Polymorphic Abstract Syntax via Grothendieck Construction

Makoto Hamana

Gunma University, Japan

FoSSaCS'11, 30 March, 2011.

This Work

The initial algebra characterisation of
polymorphic typed abstract syntax with variable binding

(1) Second-order polymorphic syntax:
e.g. System $\mathbf{F}$, and variants

(2) Higher-order polymorphic syntax:
e.g. System $\mathbf{F}_\omega$, and variants

History: Initial Algebra Semantics of Syntax

- Abstract syntax is formulated as an *initial algebra*

FO	Abstract syntax	**Set**	ADJ	1975

History: Initial Algebra Semantics of Syntax

▷ *Enriched* abstract syntax are formulated as *initial algebras* in various presheaf categories

FO Abstract syntax	$\mathbf{Set}$	ADJ	1975
Abstract syntax with binding	$\mathbf{Set}^{\mathbb{F}}$	Fiore,Plotkin,Turi Hofmann	1999
Simply-typed syn. with binding	$(\mathbf{Set}^{\mathbb{F}\downarrow U})^{U}$	Fiore	2003
Dependently-sorted syntax	$(\mathbf{Set}^{\mathbf{Fin}[\mathbb{S},\mathbf{Set}]})^{\mathbb{S}}$	Fiore	2008

Initial Algebra Semantics of Abstract Syntax

▷ Algebraic characterisations have produced fruitful applications

Abstract syntax with binding	$\mathbf{Set}^{\mathbb{F}}$	Fiore,Plotkin,Turi	1999
- with metavariables		Hamana,Fiore	2004
- Explicit substitutions		Ghani,Uustalu,Hamana	2006
- Fusion calculus		Miculan	2008
Simply-typed syn. with binding	$(\mathbf{Set}^{\mathbb{F}\downarrow U})^{U}$	Fiore	2003
- Normalisation by evaluation			
- Pre-logical predicates		Katsumata	2004
- Cyclic sharing tree structures		Hamana	2009
- Second-order equational logic		Fiore,Hur	2010

History: Initial Algebra Semantics of Syntax

▷ *Enriched* abstract syntax are formulated as *initial algebras* in various presheaf categories

FO Abstract syntax	$\mathbf{Set}$	ADJ	1975
Abstract syntax with binding	$\mathbf{Set}^{\mathbb{F}}$	Fiore,Plotkin,Turi Hofmann	1999
Simply-typed syn. with binding	$(\mathbf{Set}^{\mathbb{F}\downarrow U})^{U}$	Fiore	2003
Dependently-sorted syntax	$(\mathbf{Set}^{\mathbf{Fin}[\mathbb{S},\mathbf{Set}]})^{\mathbb{S}}$	Fiore	2008

History: Initial Algebra Semantics of Syntax

▷ *Enriched* abstract syntax are formulated as *initial algebras* in various presheaf categories

FO Abstract syntax	$\mathbf{Set}$	ADJ	1975
Abstract syntax with binding	$\mathbf{Set}^{\mathbb{F}}$	Fiore,Plotkin,Turi Hofmann	1999
Simply-typed syn. with binding	$(\mathbf{Set}^{\mathbb{F}\downarrow U})^{U}$	Fiore	2003
Polymorphic Abstract Syntax	?	?	?
Dependently-sorted syntax	$(\mathbf{Set}^{\mathbf{Fin}[\mathbb{S},\mathbf{Set}]})^{\mathbb{S}}$	Fiore	2008

History: Initial Algebra Semantics of Syntax

▷ *Enriched* abstract syntax are formulated as *initial algebras* in various presheaf categories

FO Abstract syntax	$\mathbf{Set}$	ADJ	1975
Abstract syntax with binding	$\mathbf{Set}^{\mathbb{F}}$	Fiore,Plotkin,Turi Hofmann	1999
Simply-typed syn. with binding	$(\mathbf{Set}^{\mathbb{F}\downarrow U})^{U}$	Fiore	2003
Polymorphic Abstract Syntax	$\mathbf{Set}^{\int \mathbf{G}}$	Hamana	2011
Dependently-sorted syntax	$(\mathbf{Set}^{\mathbf{Fin}[\mathbb{S},\mathbf{Set}]})^{\mathbb{S}}$	Fiore	2008

I. Abstract Syntax with Binding

- Aim: To model syntax with variable binding, e.g.

$$\frac{}{x_1, \ldots, x_n \vdash x_i} \qquad \frac{x_1, \ldots, x_n \vdash t \quad x_1, \ldots, x_n \vdash s}{x_1, \ldots, x_n \vdash t@s}$$

$$\frac{x_1, \ldots, x_n, x_{n+1} \vdash t}{x_1, \ldots, x_n \vdash \lambda(x_{n+1}.t)}$$

- Syntax generated by 3 constructors
- λ is a special unary function symbol: decreases the context

I. Abstract Syntax with Binding

- Aim: model syntax with variable binding, e.g.

$$\frac{}{x_1, \ldots, x_n \vdash x_i} \qquad \frac{x_1, \ldots, x_n \vdash t \quad x_1, \ldots, x_n \vdash s}{x_1, \ldots, x_n \vdash t@s}$$

$$\frac{x_1, \ldots, x_n, x_{n+1} \vdash t}{x_1, \ldots, x_n \vdash \lambda(x_{n+1}.t)}$$

- Category $\mathbb{F}$ for contexts

  objects: $n = \{1, \ldots, n\}$ (contexts)

  arrows: all functions $n \longrightarrow n'$ (renamings)

- Endofunctor $\Sigma_\lambda : \mathbf{Set}^{\mathbb{F}} \longrightarrow \mathbf{Set}^{\mathbb{F}}$

$$\Sigma_\lambda(A) = \mathbf{V} + A \times A + \delta A$$

Context extension $\delta A(n) = A(n+1)$

I. Abstract Syntax with Binding

- Aim: model syntax with variable binding, e.g.

$$\frac{}{x_1, \ldots, x_n \vdash x_i} \qquad \frac{x_1, \ldots, x_n \vdash t \quad x_1, \ldots, x_n \vdash s}{x_1, \ldots, x_n \vdash t@s}$$

$$\frac{x_1, \ldots, x_n, x_{n+1} \vdash t}{x_1, \ldots, x_n \vdash \lambda(x_{n+1}.t)}$$

- Category $\mathbb{F}$ for contexts
  objects: $n = \{1, \ldots, n\}$ (contexts)
  arrows: all functions $n \longrightarrow n'$ (renamings)

- Endofunctor $\Sigma_\lambda : \mathbf{Set}^{\mathbb{F}} \longrightarrow \mathbf{Set}^{\mathbb{F}}$

$$\Sigma_\lambda(A) = \mathbf{V} + A \times A + \delta A$$

- Thm. Initial Σ_λ-algebra characterises λ-terms

II. Typed Abstract Syntax with Binding

▷ Aim: To model simply-typed syntax with variable binding, e.g.

$$\frac{x : \tau \in \Gamma}{\Gamma \vdash x : \tau} \qquad \frac{\Gamma \vdash t : \sigma \Rightarrow \tau \quad \Gamma \vdash s : \sigma}{\Gamma \vdash t@s : \tau}$$

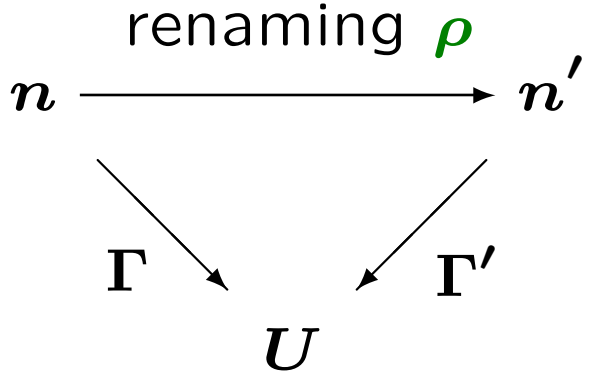
$$\frac{\Gamma, x : \sigma \vdash t : \tau}{\Gamma \vdash \lambda(x{:}\sigma.t) : \sigma \Rightarrow \tau}$$

▷ Need: typed contexts $\Gamma = \{x_1 : \tau_1, \cdots, x_n : \tau_n\}$

▷ Category $(\mathbb{F} \downarrow U)$ [Fiore'02, Miculan, Scagnetto'03]

objects: $\Gamma : n \longrightarrow U$ (contexts)

arrows:



U: set of all types

"preserves types"

II. Typed Abstract Syntax with Binding

$$\frac{x : \tau \in \Gamma}{\Gamma \vdash x : \tau} \qquad \frac{\Gamma \vdash t : \sigma \Rightarrow \tau \quad \Gamma \vdash s : \sigma}{\Gamma \vdash t@s : \tau}$$

$$\frac{\Gamma, x : \sigma \vdash t : \tau}{\Gamma \vdash \lambda(x{:}\sigma.t) : \sigma \Rightarrow \tau}$$

- ▷ Category of discourse: $(\mathbf{Set}^{\mathbb{F}\downarrow U})^U$
  - $\mathbb{F}\downarrow U$: contexts Γ
  - U: target types τ
- ▷ Endofunctor $\Sigma_{\vec{\lambda}} : (\mathbf{Set}^{\mathbb{F}\downarrow U})^U \longrightarrow (\mathbf{Set}^{\mathbb{F}\downarrow U})^U$
- ▷ Thm. Initial algebra characterises well-typed terms

Second-order Polymorphic Abstract Syntax

Examples: System F, variants of F, etc.

System F

Types

$$\tau ::= \alpha \mid b \mid \tau_1 \Rightarrow \tau_2 \mid \forall \alpha.\tau$$

$\alpha \cdots$ type variables $\quad b \cdots$ base types

Well-formed types

$$\frac{1 \leq i \leq n}{\alpha_1, \ldots, \alpha_n \vdash \alpha_i} \qquad \frac{}{\alpha_1, \ldots, \alpha_n \vdash b}$$

$$\frac{\alpha_1, \ldots, \alpha_n \vdash \sigma \quad \alpha_1, \ldots, \alpha_n \vdash \tau}{\alpha_1, \ldots, \alpha_n \vdash \sigma \Rightarrow \tau} \qquad \frac{\alpha_1, \ldots, \alpha_n, \alpha_{n+1} \vdash \tau}{\alpha_1, \ldots, \alpha_n \vdash \forall \alpha_{n+1}.\tau}$$

Well-typed terms

$$\frac{\Xi, \alpha \mid \Gamma \vdash t : \tau}{\Xi \mid \Gamma \vdash \Lambda\alpha.\, t : \forall\alpha.\, \tau} \qquad \frac{\Xi \mid \Gamma \vdash t : \forall\alpha.\, \tau \quad \Xi \vdash \sigma}{\Xi \mid \Gamma \vdash t\,\sigma : \tau[\alpha := \sigma]}$$

$$\ldots$$

Notes

- $\Xi = \alpha_1, \ldots, \alpha_n$ is a *type context*
- $\Gamma = x_1 : \tau_1, \ldots, x_k : \tau_k$ is a *term context*
- Well-formedness

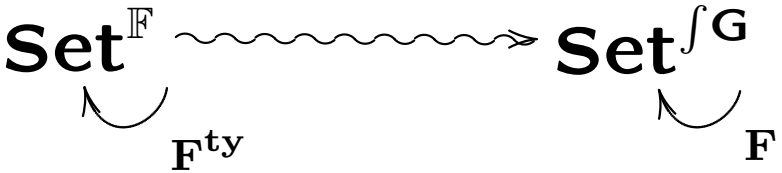
$$\Xi \mid \Gamma \vdash t : \tau \quad \text{is well-formed}$$
$$\Leftrightarrow \Xi \vdash \Gamma \quad \text{and} \quad \Xi \vdash \tau$$

How to Formulate Syntax of $\mathbf{F}$

- Three steps

(2) Contexts

$$\mathbf{G}(n) = \mathbb{F}{\downarrow}(\mathbb{T}(n)) \times \mathbb{T}(n)$$



(1) Types $\mathbb{T}$

(3) $\mathbf{T}$:System $\mathbf{F}$ terms

(1) Polymorphic Types

$$\frac{1 \leq i \leq n}{\alpha_1, \ldots, \alpha_n \vdash \alpha_i} \qquad \frac{}{\alpha_1, \ldots, \alpha_n \vdash b}$$

$$\frac{\alpha_1, \ldots, \alpha_n \vdash \sigma \quad \alpha_1, \ldots, \alpha_n \vdash \tau}{\alpha_1, \ldots, \alpha_n \vdash \sigma \Rightarrow \tau} \qquad \frac{\alpha_1, \ldots, \alpha_n, \alpha_{n+1} \vdash \tau}{\alpha_1, \ldots, \alpha_n \vdash \forall \alpha_{n+1}. \tau}$$

- Aim: Construct types $\mathbb{T} \in \mathbf{Set}^{\mathbb{F}}$
- Endofunctor $\mathbf{F}^{\mathbf{ty}} : \mathbf{Set}^{\mathbb{F}} \longrightarrow \mathbf{Set}^{\mathbb{F}}$ for types by

$$\mathbf{F}^{\mathbf{ty}}(A) = \mathbb{V} + \mathbf{B} + A \times A + \delta A.$$

- Initial $\mathbf{F}^{\mathbf{ty}}$-algebra $(\mathbb{T}, \mathsf{in})$

$$\mathbb{T}(n) = \{\tau \mid \alpha_1, \ldots, \alpha_n \vdash \tau\}$$

(2) Contexts

- Strategy to model terms: define the category, similar to

$$(\mathbf{Set}^{\mathbb{F}\downarrow U})^{U} \simeq \mathbf{Set}^{(\mathbb{F}\downarrow U)\times U}$$

- Choose $U \in \mathbf{Set}$ in terms of $\mathbb{T} \in \mathbf{Set}^{\mathbb{F}}$
- Attempt 1: $U = \coprod_{n\in\mathbb{N}} \mathbb{T}(n)$

$$\mathbf{Set}^{(\mathbb{F}\downarrow \coprod_{n\in\mathbb{N}} \mathbb{T}(n))\times \coprod_{n\in\mathbb{N}} \mathbb{T}(n)}$$

- But the index n does not synchronize
- Should be equal: (modelling $\Xi = n$)

$$\Xi \mid \Gamma \vdash t : \tau \quad \text{is well-formed}$$
$$\Leftrightarrow \Xi \vdash \Gamma \quad \text{and} \quad \Xi \vdash \tau$$

(2) Contexts

- Strategy: choose something similar to

$$(\mathbf{Set}^{\mathbb{F}\downarrow U})^{U} \simeq \mathbf{Set}^{(\mathbb{F}\downarrow U)\times U}$$

- Attempt 2:

$$\mathbf{Set}^{\coprod_{n\in\mathbb{N}}(\mathbb{F}\downarrow\mathbb{T}(n)\times\mathbb{T}(n))}$$

- Insufficient: this does not model renaming between two terms in different type contexts n and n'.

(2) Contexts

- Strategy: choose something similar to

$$(\mathbf{Set}^{\mathbb{F}\downarrow U})^{U} \simeq \mathbf{Set}^{(\mathbb{F}\downarrow U)\times U}$$

- Right way: the Grothendieck construction

$$\mathbf{Set}^{\int \mathbf{G}}$$

- where $\mathbf{G} : \mathbb{F}^{\mathrm{op}} \longrightarrow \mathbf{Cat}$ is a functor

$$\mathbf{G}(n) \stackrel{def}{=} \mathbb{F}\downarrow(\mathbb{T}(n)) \times \mathbb{T}(n)$$

parameterised by a type context n

The Grothendieck Construction

▷ A construction from

an Indexed category to a Category

$$\mathcal{F} : \mathcal{C}^{\mathrm{op}} \longrightarrow \mathbf{Cat} \quad \longmapsto \quad \int \mathcal{F} \quad \in \mathbf{Cat}$$

▷ Idea: Construct a single category
to "glue" all indexed categories together, like

$$\begin{aligned} \int \mathcal{F} \quad &\approx \quad \sum_{I \in \mathcal{C}} \mathcal{F}(I) \;=\; \mathcal{F}(I_1) + \mathcal{F}(I_2) + \cdots \\ &= \quad \{(I, A) \mid I \in \mathcal{C},\ A \in \mathcal{F}(I)\} \end{aligned}$$

The Grothendieck Construction

▷ A construction from

an Indexed category to a Category

$$\mathcal{F} : \mathcal{C}^{\mathrm{op}} \longrightarrow \mathbf{Cat} \quad \longmapsto \quad \int \mathcal{F} \quad \in \mathbf{Cat}$$

▷ Def. The Grothendieck construction of $\mathcal{F}$ is a category $\int \mathcal{F}$

- objects: (I, A) where $I \in \mathcal{C}$ and $A \in \mathcal{F}(I)$,
- arrows: $(u, \gamma) : (I, A) \longrightarrow (J, B)$ where
  - $u : I \longrightarrow J$ in $\mathcal{C}$ and
  - $\gamma : \mathcal{F}(u)(A) \longrightarrow B$ in $\mathcal{F}(J)$.

Category of Context-with-Types

- $\mathbf{G} : \mathbb{F}^{\mathrm{op}} \longrightarrow \mathbf{Cat}\,; \quad \mathbf{G}(n) \stackrel{def}{=} \mathbb{F}\downarrow(\mathbb{T}(n)) \times \mathbb{T}(n)$
- Category $\int \mathbf{G}$
  - objects $(n \mid \Gamma \vdash \tau)$
  - arrows $(\rho, \pi) : (m \mid \Gamma \vdash \tau) \longrightarrow (n \mid \Delta \vdash \sigma)$
    - renaming $\rho : m \longrightarrow n$ between type contexts
    - renaming $\pi : (\mathbb{F}\downarrow\mathbb{T}\rho)\,(\Gamma) \longrightarrow \Delta$ between term contexts

    where $\mathbb{T}(\rho)(\tau) = \sigma$
- Gives correct renaming between type and term contexts
- Gives correct renaming between term judgments
- Grothendieck is quite right!

(3) Terms of System $\mathbf{F}$

- Endofunctor $\mathbf{F} : \mathbf{Set}^{\int \mathbf{G}} \to \mathbf{Set}^{\int \mathbf{G}}$ for terms

$$
\begin{aligned}
\mathbf{F}(A)(n \mid \Gamma \vdash \tau) = \quad & \mathbf{V}(n \mid \Gamma \vdash \tau) \\
& + \coprod_{\tau_1, \tau_2 \in \mathbb{T}(n)} (\tau \equiv \tau_1 \Rightarrow \tau_2) \times A(n \mid \Gamma, \tau_1 \vdash \tau_2) \\
& + \coprod_{\sigma \in \mathbb{T}(n)} A(n \mid \Gamma \vdash \sigma \Rightarrow \tau) \times A(n \mid \Gamma \vdash \sigma) \\
& + \coprod_{\tau' \in \mathbb{T}(n+1)} (\tau \equiv \forall(\alpha.\tau')) \times A(n+1 \mid \mathsf{wk}(\Gamma) \vdash \tau') \\
& + \coprod_{\substack{\sigma \in \mathbb{T}(n) \\ \tau' \in \mathbb{T}(n+1)}} (\tau \equiv \tau'[\alpha := \sigma]) \times A(n \mid \Gamma \vdash \forall(\alpha.\tau'))
\end{aligned}
$$

- Dependent polynomial functor [Gambino,Hyland,Kock]
- Preserves ω-colimits

(3) Terms of System $\mathbf{F}$

- Endofunctor $\mathbf{F} : \mathbf{Set}^{\int \mathbf{G}} \longrightarrow \mathbf{Set}^{\int \mathbf{G}}$ for terms

$$
\begin{aligned}
\mathbf{F}(A)(n \mid \Gamma \vdash \tau) = \quad & \mathbf{V}(n \mid \Gamma \vdash \tau) \\
& + \coprod_{\tau_1, \tau_2 \in \mathbb{T}(n)} (\tau \equiv \tau_1 \Rightarrow \tau_2) \times A(n \mid \Gamma, \tau_1 \vdash \tau_2) \\
& + \coprod_{\sigma \in \mathbb{T}(n)} A(n \mid \Gamma \vdash \sigma \Rightarrow \tau) \times A(n \mid \Gamma \vdash \sigma) \\
& + \coprod_{\tau' \in \mathbb{T}(n+1)} (\tau \equiv \forall(\alpha.\tau')) \times A(n+1 \mid \mathsf{wk}(\Gamma) \vdash \tau') \\
& + \coprod_{\substack{\sigma \in \mathbb{T}(n) \\ \tau' \in \mathbb{T}(n+1)}} (\tau \equiv \tau'[\alpha := \sigma]) \times A(n \mid \Gamma \vdash \forall(\alpha.\tau'))
\end{aligned}
$$

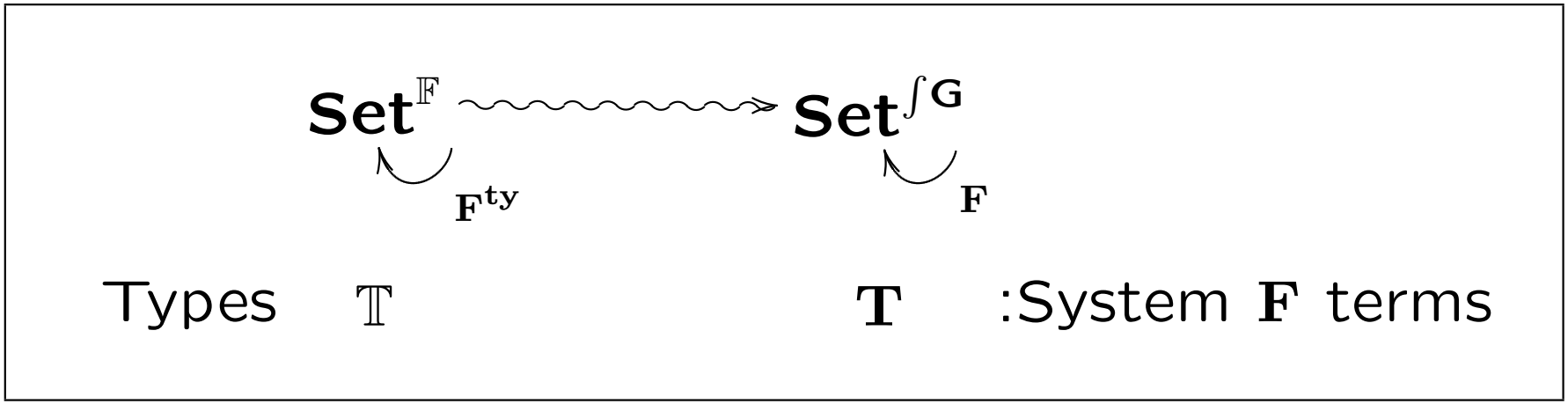
- Theorem

  All system $\mathbf{F}$-terms $\mathbf{T}$ forms an initial $\mathbf{F}$-algebra.

- Can be generic

Summary

- Initial algebra characterisations of polymorphic syntax



  - The Grothendieck construction is a key

Future Work

- Applications
- Dependent type definition in Agda, Coq