Complete Algebraic Semantics for Second-Order Rewriting Systems based on Abstract Syntax with Variable Binding

Makoto Hamana

Faculty of Informatics, Gunma University, Japan

SYCO 10

19th December, 2022, Edinburgh
This Talk

- Complete algebraic semantics of second-order rewriting
This Talk

- **Complete algebraic semantics** of second-order rewriting
- Based on my paper
  - Complete Algebraic Semantics for Second-Order Rewriting Systems based on Abstract Syntax with Variable Binding
  - MSCS, CUP, 2022,
  - Special Issue of John Power Festschrift
First-order Rewriting: Review

First-Order Term Rewriting System (TRS) $\mathcal{R}$:

- $\text{fact}(0) \rightarrow S(0)$
- $\text{fact}(S(x)) \rightarrow \text{fact}(x) \ast S(x)$

**Rewrite steps:**

- $\text{fact}(S(S(0))) \Rightarrow \text{fact}(S(0)) \ast S(S(0)) \Rightarrow (\text{fact}(0) \ast S(0)) \ast S(S(0))$
- $\Rightarrow (S(0) \ast S(0)) \ast S(S(0)) \Rightarrow S(S(0))$ (normal form)

**Fundamental problem**

- **Termination** (Strong Normalisation)
- How can we prove the termination of $\mathcal{R}$?
Thm. [Huet and Lankford’78]

A first-order term rewriting system $\mathcal{R}$ is terminating

$\iff$

there exists a well-founded monotone $\Sigma$-algebra $(A, >_A)$ that is compatible with $\mathcal{R}$.

**Termination proof method**

$[\Leftarrow]$ Find a well-founded monotone $\Sigma$-algebra that is compatible with $\mathcal{R}$. 
First-order Rewriting: Review

First-Order Term Rewriting System (TRS) $\mathcal{R}$:

$$\begin{align*}
\text{fact}(0) & \rightarrow S(0) \\
\text{fact}(S(x)) & \rightarrow \text{fact}(x) \ast S(x)
\end{align*}$$

**Semantics:** well-founded monotone $\Sigma$-algebra $(\mathbb{N}, >)$ given by

\[\begin{align*}
\text{fact}^N(x) &= 2x + 2 \\
x \ast^N y &= x + y \\
S^N(x) &= 2x + 1 \\
0^N &= 1
\end{align*}\]

Then it is compatible with $\mathcal{R}$ as

\[\begin{align*}
\text{fact}^N(0^N) &= 2 + 2 \\
\text{fact}^N(S^N(x)) &= 2(2x + 1) + 2 \\
&> 2x + 2x + 1 = \text{fact}^N(x) \ast S^N(x)
\end{align*}\]

Hence $\mathcal{R}$ is terminating.
Aim: Sound and Complete Algebraic Characterisation

Thm. [Huet and Lankford’78]

A first-order term rewriting system $\mathcal{R}$ is terminating

$\iff$

there exists a well-founded monotone $\Sigma$-algebra $A$ that is compatible with $\mathcal{R}$.

▷ Aim: Extend this to second-order rewriting

▷ Give: Complete algebraic semantics of second-order rewriting
Example of Second-Order Rewriting: Prenex normal forms

\[ P \land \forall(x. Q[x]) \rightarrow \forall(x. P \land Q[x]) \]
\[ \neg \forall(x. Q[x]) \rightarrow \exists(x. \neg(Q[x])) \]
\[ \forall(x. Q[x]) \land P \rightarrow \forall(x. Q[x] \land P) \]
\[ \neg \exists(x. Q[x]) \rightarrow \forall(x. \neg(Q[x])) \]

Signature: \( \neg, \land, \lor, \forall, \exists \)
Example of Second-Order Rewriting: Prenex normal forms

\[\begin{align*}
P \land \forall(x. Q[x]) & \rightarrow \forall(x. P \land Q[x]) & \neg \forall(x. Q[x]) & \rightarrow \exists(x. \neg(Q[x])) \\
\forall(x. Q[x]) \land P & \rightarrow \forall(x. Q[x] \land P) & \neg \exists(x. Q[x]) & \rightarrow \forall(x. \neg(Q[x]))
\end{align*}\]

Signature: \(\neg, \land, \lor, \forall, \exists\)

Second-Order Rewriting System is defined on

Second-Order Abstract Syntax [Hamana’04, Fiore LICS’06]

▷ Abstract syntax with variable binding [Fiore, Plotkin, Turi LICS’99]

▷ Metavariables with arities (e.g. \(p, q\))

▷ Substitutions (Metavars, object vars)
Example: the $\lambda$-calculus as a Second-Order Rewriting System

\[ \lambda(x. M[x]) \ @ N \rightarrow M[N] \]
\[ \lambda(x. M \ @ x) \rightarrow M \]

▷ Signature: $\lambda, @$
Aim: To model syntax with variable binding, e.g.

\[
\begin{align*}
  x_1, \ldots, x_n & \vdash x_i \\
  x_1, \ldots, x_n & \vdash t \\
  x_1, \ldots, x_n & \vdash t \at s \\
  x_1, \ldots, x_n \vdash t
\end{align*}
\]

Syntax generated by 3 constructors

\(\lambda\) is a **special** unary function symbol:

- it decreases the context
Aim: model syntax with variable binding, e.g.

\[
\begin{align*}
& n \vdash i \\
& n \vdash t \quad n \vdash s \quad \quad \Rightarrow \quad n \vdash t \circ s \\
& n + 1 \vdash t \\
& n \vdash \lambda(n + 1.t)
\end{align*}
\]

Category \( \mathbb{F} \) for variable contexts

- objects: \( n = \{1, \ldots, n\} \) (variable contexts)
- arrows: all functions \( n \to n' \) (renamings)

Presheaf category \( \text{Set}^\mathbb{F} \)
Models of Syntax with Binding: $\Sigma$-Algebras in $\text{Set}^F$

Def. A binding signature $\Sigma$ is a set of function symbols with binding arities:

$$f : \langle n_1, \ldots, n_l \rangle$$

which has $l$ arguments and binds $n_i$ variables in the $i$-th argument.

Def. A $\Sigma$-algebra $A = (A, [f^A]_{f \in \Sigma})$ in $\text{Set}^F$ consists of

- carrier: a presheaf $A \in \text{Set}^F$
- operations: arrows of $\text{Set}^F$

$$f^A : \delta^{n_1} A \times \ldots \times \delta^{n_l} A \longrightarrow A$$

corresponding to function symbols $f : \langle n_1, \ldots, n_l \rangle \in \Sigma$.

- Context extension: $\delta A \in \text{Set}^F$; $(\delta A)(n) = A(n + 1)$
Example: $\lambda$-terms

- Binding signature $\Sigma_\lambda$ for $\lambda$-terms

  \[ \lambda : \langle 1 \rangle, \quad @ : \langle 0,0 \rangle \]

- Carrier: the presheaf $\Lambda$ of all $\lambda$-terms

  \[ \Lambda(n) = \{ t \mid n \vdash t \} \]

  \[ \Lambda(\rho) : \Lambda(m) \to \Lambda(n) \quad \text{renaming on } \lambda\text{-terms for } \rho : m \to n \text{ in } \mathbb{F}. \]

- Forms a $V + \Sigma_\lambda$-algebra

  \[
  \begin{align*}
  \text{var}^\Lambda & : V \to \Lambda \\
  @^\Lambda & : \Lambda \times \Lambda \to \Lambda \\
  \lambda^\Lambda & : \delta \Lambda \to \Lambda \\
  i & \mapsto i \\
  s , t & \mapsto s @ t \\
  \lambda^\Lambda(n) & : \Lambda(n + 1) \to \Lambda(n) \\
  t & \mapsto \lambda n + 1 . t
  \end{align*}
  \]

- Presheaf of variables: $V \in \text{Set}^\mathbb{F}$; $V(n) = \{ 1, \ldots, n \}$

- Thm. $\Lambda (= T_\Sigma V)$ is an initial $V + \Sigma_\lambda$-algebra.
Second-Order Abstract Syntax

- Abstract syntax with variable binding
- Metavariabes with arities
- Substitutions (Metavars, object vars)
A \textbf{\(\Sigma\)-monoid} [Fiore, Plotkin, Turi’99] is

- a \(\Sigma\)-algebra \(A\) with
- a monoid structure

\[
\begin{align*}
V & \xrightarrow{\nu} A \\
\mu & \quad A \bullet A
\end{align*}
\]

in the monoidal category \((\text{Set}^F, \bullet, V)\),

- both are compatible.

\textbf{Idea}

- Unit \(\nu\) models the embedding of variables
- Multiplication \(\mu\) models \textbf{substitution for object variables}
Given a binding signature $\Sigma$

- The presheaf of all $\Sigma$-terms

$$T_\Sigma V(n) = \{ t \mid n \vdash t \}$$

- Multiplication $\mu : T_\Sigma V \bullet T_\Sigma V \to T_\Sigma V$

$$\mu_n^{(m)}(t; s_1, \ldots, s_m) \triangleq t[1 := s_1, \ldots, n := s_m]$$

(the substitution of $\Sigma$-terms for de Bruijn variables)
Algebraic Characterisation of Syntax with Binding

Given a binding signature $\Sigma$

- The presheaf of all $\Sigma$-terms

$$T_\Sigma V(n) = \{ t \mid n \vdash t \}$$

- Multiplication $\mu : T_\Sigma V \bullet T_\Sigma V \rightarrow T_\Sigma V$

$$\mu^{(m)}_n(t; s_1, \ldots, s_m) \triangleq t[1 := s_1, \ldots, n := s_m]$$

(the substitution of $\Sigma$-terms for de Bruijn variables)

- Thm. [Fiore, Plotkin, Turi'99]
  - $(T_\Sigma V, \nu, \mu)$ is an initial $\Sigma$-monoid.
  - $(T_\Sigma V, \nu)$ is an initial $V + \Sigma$-algebra.

- How to model metavariables and substitutions for metavariables?
Algebraic Characterisation of Syntax with Binding

Given a binding signature $\Sigma$

- The presheaf of all $\Sigma$-terms

$$T_\Sigma V(n) = \{ t \mid n \vdash t \}$$

- Multiplication $\mu : T_\Sigma V \bullet T_\Sigma V \to T_\Sigma V$

$$\mu^{(m)}_n(t; s_1, \ldots, s_m) \triangleq t[1 := s_1, \ldots, n := s_m]$$

(the substitution of $\Sigma$-terms for de Bruijn variables)

- Thm. [Fiore, Plotkin, Turi’99]
  - $(T_\Sigma V, \nu, \mu)$ is an initial $\Sigma$-monoid.
  - $(T_\Sigma V, \nu)$ is an initial $V + \Sigma$-algebra.

How to model metavariables and substitutions for metavariables?

Free $\Sigma$-monoids [Hamana, APLAS’04]
Meta-terms: Terms with Metavariabes [Aczel '78]

- A binding signature $\Sigma$

- $Z$ is an $\mathbb{N}$-indexed set of metavariables parameterised by arities:

  $$Z(l) \triangleq \{ M \mid M^l, \text{ where } l \in \mathbb{N} \}.$$

- Raw meta-terms generated by $Z$:

  $$t ::= x \mid f(x_1 \cdots x_i, t_1, \ldots, x_1 \cdots x_i, t_l) \mid M[t_1, \ldots, t_l]$$

- A meta-term $t$ is a raw meta-term derived from:

  $$\frac{x \in n \quad f : \langle i_1, \ldots, i_l \rangle \in \Sigma \quad n+i_1 \vdash t_1 \ldots n+i_l \vdash t_l}{n \vdash x} \quad \frac{n \vdash x}{n \vdash f(n+1 \ldots n+i_1.t_1, \ldots, n+1 \ldots n+i_l.t_l)} \quad \frac{M \in Z(l) \quad n \vdash t_1 \ldots n \vdash t_l}{n \vdash M[t_1, \ldots, t_l]}$$
Meta-terms: Terms with Metavariabes

▶ Presheaf $M_\Sigma Z \in \text{Set}^\mathcal{F}$

$$M_\Sigma Z(n) = \{ t \mid n \vdash t \}$$

▶ $V + \Sigma$-algebra $(M_\Sigma Z, [\nu, f_T]_{f \in \Sigma})$

$$\nu(n) : V(n) \longrightarrow M_\Sigma Z(n),$$

$$x \longmapsto x$$

$$f^T : \delta^{i_1} M_\Sigma Z \times \cdots \times \delta^{i_l} M_\Sigma Z \longrightarrow M_\Sigma Z$$

$$(t_1, \ldots, t_l) \vdash f(n + i_1 t_1, \ldots, n + i_l t_l).$$

▶ Multiplication $\mu : M_\Sigma Z \bullet M_\Sigma Z \rightarrow M_\Sigma Z$

$$t, s \longmapsto t[1 := s_1, \ldots, n := s_n]$$

⋯ substitution of meta-terms for object variables
Thm. \((M_\Sigma Z, \nu, \mu)\) forms a free \(\Sigma\)-monoid over \(Z\).

- Freeness of \(M_\Sigma Z\): in \(\text{Set}^F\), given assignment \(\theta\)

\[
\begin{align*}
Z & \xrightarrow{\eta_Z} M_\Sigma Z \\
\downarrow \theta & \downarrow \exists! \theta^\# \Sigma\text{-monoid morphism} \\
A & \Sigma\text{-monoid}
\end{align*}
\]

- The unique \(\Sigma\)-monoid morphism \(\theta^\#\) that extends \(\theta\).
Instance: Substitution for Metavariabes

Case $A = T_{\Sigma}V$ · · · a $\Sigma$-monoid of terms,

\[
\begin{array}{c}
Z \xrightarrow{\eta_Z} M_{\Sigma}Z \\
\downarrow \quad \theta \\
\exists! \theta^\# \quad \Sigma\text{-monoid morphism} \\
T_{\Sigma}V
\end{array}
\]

- $\theta^\#$ is a substitution of terms for metavariables $Z$
- E.g. $\Sigma$: signature for $\lambda$-terms, for $\theta(M^{(1)}) = a @ a$

\[
\theta^\#( \lambda(x.M[x]@y) ) = \lambda( x.(x@x)@y )
\]

- Other examples of $\Sigma$-monoid $A$:
  - $M_{\Sigma}Z$: meta-substitution: substitution of meta-terms for metavars
  - Any $\Sigma$-monoid as a model – $\theta^\#$ is compositional interpretation
Second-Order Rewriting System

**Eg. A transformation to prenex normal forms**

\[ P \land \forall(x.Q[x]) \rightarrow \forall(x.P \land Q[x]) \quad \neg \forall(x.Q[x]) \rightarrow \exists(x.\neg(Q[x])) \]

**Def.**
Rewrite rules \( \mathcal{R} \) \( l \rightarrow r \) on meta-terms \( \mathcal{M}_\Sigma \mathcal{Z} \)

(with some syntactic conditions)

Rewrite relation \( \rightarrow_\mathcal{R} \) on terms \( \mathcal{T}_\Sigma \mathcal{V} \)

\[
\frac{l \rightarrow r \in \mathcal{R} \quad s \rightarrow_\mathcal{R} t}{\theta^\sharp(l) \rightarrow_\mathcal{R} \theta^\sharp(r) \quad f(\ldots, \overline{x}.s, \ldots) \rightarrow_\mathcal{R} f(\ldots, \overline{x}.t, \ldots)}
\]

▷ Substitution \( \theta : \mathcal{Z} \rightarrow \mathcal{T}_\Sigma \mathcal{V} \) maps metavariables to terms

▷ NB. rewriting is defined on terms (without metavariables)
Presheaf with relation \((A, >_A)\)

Def. A presheaf \(A \in \textbf{Set}^F\) is equipped with a binary relation \(>_A\), if

1. \(>_A\) is a family \(\{>_A(n)\}_{n \in F}\),
2. which is compatible with presheaf action.

(for all \(a, b \in A(m)\) and \(\rho : m \to n\) in \(F\), if \(a >_A(m) b\), then \(A(\rho)(a) >_A(n) A(\rho)(b)\).)
Monotone Algebra

Def. A **monotone** $\mathbf{V} + \Sigma$-algebra $(A, >_A)$ is a $\mathbf{V} + \Sigma$-algebra $(A, [\nu, f^A]_{f \in \Sigma})$

▷ equipped with a relation $>_A$ such that

▷ every operation $f^A$ is monotone.

**Thm.** $(T_\Sigma \mathbf{V}, \rightarrow_\mathcal{R})$ is a monotone $\mathbf{V} + \Sigma$-algebra.
Models of Rewrite System $\mathcal{R}$: $\mathcal{R}$-$\mathcal{A}$-algebras

A $(\mathcal{V} + \Sigma, \mathcal{R})$-algebra $(A, >_A)$ is a monotone $\mathcal{V} + \Sigma$-algebra satisfying all rules in $\mathcal{R}$ as:

\[
\begin{align*}
  Z & \xrightarrow{\eta_Z} M_\Sigma Z & l \rightarrow r \in \mathcal{R} & \cdots \text{a rule} \\
  \theta & \downarrow & \theta^\# & \text{a unique } \Sigma \text{-monoid mor. extends } \theta \\
  \text{initial } \mathcal{V} + \Sigma \text{-algebra} & \xrightarrow{\mathcal{T}_\Sigma \mathcal{V}} \theta^\#_n(l) \rightarrow_{\mathcal{R}} \theta^\#_n(r) & \cdots \text{a rewrite} \\
  \mathcal{V} + \Sigma \text{-algebra} & \xrightarrow{!_A} A & !_A \theta^\#_n(l) >_A !_A \theta^\#_n(l) & \cdots \text{an interpretation}
\end{align*}
\]
Soundness and Completeness of Models

Prop. \( s \xrightarrow{\mathcal{R}} t \)

\( \iff \)

\( !_A(s) >_A !_A(t) \) for all \((V + \Sigma, \mathcal{R})\)-algebras \( A \), assignments \( \theta \). 

Proof. \( [\Rightarrow] \): By induction of the proof of rewrite.
\[ [\Leftarrow] \): Take \( (\mathcal{A}, >_A) = (T_\Sigma V, \rightarrow_{\mathcal{R}}) \).
Complete Characterisation of Terminating Second-Order Rewriting

**Thm.** A second-order rewriting system $\mathcal{R}$ is terminating iff there is a well-founded $(V + \Sigma, \mathcal{R})$-algebra $(A, >_A)$.

**Proof.** ($\iff$): Suppose a well-founded $(V + \Sigma, \mathcal{R})$-algebra $(A, >_A)$.

Assume $\mathcal{R}$ is non-terminating:

$$t_1 \rightarrow_\mathcal{R} t_2 \rightarrow_\mathcal{R} \cdots$$

By soundness,

$$!_A(t_1) >_A(n) !_A(t_2) >_A \cdots$$

Contradiction.

($\Rightarrow$): When $\mathcal{R}$ is terminating, the $(V + \Sigma, \mathcal{R})$-algebra $(T_{\Sigma V}, \rightarrow_\mathcal{R})$ is a well-founded algebra.

Because of the algebraic characterisations of abstract syntax with binding [FPT’99] and meta-terms [H.04]
Application: Termination by Interpretation

\[
P \land \forall (x. Q[x]) \rightarrow \forall (x. P \land Q[x]) \quad \neg \forall (x. Q[x]) \rightarrow \exists (x. \neg (Q[x])) \\
\forall (x. Q[x]) \land P \rightarrow \forall (x. Q[x] \land P) \quad \neg \exists (x. Q[x]) \rightarrow \forall (x. \neg (Q[x]))
\]

Take a well-founded monotone \( V + \Sigma \)-algebra \( (K, >_K) \)
where \( K(n) = \mathbb{N} \) with \( >_{K(n)} = > \) on \( \mathbb{N} \).

Operations

\[
\begin{align*}
\nu^K_n(i) &= 0 \\
\wedge^K_n(x, y) &= \vee^K_n(x, y) = 2x + 2y \\
\neg^K_n(x) &= 2x \\
\forall^K_n(x) &= \exists^K_n(x) = x + 1.
\end{align*}
\]

\( (V + \Sigma, \mathcal{R}) \)-algebra

\[
!\theta^\#_0(P \land \forall (1. Q[1])) = 2x + 2(y + 1) >_{K(0)} (2x + 2y) + 1 = !\theta^\#_0(\forall (1. P \land Q[1])) \\
!\theta^\#_0(\neg \exists (1. Q[1])) = 2(y + 1) >_{K(0)} 2y + 1 = !\theta^\#_0(\forall (1. \neg (Q[1]))).
\]
Complete algebraic semantics of second-order rewriting systems

Based on my paper

- Complete Algebraic Semantics for Second-Order Rewriting Systems based on Abstract Syntax with Variable Binding
- MSCS, CUP, 2022, Special Issue of John Power Festschrift
Summary

- **Complete algebraic semantics** of second-order rewriting systems
- Based on my paper
  - Complete Algebraic Semantics for Second-Order Rewriting Systems based on Abstract Syntax with Variable Binding
  - MSCS, CUP, 2022, Special Issue of John Power Festschrift
- Thanks to John Power, Gordon Plotkin
Summary

- Complete algebraic characterisation of second-order rewriting systems
- using algebraic models of second-order abstract syntax

Further Topics and Applications

- Meta-rewriting: rewriting on meta-terms using monotone Σ-monoids
- Modularity of Termination for Second-Order rewriting [H. LMCS’21]
  - A: terminating & B terminating \( \Rightarrow \) A \( \cup \) B: terminating with several conditions
- Tool SOL for termination and confluence checking
  - 1st places in the Higher-order Category of
    - International Confluence Competition 2020
    - Termination Competition 2022

http://solweb.mydns.jp/webcui/sol/