

Equational logic for higher-order abstract syntax

Alexander Kurz, Daniela Petrişan

University of Leicester

23rd September 2008

Domains IX

Overview

- ▶ Motivation

- ▷ Functors with finitary presentations

- ▷ Equational logic for higher-order abstract syntax

- ▷ Connection with nominal sets

Motivation

- Syntax with variable binders cannot be captured as an initial algebra for functors on Set.
- But we can do it if we move to functors on a presheaf category (Fiore, Plotkin and Turi).
- In particular, the lambda terms up to α -equivalence form an initial algebra for a functor.

Approach

- Functors with finitary presentation.
 - Introduced by Bonsangue and Kurz in coalgebraic logic.
 - Give rise to adequate logics for coalgebras:

$$L \left(\mathcal{A} \rightleftarrows \mathcal{X} \right) T$$

- Moving to many-sorted varieties is necessary in certain situations
- Another application: modularity! How can we describe logics for $T_1 \circ T_2$ -coalgebras?

Approach

- Functors with finitary presentation.
 - Introduced by Kurz and Bonsangue in coalgebraic logic.
 - Give rise to adequate logics for coalgebras:

$$L \left(\mathcal{A} \begin{array}{c} \longleftarrow \\ \longleftarrow \\ \longrightarrow \\ \longrightarrow \end{array} \mathcal{X} \right) T$$

- Moving to many-sorted varieties is necessary in certain situations
 - Another application: modularity! How can we describe logics for $T_1 \circ T_2$ -coalgebras?

Overview

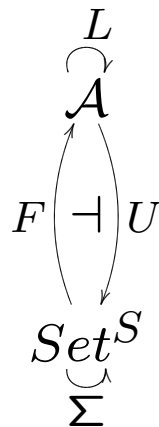
▷ Motivation

▶ **Functors with finitary presentations**

▷ Equational logic for higher-order abstract syntax

▷ Connection with nominal sets

Finitary presentations for functors



$$LA \cong F\Sigma UA/E$$

where $E \subseteq (M\Sigma MV)^2$ and M is the monad UF

Two results

A characterization theorem(Kurz and Rosický)

L has a finitary presentation by operations and equations if and only if L preserves sifted colimits.

Alg(L) as an equational class

Let $\mathcal{A} = \text{Alg}(\Sigma_{\mathcal{A}}, E_{\mathcal{A}})$ be an S -sorted variety and let $L : \mathcal{A} \rightarrow \mathcal{A}$ be a functor presented by operations Σ_L and equations E_L . Then $\text{Alg}(L) \cong \text{Alg}(\Sigma_{\mathcal{A}} + \Sigma_L, E_{\mathcal{A}} + E_L)$.

Two results

A characterization theorem(Kurz and Rosický)

L has a finitary presentation by operations and equations if and only if L preserves sifted colimits.

$\text{Alg}(L)$ as an equational class

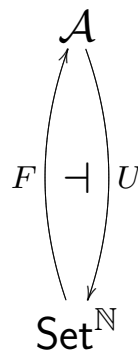
Let $\mathcal{A} = \text{Alg}(\Sigma_{\mathcal{A}}, E_{\mathcal{A}})$ be an S -sorted variety and let $L : \mathcal{A} \rightarrow \mathcal{A}$ be a functor presented by operations Σ_L and equations E_L . Then $\text{Alg}(L) \cong \text{Alg}(\Sigma_{\mathcal{A}} + \Sigma_L, E_{\mathcal{A}} + E_L)$.

Overview

- ▷ Motivation
- ▷ Functors with finitary presentations
- ▶ Equational logic for higher-order abstract syntax
- ▷ Connection with nominal sets

The functor-category $\text{Set}^{\mathbb{F}}$

- \mathbb{F} is the full subcategory of Set with objects $\underline{n} = \{1, \dots, n\}$ and $\underline{0} = \emptyset$.
- $\mathcal{A} = \text{Set}^{\mathbb{F}}$ is a many-sorted unary variety:
 - the sorts: objects of \mathbb{F}
 - operation symbols: morphisms of \mathbb{F}
 - equations: $h(x) = f(g(x))$ (when equality holds in \mathbb{F}) or $id_n(x) = x$



A suitable functor to describe the presheaf of λ -terms

- A coproduct structure on \mathbb{F}

$$\begin{array}{ccc} & & \underline{1} \\ & & \downarrow \text{new} \\ \underline{n} & \xrightarrow{i} & \underline{n + 1} \end{array}$$

where i is the inclusion and $\text{new}(1) = n + 1$.

- The type constructor $\delta : \mathcal{A} \rightarrow \mathcal{A}$ for context extension:

$$\delta(A)(\rho) = A(\rho + id_1) \quad \forall A \in \mathcal{A} \quad \forall \rho \in \mathbb{F}^{Morph}$$

- Let $L : \mathcal{A} \rightarrow \mathcal{A}$ be the functor given by

$$LX = \delta X + X \times X$$

The algebraic structure of ΛV_α

- For an arbitrary presheaf of variables V , the α -equivalence classes of λ -terms over V form a presheaf in $\text{Set}^{\mathbb{F}}$: ΛV_α .

Theorem.(Fiore, Plotkin, Turi) ΛV_α is the free L -algebra on the presheaf of variables V

- But $\text{Alg}(L)$ is an equational class, and a presentation can be obtained from:
 - an equational presentation of \mathcal{A} and
 - a finitary presentation of L .

An equational presentation for \mathcal{A} : the signature

We consider the following **operation symbols**:

$$\Sigma_{\mathcal{A}} = \{\sigma_n^{(i)} \mid 1 < n, 1 \leq i < n\} \cup \{w_n \mid n \geq 0\} \cup \{c_n \mid n > 0\} \cup \{\sigma_0\}$$

with the intended interpretation:

$\sigma_n^{(i)}$ - the transposition $(i, i + 1)$ of the set \underline{n} ,

c_n - a contraction $c_n : \underline{n + 1} \rightarrow \underline{n}$, given by

$$c_n(i) = i \quad \forall i \leq n, \quad c_n(n + 1) = n$$

w_n - the inclusion of \underline{n} into $\underline{n + 1}$.

σ_0 - the empty function.

An equational presentation for \mathcal{A} : the equations $E_{\mathcal{A}}$ (1)

-the equations coming from the presentation of the **symmetric group**:

$$\begin{aligned}(\sigma_n^{(i)})^2(x) &= id_n(x) & 1 \leq i < n \\ \sigma_n^{(i)} \sigma_n^{(j)}(x) &= \sigma_n^{(j)} \sigma_n^{(i)}(x) & j \neq i \pm 1; 1 \leq i, j < n \\ (\sigma_n^{(i)} \sigma_n^{(i+1)})^3(x) &= id_n(x) & 1 \leq i < n - 1\end{aligned} \quad (E_1)$$

An equational presentation for \mathcal{A} : the equations $E_{\mathcal{A}}$ (2)

-and some **extra** equations:

$$c_n \sigma_{n+1}^{(n)}(y) = c_n(y) \quad (\text{E}_2)$$

$$c_n w_n(x) = id_n(x) \quad (\text{E}_3)$$

$$\sigma_{n+1}^{(i)} w_n(x) = w_n \sigma_n^{(i)}(x) \quad 1 \leq i < n \quad (\text{E}_4)$$

$$\sigma_{n+2}^{(n+1)} w_{n+1} w_n(x) = w_{n+1} w_n(x) \quad (\text{E}_5)$$

$$\sigma_n^{(i)} c_n(y) = c_n \sigma_{n+1}^{(i)}(y) \quad i < n - 1 \quad (\text{E}_6)$$

$$c_n \sigma_{n+1}^{(n-1)} \sigma_{n+1}^{(n)} w_n(x) = \sigma_n^{(n-1)} w_{n-1} c_{n-1}(x) \quad (\text{E}_7)$$

$$c_n c_{n+1} \sigma_{n+2}^{(n)} = c_n c_{n+1} \quad (\text{E}_8)$$

$$((2, n-1)(1, n) w_{n-1} c_{n-1})^2 = (w_{n-1} c_{n-1} (2, n-1)(1, n))^2 \quad (\text{E}_9)$$

(E₉) comes from the presentation of the monoid of functions on \underline{n} , given by Aizenstat.

A finitary presentation for L

- **The operation symbols:** lam_n, app_n for each $n \in \mathbb{N}$; (semantically they correspond to λ -abstraction and application).
- The respective signature functor $\Sigma_L : \text{Set}^{\mathbb{N}} \rightarrow \text{Set}^{\mathbb{N}}$ is given by:

$$(\Sigma_L X)_m = \{lam_{m+1}\} \times X_{m+1} + \{app_m\} \times X_m \times X_m$$

- For any presheaf $V \in \mathcal{A}$ let $\rho_V : \Sigma UV \rightarrow ULV$ be the map defined by

$$lam_{n+1}(t) \mapsto t \quad \forall t \in V(n+1) = (\delta V)(n)$$

$$app_n(t_1, t_2) \mapsto (t_1, t_2) \quad \forall t_1, t_2 \in V(n)$$

A finitary presentation for L - the equations

- The equations E_L should correspond to the kernel pair of the adjoint transpose $\rho_V^\sharp : F\Sigma U FV \rightarrow LV$.

$$\sigma_n^{(i)} lam_{n+1}(t) = lam_{n+1}(\sigma_{n+1}^{(i)} t) \quad [t]$$

$$w_n lam_{n+1}(t) = lam_{n+2}(\sigma_{n+2}^{(n+1)} w_{n+1} t) \quad [t]$$

$$c_n lam_{n+2}(t') = lam_{n+1}(\sigma_{n+1}^{(n)} c_{n+1} \sigma_{n+2}^{(n)} \sigma_{n+2}^{(n+1)} t') \quad [t']$$

$$\sigma_n^{(i)} app_n(t_1, t_2) = app_n(\sigma_n^{(i)} t_1, \sigma_n^{(i)} t_2) \quad [t_1, t_2]$$

$$w_n app_n(t_1, t_2) = app_{n+1}(w_n t_1, w_n t_2) \quad [t_1, t_2]$$

$$c_n app_{n+1}(t_1, t_2) = app_n(c_n t_1, c_n t_2) \quad [t_1, t_2]$$

Representing different implementations of λ -terms

- If V is the presheaf defined by $V(\rho) = \rho$ for all morphisms ρ in \mathbb{F} , the free L -algebra over V gives an implementation of λ -terms by the De Bruijn levels method.
- How can we obtain different implementations for λ -terms?
- One possible approach: equip \mathbb{F} with different coproduct structures!
- But this implies working with a different functor than L .
- Let's keep L and use different presheaves of variables!

Representing different implementations of λ -terms

- If V is the presheaf defined by $V(\rho) = \rho$ for all morphisms ρ in \mathbb{F} , the free L -algebra over V gives an implementation of λ -terms by the De Bruijn levels method.
- How can we obtain different implementations for λ -terms?
- One possible approach: equip \mathbb{F} with different coproduct structures!
- But this implies working with a different functor than L .
- Let's keep L and use different presheaves of variables!

Overview

- ▷ Motivation
- ▷ Functors with finitary presentations
- ▷ Equational logic for higher-order abstract syntax
- ▶ Connection with nominal sets

Connection with nominal sets

- Replace \mathbb{F} with \mathbb{I}
- Replace \mathbb{F} with \mathbb{S}

objects: finite subsets of a countable set of atoms \mathbb{A}

morphisms: injective maps

- An equational presentation for $\text{Set}^{\mathbb{S}}$:

- operation symbols: $(a, b)_S : S \cup \{a\} \rightarrow S \cup \{b\}$ and $w_{S,c} : S \rightarrow S \cup \{c\}$, for $S \subseteq_{fin} \mathbb{A}$ and $a, b, c \notin S$

- equations:
- $$(b, a)(a, b) = id : S \cup \{a\}$$
- $$(a, b)(c, d) = (c, d)(a, b) : S \cup \{b, d\}$$
- $$(b, c)(a, b) = (a, c) : S \cup \{c\}$$
- $$(a, b)w_c = w_c(a, b) : S \cup \{c, b\}$$
- $$(a, b)w_a = w_b : S \cup \{b\}$$
- $$w_a w_b = w_b w_a : S \cup \{a, b\}$$

Syntactical differences

- A new signature: $lam_{S,a} : S \cup \{a\} \rightarrow S$, $app_S : S \times S \rightarrow S$ and $v_a : \rightarrow \{a\}$ for all $a \in \mathbb{A}$, $S \subset_{fin} \mathbb{A}$.

- A slightly different type constructor $\delta : \text{Set}^{\mathbb{S}} \rightarrow \text{Set}^{\mathbb{S}}$

- Equations:

$$t : S \cup \{a, c\} \vdash (b, c)lam_a(t) = lam_a((b, c)t) : S \cup \{b\}$$

$$t : S \cup \{a\} \vdash w_b lam_a(t) = lam_a(w_b t) : S \cup \{b\}$$

$$t_1, t_2 : S \vdash w_b app(t_1, t_2) = app(w_b t_1, w_b t_2) : S \cup \{b\}$$

$$t_1, t_2 : S \cup \{a\} \vdash (a, b)app(t_1, t_2) = app((a, b)t_1, (a, b)t_2) : S \cup \{b\}$$

$$\vdash (a, b)v_a = v_b$$

$\alpha\beta\eta$ -equivalence

$$\begin{array}{ll} \alpha : S \vdash & app(lam_a(w_S v_a), \alpha) = \alpha : S \\ \alpha : S; \beta : S \vdash & app(lam_a(w_a \beta), \alpha) = \beta : S \\ \alpha, \beta : S \cup \{a\}; \gamma : S \vdash & app(lam_a(app(\alpha, \beta)), \gamma) = \\ & app(app(lam_a(\alpha), \gamma), app(lam_a(\beta), \gamma)) : S \\ \alpha : S \cup \{a, b\}; \beta : S \vdash & app(lam_a(lam_b(\alpha)), \beta) = \\ & lam_b(app(lam_a(\alpha), w_a \beta)) : S \\ \alpha : S \cup \{a\} \vdash & app(w_b lam_a(\alpha), w_S v_b) = (a, b)\alpha : S \cup \{b\} \\ \alpha : S; \beta : a \vdash & lam_a(app(w_a \alpha, w_S v_a)) = \alpha : S \end{array}$$

Thank you!