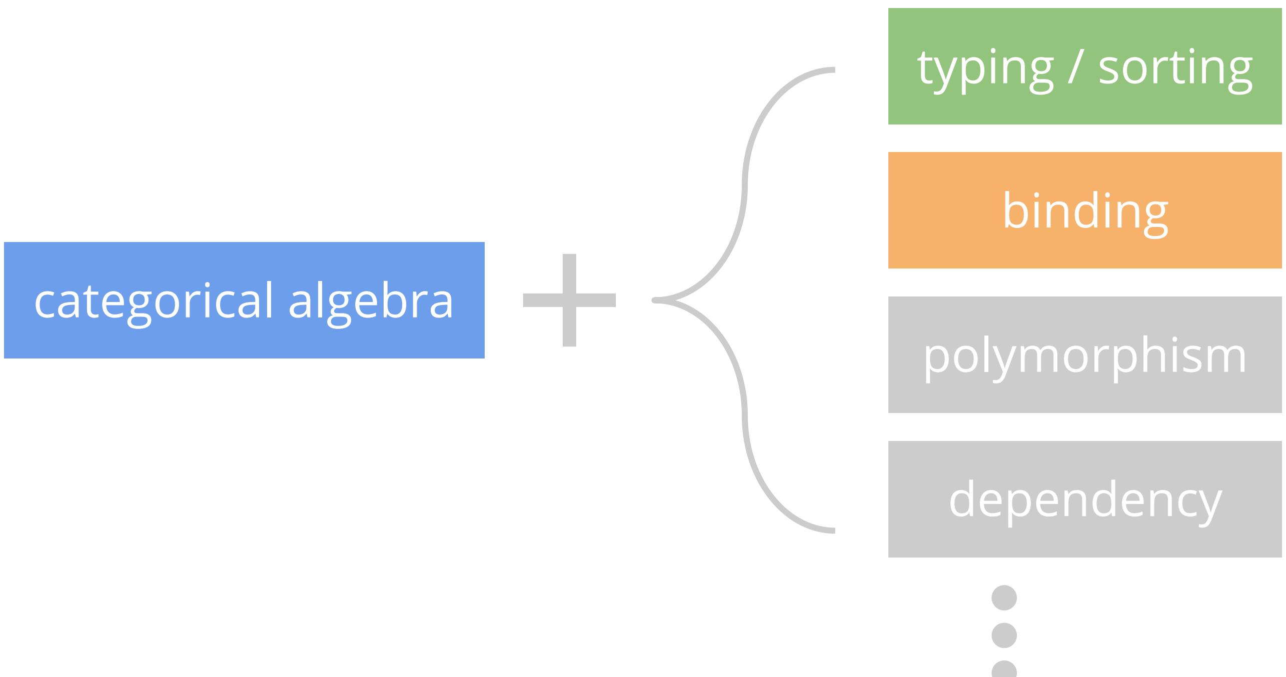
specification for a programming language

internal language of a category

Algebraic type theory



Algebraic simple type theory



e.g. λ-calculus, computational λ-calculus, predicate logic

Typing judgements

$$x_1 : \tau_1, \ldots, x_n : \tau_n \vdash t : \tau$$

Examples

Monoid action $x : M, a : A \vdash x \bullet a : A$

Integration $f : \mathbf{R} \longrightarrow \mathbf{R} \vdash \int_x f(x)\, dx : \mathbf{R}$

$$\underbrace{x_1 : \tau_1, \ldots, x_n : \tau_n}_{\text{contexts}} \vdash \underbrace{t : \tau}_{\text{terms and types}}$$

$$\underbrace{x_1 : \tau_1, \ldots, x_n : \tau_n}_{\text{contexts}} \vdash \underbrace{t : \tau}_{\text{terms and types}}$$

Cartesian context structures

S	$\in \mathrm{Set}$	the **sorts**, or **types**
$\mathscr{C}$	$\in \mathrm{Cat}$	the **variable contexts**
ε (terminal)	$\in \mathscr{C}$	the **empty context**
$\langle - \rangle$	$: S \longrightarrow \mathscr{C}$	**types as contexts**
$- \times \langle = \rangle$	$: \mathscr{C} \times S \longrightarrow \mathscr{C}$	**context extension**

e.g. any cartesian category

Cartesian context structures

S	$\in \mathrm{Set}$	the **sorts**, or **types**
$\mathscr{C}$	$\in \mathrm{Cat}$	the **variable contexts**
ε (terminal)	$\in \mathscr{C}$	the **empty context**
$\langle - \rangle$	$: S \to \mathscr{C}$	**types as contexts**
$- \times \langle = \rangle$	$: \mathscr{C} \times S \to \mathscr{C}$	**context extension**

e.g. any cartesian category

Cartesian context structures

S	$\in \mathrm{Set}$	the **sorts**, or **types**
$\mathscr{C}$	$\in \mathrm{Cat}$	the **variable contexts**
ε (terminal)	$\in \mathscr{C}$	the **empty context**
$\langle - \rangle$	$: S \longrightarrow \mathscr{C}$	**types as contexts**
$- \times \langle = \rangle$	$: \mathscr{C} \times S \longrightarrow \mathscr{C}$	**context extension**

e.g. any cartesian category

Cartesian context structures

S	$\in \mathrm{Set}$	the **sorts**, or **types**
$\mathscr{C}$	$\in \mathrm{Cat}$	the **variable contexts**
ε (terminal)	$\in \mathscr{C}$	the **empty context**
$\langle - \rangle$	$: S \longrightarrow \mathscr{C}$	**types as contexts**
$- \times \langle = \rangle$	$: \mathscr{C} \times S \longrightarrow \mathscr{C}$	**context extension**

e.g. any cartesian category

Cartesian context structures

S	$\in \mathrm{Set}$	the **sorts**, or **types**
$\mathscr{C}$	$\in \mathrm{Cat}$	the **variable contexts**
ε (terminal)	$\in \mathscr{C}$	the **empty context**
$\langle - \rangle$	$: S \longrightarrow \mathscr{C}$	**types as contexts**
$- \times \langle = \rangle$	$: \mathscr{C} \times S \longrightarrow \mathscr{C}$	**context extension**

e.g. any cartesian category

$$\underbrace{x_1 : \tau_1, \ldots, x_n : \tau_n}_{\text{contexts}} \vdash \underbrace{t : \tau}_{\text{terms and types}}$$

Term-typing structure

We consider presheaves $\mathscr{C}^{\mathrm{op}} \to \mathbf{Set}$ on a cartesian context structure $(\mathscr{C}, \mathrm{S})$, fibred over S.

$$\begin{tikzcd} \mathrm{T}(\Gamma) \arrow[d, "\tau_\Gamma"] \\ \underline{\mathrm{S}}(\Gamma) \end{tikzcd}$$

$\mathrm{T}(\Gamma)$ — presheaf of terms (in context Γ)

τ_Γ — assignment of types to terms

$\underline{\mathrm{S}}(\Gamma)$ — presheaf of types (in context Γ)

Term-typing structure

We consider presheaves $\mathscr{C}^{\mathrm{op}} \to \mathrm{Set}$ on a cartesian context structure $(\mathscr{C}, \mathrm{S})$, fibred over S.

$$\begin{tikzcd} \mathrm{T}(\Gamma) \arrow[d, "\tau_\Gamma"] \\ \mathrm{S} \end{tikzcd}$$

presheaf of terms (in context Γ)

assignment of types to terms

constant presheaf of types

Terms with a specified type

NB. The fibre T_σ is the set of terms with type σ.

$$\begin{tikzcd}
T_\sigma \arrow[r] \arrow[d] \arrow[dr, phantom, "pb"] & T \arrow[d, "\tau"] \\
1 \arrow[r, "\sigma"'] & \underline{S}
\end{tikzcd}$$

Presheaf of variables

For any context $\Gamma \in \mathscr{C}$, $V(\Gamma)$ is the set of variables in Γ.

$$\begin{tikzcd}
V \overset{\text{def}}{=} \coprod_{\sigma \in S} y\langle\sigma\rangle \arrow[d, "\nu"] \\
\underline{S}
\end{tikzcd}$$

Models of simple type theory

$$\underbrace{x_1 : \tau_1, \ldots, x_n : \tau_n}_{\text{contexts}} \vdash \underbrace{t : \tau}_{\text{terms and types}}$$

Models of simple type theory

$$\underbrace{x_1 : \tau_1, \ldots, x_n : \tau_n}_{\text{contexts}} \vdash \underbrace{t : \tau}_{\text{terms and types}}$$

\+ algebraic structure

Models of simple type theory

$$\Gamma \vdash \underbrace{op(t)}_{\text{term operators}} : \underbrace{Op(\tau)}_{\text{type operators}}$$

$$(t_1, t_2) \mid \pi_1\, t \mid t_1\, t_2 \mid \lambda(x)\, t \mid \dots$$

$$\tau \to \sigma \mid \tau \times \sigma \mid T(\tau) \mid \dots$$

Algebraic structure on types

Type structure is as in universal algebra. For instance, the following operators

$$\tau \to \sigma \mid \tau \times \sigma \mid T(\tau) \mid U$$

induce a signature endofunctor on Set

$$\Sigma_{ty} = X \mapsto X^2 + X^2 + X + 1$$

the algebras for which are sets S with the appropriate structure

$$[[\![\to]\!], [\![\times]\!], [\![T]\!], [\![U]\!]] : \Sigma_{ty}\, S \to S$$

(NB. These signature functors are polynomial.)

How should we define the algebraic structure on terms?

How should we define the algebraic structure on terms?

Natural deduction rules present algebraic structure

How should we define the algebraic structure on terms?

Natural deduction rules present algebraic structure

Polynomials present natural deduction rules

Polynomials & polynomial functors

In a locally cartesian-closed category $\mathcal{E}$, a polynomial is a diagram:

$$A \xleftarrow{f} B \xrightarrow{g} C \xrightarrow{h} D$$

The polynomial functor associated to the polynomial is given by:

$$\Sigma_h \, \Pi_g \, f^* : \mathcal{E}/A \longrightarrow \mathcal{E}/D$$

Polynomials & polynomial functors

We will consider polynomials in $\mathrm{Psh}(\mathscr{C})$, inducing polynomial functors $\mathrm{Psh}(\mathscr{C})/\underline{S} \to \mathrm{Psh}(\mathscr{C})/\underline{S}$, where $(\mathscr{C}, S)$ is a cartesian context structure with algebraic structure $\Sigma_{ty} S \to S$.

Let P be a polynomial in $\mathrm{Psh}(\mathscr{C})$. Algebras for the corresponding polynomial functor are bundles $\tau : T \to \underline{S}$ together with morphisms as in the following.

$$F_P\left(\begin{array}{c} T \\ \downarrow \tau \\ \underline{S} \end{array}\right) \xrightarrow{[\![P]\!]} \left(\begin{array}{c} T \\ \downarrow \tau \\ \underline{S} \end{array}\right)$$

Algebraic structure & natural deduction

$$\frac{\Gamma \vdash a : A \qquad \Gamma \vdash b : B}{\Gamma \vdash \mathrm{pair}(a, b) : \mathrm{Prod}(A, B)}\ \text{Prod-}\textsc{intro}$$

Algebraic structure & natural deduction

two type metavariables

$$\frac{\Gamma \vdash a : A \qquad \Gamma \vdash b : B}{\Gamma \vdash \mathrm{pair}(a, b) : \mathrm{Prod}(A, B)}\ \textsc{Prod-intro}$$

$\underline{S} \times \underline{S}$

A B

in $\mathrm{Psh}(\mathscr{C})$

Algebraic structure & natural deduction

$$\dfrac{\Gamma \vdash a : A \qquad \Gamma \vdash b : B}{\Gamma \vdash \mathrm{pair}(a, b) : \mathrm{Prod}(A, B)}\ \text{Prod-}\textsc{intro}$$

output sort

$$\underline{S} \times \underline{S} \xrightarrow{[\![\mathrm{Prod}]\!]} \underline{S}$$

in $\mathrm{Psh}(\mathscr{C})$

Algebraic structure & natural deduction

two hypotheses

$$\frac{\Gamma \vdash a : A \qquad \Gamma \vdash b : B}{\Gamma \vdash \mathrm{pair}(a, b) : \mathrm{Prod}(A, B)}\ \text{Prod-}\textsc{intro}$$

$$\underbrace{\underline{S} \times \underline{S}}_{\Gamma \vdash a : A} + \underbrace{\underline{S} \times \underline{S}}_{\Gamma \vdash b : B} \longrightarrow \underline{S} \times \underline{S} \xrightarrow{[\![\mathrm{Prod}]\!]} \underline{S}$$

in Psh($\mathscr{C}$)

Algebraic structure & natural deduction

$$\frac{\Gamma \vdash a : A \qquad \Gamma \vdash b : B}{\Gamma \vdash \mathrm{pair}(a, b) : \mathrm{Prod}(A, B)}\ \text{Prod-}\textsc{intro}$$

$$\overset{A}{\underline{S}} \times \overset{B}{\underline{S}} + \underset{A'}{\underline{S}} \times \underset{B'}{\underline{S}} \xrightarrow{\nabla_2} \underline{S} \times \underline{S} \xrightarrow{[\![\mathrm{Prod}]\!]} \underline{S}$$

in Psh($\mathscr{C}$)

Algebraic structure & natural deduction

$$\frac{\Gamma \vdash a : A \qquad \Gamma \vdash b : B}{\Gamma \vdash \mathrm{pair}(a, b) : \mathrm{Prod}(A, B)}\ \text{Prod-}\textsc{intro}$$

$$\underline{S} \xleftarrow{[\pi_1, \pi_2]} \underline{S} \times \underline{S} + \underline{S} \times \underline{S} \xrightarrow{\nabla_2} \underline{S} \times \underline{S} \xrightarrow{[\![\mathrm{Prod}]\!]} \underline{S}$$

A B

in Psh($\mathscr{C}$)

Algebraic structure & natural deduction

$$\frac{\Gamma \vdash a : A \qquad \Gamma \vdash b : B}{\Gamma \vdash \mathrm{pair}(a, b) : \mathrm{Prod}(A, B)}\ \text{Prod-}\textsc{intro}$$

$$\underline{S} \xleftarrow{[\pi_1, \pi_2]} \underline{S} \times \underline{S} + \underline{S} \times \underline{S} \xrightarrow{\nabla_2} \underline{S} \times \underline{S} \xrightarrow{[\![\mathrm{Prod}]\!]} \underline{S}$$

A B

in $\mathrm{Psh}(\mathscr{C})$

Algebraic structure & natural deduction

introduction rule

$$\frac{\Gamma \vdash a : A \qquad \Gamma \vdash b : B}{\Gamma \vdash \mathrm{pair}(a, b) : \mathrm{Prod}(A, B)}\ \textsc{Prod-intro}$$

polynomial

$$S \xleftarrow{[\pi_1, \pi_2]} S \times S + S \times S \xrightarrow{\nabla_2} S \times S \xrightarrow{[\![\mathrm{Prod}]\!]} S$$

functor algebra

$$\begin{tikzcd}
T \times T \arrow[r, "{[\![\mathrm{pair}]\!]}"] \arrow[d, "\tau \times \tau"'] & T \arrow[d, "\tau"] \\
S \times S \arrow[r, "{[\![\mathrm{Prod}]\!]}"'] & S
\end{tikzcd}$$

in $\mathrm{Psh}(\mathscr{C})$

Binding algebraic structure & natural deduction

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \mathrm{abs}(x : A \,.\, t) : \mathrm{Fun}(A, B)}\ \text{Fun-}\textsc{intro}$$

Binding algebraic structure & natural deduction

two type metavariables

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \mathrm{abs}(x : A\,.\,t) : \mathrm{Fun}(A, B)}\ \text{Fun-}\textsc{intro}$$

$$\underline{S} \times \underline{S}$$

$A \qquad B$

in $\mathrm{Psh}(\mathscr{C})$

Binding algebraic structure & natural deduction

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \mathrm{abs}(x : A\,.\,t) : \mathrm{Fun}(A, B)}\ \textsc{Fun-intro}$$

output sort

$$\underline{S} \times \underline{S} \xrightarrow{[\![\mathrm{Fun}]\!]} \underline{S}$$

in Psh($\mathscr{C}$)

Binding algebraic structure & natural deduction

one variable

one term

$$\dfrac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \mathrm{abs}(x : A\, .\, t) : \mathrm{Fun}(A, B)}\ \textsc{Fun-intro}$$

$$V \times \underline{S} \longrightarrow \underline{S} \times \underline{S} \xrightarrow{[\![\mathrm{Fun}]\!]} \underline{S}$$

A

B

in Psh($\mathscr{C}$)

Binding algebraic structure & natural deduction

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \mathrm{abs}(x : A\ .\ t) : \mathrm{Fun}(A, B)}\ \textsc{Fun-intro}$$

types of variables

$$V \times \underline{S} \xrightarrow{\nu \times \mathrm{id}} \underline{S} \times \underline{S} \xrightarrow{[\![\mathrm{Fun}]\!]} \underline{S}$$

in Psh($\mathscr{C}$)

Binding algebraic structure & natural deduction

term sort

$$\dfrac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \mathrm{abs}(x : A \,.\, t) : \mathrm{Fun}(A, B)}\ \text{Fun-}\textsc{intro}$$

$$\underline{S} \xleftarrow{\pi_2} V \times \underline{S} \xrightarrow{\nu \times \mathrm{id}} \underline{S} \times \underline{S} \xrightarrow{[\![\mathrm{Fun}]\!]} \underline{S}$$

A B

in $\mathrm{Psh}(\mathscr{C})$

Binding algebraic structure & natural deduction

introduction rule

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \mathrm{abs}(x : A \,.\, t) : \mathrm{Fun}(A, B)}\ \textsc{Fun-intro}$$

polynomial

$$\underline{S} \xleftarrow{\pi_2} V \times \underline{S} \xrightarrow{\nu \times \mathrm{id}} \underline{S} \times \underline{S} \xrightarrow{[\![\mathrm{Fun}]\!]} \underline{S}$$

functor algebra

$$\begin{tikzcd}
\coprod_{A, B \in S} T_B(\Gamma \cdot A) \arrow[r, "{[\![\mathrm{abs}]\!]_\Gamma}"] \arrow[d, "\pi"'] & T(\Gamma) \arrow[d, "\tau_\Gamma"] \\
S \times S \arrow[r, "{[\![\mathrm{Fun}]\!]_\Gamma}"'] & S
\end{tikzcd}$$

in $\mathrm{Psh}(\mathscr{C})$

Algebraic structure & natural deduction

The natural deduction rules corresponding to introduction/elimination can be described by a second-order arity (describing the typing and binding data for each argument).

Each second-order arity induces a polynomial in $\mathrm{Psh}(\mathscr{C})$.

The algebras for their associated polynomial functors are presheaves with the corresponding (typed & binding) term structure.

We can collect the arities into a term signature Σ_{tm}, which itself induces a polynomial.

(NB. We're using the same notation for a signature and the polynomial functor it induces.)

Models of simple type theory

$$\underbrace{x_1 : \tau_1, \ldots, x_n : \tau_n}_{\text{contexts}} \vdash \underbrace{t : \tau}_{\text{terms and types}}$$

\+ algebraic structure

Model homomorphisms

$$(S, \mathscr{C}, \varepsilon, \langle - \rangle, - \times \langle = \rangle, T, \tau, [\![-]\!]_{ty}, [\![-]\!]_{tm}) \longrightarrow (S', \mathscr{C}', \varepsilon', \langle - \rangle', - \times \langle = \rangle', T', \tau', [\![-]\!]'_{ty}, [\![-]\!]'_{tm})$$

$h : S \to S'$ — a Σ_{ty}-algebra homomorphism

$H : \mathscr{C} \to \mathscr{C}'$ — a structure-preserving functor

$f : T \to H^{\star}(T')$ — a morphism in $\mathrm{Psh}(\mathscr{C})/\underline{S}$ preserving the Σ_{tm}-algebra structure

Model homomorphisms

$h : S \to S'$ — a Σ_{ty}-algebra homomorphism

$f : T \to H^{\star}(T')$ — a morphism in $\mathrm{Psh}(\mathscr{C})/\underline{S}$ preserving the Σ_{tm}-algebra structure

$$\begin{tikzcd}
\Sigma_{ty} S \arrow[r, "\Sigma_{ty} h"] \arrow[d] & \Sigma_{ty} S' \arrow[d] \\
S \arrow[r, "h"'] & S'
\end{tikzcd}$$

$$\begin{tikzcd}
\Sigma_{tm} T \arrow[r, "\Sigma_{ty} f"] \arrow[dd] & \Sigma_{tm} \mathfrak{h}(T') \arrow[d] \\
 & \mathfrak{h}(\Sigma'_{tm} T') \arrow[d] \\
T \arrow[r, "f"'] & \mathfrak{h}(T')
\end{tikzcd}$$

where $\mathfrak{h} = h^{*}(- \, H)$

Syntactic models of simple type theory

For any given term and type signature, we want a model of simple type theory freely generated by the syntax.

The model freely generated by the syntax is exactly the initial model.

Since we have no type dependency, we can construct the initial model piecewise.

Initial models of simple type theory

Initial models of simple type theory

- S initial Σ_{ty}-algebra

  as in universal algebra

Initial models of simple type theory

- S initial Σ_{ty}-algebra

  as in universal algebra

- $(\mathscr{C}, \varepsilon)$ free cartesian category on S

  concretely, the opposite of the comma category

  $(\mathbb{F} \longrightarrow \mathrm{Set}) \downarrow (\mathbb{1} \xrightarrow{S} \mathrm{Set})$

Initial models of simple type theory

- S — initial Σ_{ty}-algebra

  as in universal algebra
- $(\mathscr{C}, \varepsilon)$ — free cartesian category on S

  concretely, the opposite of the comma category

  $(\mathbb{F} \longrightarrow \mathrm{Set}) \downarrow (\mathbb{1} \xrightarrow{S} \mathrm{Set})$
- (T, τ) — initial Σ_{tm}-algebra

  using Adámek's initial algebra construction

There's one last thing...

Substitution

$$\Gamma \vdash t\ \underbrace{[{}^{u}/_{x}]}_{???} : \tau$$

Substitution

$$\Gamma \vdash t \underbrace{[{}^{u}/_{x}]}_{\text{an algebraic operation on terms}} : \tau$$

Substitution

$$\frac{}{\Gamma, x : A \vdash \mathrm{var}(x) : A}\ \mathrm{var}$$

$$\frac{\Gamma, x : A \vdash t : B \qquad \Gamma \vdash u : A}{\Gamma \vdash \mathrm{subst}(x : A\ .\ t, u) : B}\ \mathrm{subst}$$

Substitution

$$\frac{}{\Gamma, x : A \vdash \mathrm{var}(x) : A}\ \mathrm{var}$$

$$\underline{S} \longleftarrow 0 \longrightarrow V \xrightarrow{\nu} \underline{S}$$

$$\frac{\Gamma, x : A \vdash t : B \qquad \Gamma \vdash u : A}{\Gamma \vdash \mathrm{subst}(x : A\,.\,t, u) : B}\ \mathrm{subst}$$

$$\underline{S} \xleftarrow{[\pi_2, \pi_1]} V \times \underline{S} + \underline{S} \times \underline{S} \xrightarrow{\nu \times \mathrm{id} + \mathrm{id}} \underline{S} \times \underline{S} \xrightarrow{\pi_2} \underline{S}$$

Substitution

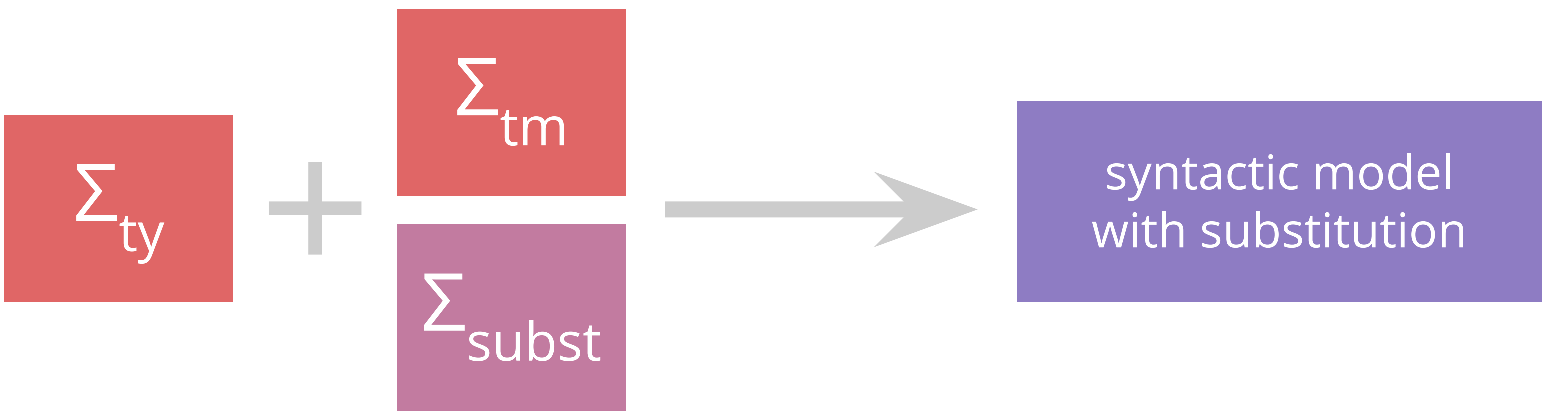
$$\frac{}{\Gamma, x : A \vdash \mathrm{var}(x) : A}\ \mathrm{var}$$

$$\frac{\Gamma, x : A \vdash t : B \qquad \Gamma \vdash u : A}{\Gamma \vdash \mathrm{subst}(x : A\ .\ t, u) : B}\ \mathrm{subst}$$

subject to equational laws…

Initial models of simple type theories

Initial models of simple type theories with substitution



A partial answer

Q. What is a simple type theory?

A partial answer

Q. What is a simple type theory?

A. An initial model

$$(\mathscr{C}, \mathsf{T} \xrightarrow{\tau} \mathsf{S}, [\![-]\!]_{ty}, [\![-]\!]_{tm + subst})$$

A partial answer

Q. What is a simple type theory?

A. An initial model

$$(\mathscr{C}, T \xrightarrow{\tau} S, [\![-]\!]_{ty}, [\![-]\!]_{tm + subst})$$

We can now construct the classifying category and equational logic...

Conclusion

- Models of simple type theory consist of structures for contexts, typed terms and algebraic structure.
- Natural deduction rules that present simple type theories can themselves be presented by polynomials.
- The initial model of simple type theory is the syntactic model of the type theory, and can be constructed explicitly with a free algebra construction.
- We can construct the syntactic model with substitution, from which we can derive a classifying category, demonstrating that the type theory is its internal language.