Co/algebraic essentials

and their impact on languages and compilers

Peter Padawitz
TU Dortmund

February 3, 2011

Road map

Let S be a set of **sorts**. An S**-sorted set** A is a family $\{A_s \mid s \in S\}$ of sets.

An S**-sorted function** $f : A \to B$ is a family $\{f_s : A_s \to B_s \mid s \in S\}$ of functions.

Given $BS \subseteq S$ and a BS-sorted set BA, Set^S_{BA} denotes the category whose objects are pairs $(A, f : A|_{BS} \xrightarrow{\sim} BA)$ consisting of an S-sorted set A and a BS-sorted bijection f and whose morphisms from (A, f) to (B, g) are S-sorted functions $h : A \to B$ such that for all $s \in BS$, $g_s \circ h_s = f_s$.

For all $s_1, \ldots, s_n, s, s' \in S$,

$$\begin{aligned}
A_1 &=_{def} \{*\}, \\
A_{s_1 \times \cdots \times s_n} &=_{def} A_{s_1} \times \ldots \times A_{s_n} =_{def} \{(a_1, \ldots, a_n) \mid a_i \in A_{s_i},\ 1 \leq i \leq n\}, \\
A_{s_1 + \cdots + s_n} &=_{def} A_{s_1} + \cdots + A_{s_n} =_{def} \{(a, i) \mid a \in A_{s_i},\ 1 \leq i \leq n\}, \\
A_{s^{s'}} &=_{def} (A_{s'} \to A_s).
\end{aligned}$$

The set of **signatures** is defined inductively as follows:

- $\Sigma = (S, F)$ is a signature if S is a set of **sorts** and F is an $S^* \times S^+$-sorted set of function symbols.
- $\Sigma = (S, F, B\Sigma)$ is a signature if $B\Sigma$ is a signature, called the **base signature of** Σ, and (S, F) is a signature such that S and F contain the sorts resp. function symbols of the base signature.

$f : v \to w \in F \setminus BF$ is a **constructor** if $w \in S \setminus BS$. f is a **destructor** if $v \in S \setminus BS$.

Σ is a **constructor signature** if $F \setminus BF$ consists of constructors. S and F implicity include **sum sorts** $s_1 + \cdots + s_n$ and **injections** $\iota_i : s_i \to s_1 + \cdots + s_n$ for all $s_1, \ldots, s_n \in S$.

Σ is a **destructor signature** if $F \setminus BF$ consists of destructors. S and F implicity include **product sorts** $s_1 \times \cdots \times s_n$, **projections** $\pi_i : s_1 \times \cdots \times s_n \to s_i$, **power sorts** $s^{s'}$ and **applications** $\$a : s^{s'} \to s$ for all $s_1, \ldots, s_n, s \in S$, $s' \in BS$, $a \in BA_{s'}$ and BS-sorted sets BA.

☞ A signature for regular expressions

$$
\begin{aligned}
Reg &= (\; S, F, B\Sigma \;) \\
&= (\; \{reg, symbol\}, \\
&\qquad \{\emptyset, \; \epsilon : \epsilon \longrightarrow reg, \\
&\qquad \_ : symbol \longrightarrow reg, \\
&\qquad \_ \mid \_ : reg\ reg \longrightarrow reg, \\
&\qquad \_ \cdot \_ : reg\ reg \longrightarrow reg, \\
&\qquad star : reg \longrightarrow reg\}, \\
&\qquad (\{symbol\}, \emptyset) \;)
\end{aligned}
$$

☞ A signature for acceptors

$$
\begin{aligned}
Accept &= (\; S, F, B\Sigma \;) \\
&= (\; \{state, symbol, bool\}, \\
&\qquad \{\delta : state \longrightarrow state^{symbol}, \\
&\qquad final : state \longrightarrow bool\}, \\
&\qquad (\{symbol, bool\}, \emptyset) \;)
\end{aligned}
$$

Let $\Sigma = (S, F, B\Sigma)$ be a signature, X be an S-sorted set of **variables** and Y be an S-sorted set of **covariables**.

The S-sorted set $T_\Sigma(X)$ of **Σ-terms over** X is inductively defined as follows:

- For all $s \in S$, $X_s \subseteq T_\Sigma(X)_s$.
- For all $f : s_1 \dots s_n \to s \in F$ and $t_i \in T_\Sigma(X)_{s_i}$, $1 \leq i \leq n$, $f\langle t_1, \dots, t_n\rangle \in T_\Sigma(X)_s$.

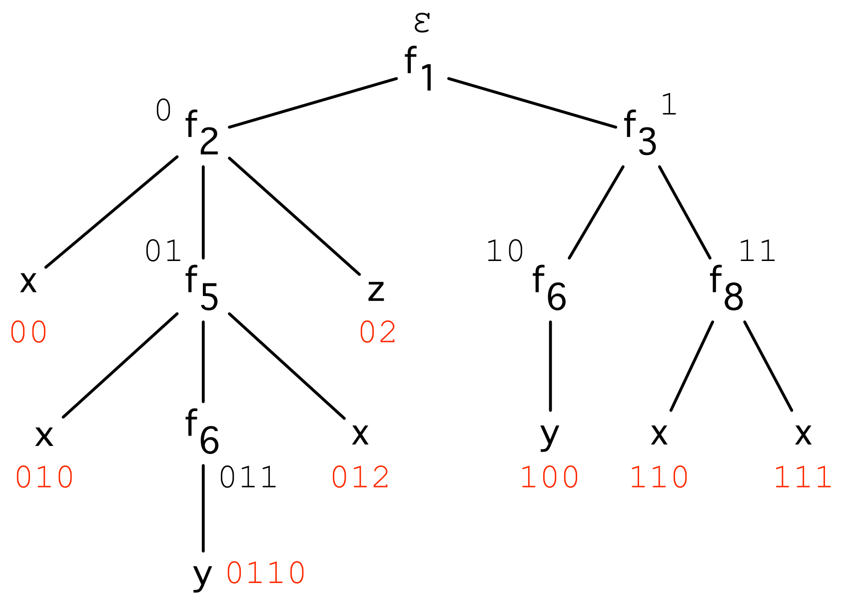
Given $t \in T_\Sigma(X)$, $var(t)$ denotes the set of variables occurring in t.

The S-sorted set $coT_\Sigma(Y)$ of **Σ-coterms over** X is inductively defined as follows:

- For all $s \in S$, $Y_s \subseteq coT_\Sigma(Y)_s$.
- For all $f : s \to s_1 \dots s_n \in F$ and $t_i \in coT_\Sigma(Y)_{s_i}$, $1 \leq i \leq n$, $[t_1, \dots, t_n]f \in coT_\Sigma(Y)_s$.

Given $t \in coT_\Sigma(Y)$, $cov(t)$ denotes the set of covariables occurring in t.

A term resp. coterm t over $\mathbb{N}^*$ such that all function symbols of t belong to $F \setminus BF$ and for all $x \in var(t) \cup cov(t)$, $sort(x) \in BS$ and $t(x) = x$, is called a **Σ-generator** resp. **Σ-observer**.



The tree representing the term $f_1\langle f_2\langle x, f_5\langle x, f_6\langle y\rangle, x\rangle, z\rangle, f_3\langle f_6\langle y\rangle, f_8\langle x, x\rangle\rangle\rangle$
or the coterm $[[[x, [x, [y]f_6, x]f_5, z]f_2, [[y]f_6, [x, x]f_8]f_3]f_1$

☞ The *Reg*-terms over X_{symbol} are the regular expressions over X_{symbol}.

☞ For each *Accept*-observer t there are $a_1, \ldots, a_n \in Z =_{def} BA_{symbol}$ such that

$$t = parse(\_, a_1 \ldots a_n) =_{def} final(\delta^*(\_, a_1 \ldots a_n))$$
$$=_{def} final(\delta(\ldots(\delta(\delta(\_, a_1), a_2), \ldots, a_n)) =_{def} [[\ldots[[[x]final]\delta\$a_n]\ldots]\delta\$a_2]\delta\$a_1.$$

Hence t is representable by $a_1 \ldots a_n \in Z^*$.

A Σ-**algebra** A consists of an S-sorted set, the **carrier** of A, also denoted by A, and

- for each $f : w \to s_1 \dots s_n \in F$, a function $f^A : A_w \to A_{s_1+\dots+s_n}$,

such that

- for all injections $\iota_i : s_i \to s_1 + \dots + s_n$, $a \in A_{s_i}$, $\iota_i^A(a) = (a, i)$,
- for all projections $\pi_i : s_1 \times \dots \times s_n \to s_i$ and $a \in A_{s_1 \times \dots \times s_n}$, $\pi_i^A(a) = a_i$,
- for all applications $\$a : s^{s'} \to s$ and $f : A_{s'} \to A_s$, $(\$a)^A(f) = f(a)$.

☞ The regular expressions over BA_{symbol} form the *Reg*-algebra $T_{Reg}(BA)$.

☞ The languages over BA_{symbol} form the *Reg*-algebra $Lang(BA)$.

☞ Let $BA_{bool} = 2$. $Lang(BA)$ is also an *Accept*-algebra: For all $L \subseteq Z^*$ and $a \in Z$,

$$\begin{aligned} Lang(BA)_{state} &=_{def} \mathcal{P}(Z^*), \\ \delta^{Lang(BA)}(L, a) &=_{def} \{w \in Z^* \mid aw \in L\}, \\ final^{Lang(BA)}(L) &=_{def} (\epsilon \in L). \end{aligned}$$

The *Accept*-subalgebra

$$\langle L \rangle =_{def} \{(\delta^*)^{Lang(BA)}(L, w) \mid w \in Z^*\}$$

is a minimal acceptor of L.

Let A and B be Σ-algebras, $h : A \longrightarrow B$ be an S-sorted function and $f \in F$.

h is a **Σ-homomorphism** if for all $f \in F$,

$$h \circ f^A = f^B \circ h.$$

Let BA be a $B\Sigma$-algebra.

A **$\Sigma{\downarrow}BA$-algebra** (A, g) is a pair consisting of a a Σ-algebra A and a $B\Sigma$-isomorphism $g : A|_{B\Sigma} \longrightarrow BA$.

Given $\Sigma{\downarrow}BA$-algebras (A, f) and (B, g), a Σ-homomorphism $h : A \longrightarrow B$ is a **$\Sigma{\downarrow}BA$-homomorphism** if $g \circ h|_{\Sigma} = f$.

$Alg_{\Sigma\downarrow BA}$ denotes the category of $\Sigma{\downarrow}BA$-algebras and $\Sigma{\downarrow}BA$-homomorphisms.

Term evaluation $\_^A : T_\Sigma(X) \to (A^X \to A)$ is inductively defined as follows:
Let $g \in A^X$.

- For all $x \in X$, $x^A(g) = g(x)$.
- For all $f : s_1 \dots s_n \to s \in F \setminus BF$ and $t_i \in T_\Sigma(X)_{s_i}$, $1 \leq i \leq n$,

$$(f\langle t_1, \dots, t_n\rangle)^A(g) = f^A(t_1^A(g), \dots, t_n^A(g)).$$

☞ For all regular expressions $R \in T_{Reg}(BA)$, $R^{Lang(BA)}(id_{BA})$ is the language of R.

Coterm evaluation $\_^A : coT_\Sigma(Y) \to (A \to A \cdot Y)$ is inductively defined as follows:

- For all $s \in S$, $x \in Y_s$ and $a \in A_s$, $x^A(a) = (a, x)$.
- For all $f : s \to s_1 \dots s_n \in F \setminus BF$, $t_i \in coT_\Sigma(X)_{s_i}$, $1 \leq i \leq n$, and $a \in A_s$,

$$f^A(a) = (b, i) \;\Rightarrow\; ([t_1, \dots, t_n]f)^A(a) = t_i^A(b).$$

☞ For all *Accept*-observers $parse(\_, a_1 \dots a_n)$ and $L \subseteq Z^*$,

$$parse(\_, a_1 \dots a_n)^{Lang(BA)}(L) \;=\; (a_1 \dots a_n \in L).$$

Signatures induce functors with fixpoints

Let BS be the sorts of $B\Sigma$ and BA be a BS-sorted set.

If Σ is a constructor signature, then Σ and BA induce the functor $\Sigma_{BA} : Set^S_{BA} \to Set^S_{BA}$:
For all $A \in Set^S_{BA}$ and $s \in S$,

$$\Sigma_{BA}(A)_s =_{def} \begin{cases} \coprod_{f:s_1\ldots s_n \to s \in F}(A_{s_1} \times \ldots \times A_{s_n}) & \text{if } s \in S \setminus BS, \\ A_s & \text{if } s \in BS. \end{cases}$$

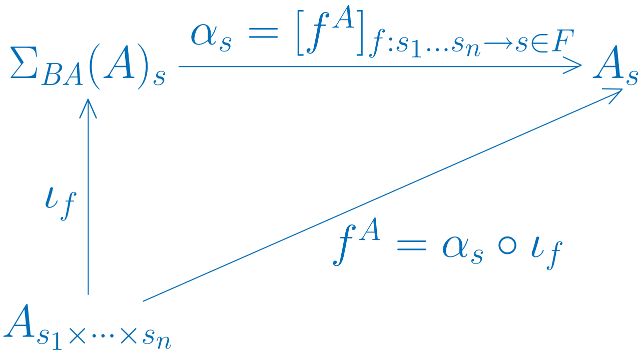
☞ $$Reg_{BA}(A)_{reg} =_{def} 1 + 1 + BA_{symbol} + A^2_{reg} + A^2_{reg} + A_{reg}.$$

If Σ is a destructor signature, then Σ and BA induce the functor $\Sigma_{BA} : Set^S_{BA} \to Set^S_{BA}$:
For all $A \in Set^S_{BA}$ and $s \in S$,

$$\Sigma_{BA}(A)_s =_{def} \begin{cases} \prod_{f:s \to s_1\ldots s_n \in F}(A_{s_1} + \cdots + A_{s_n}) & \text{if } s \in S \setminus BS, \\ A_s & \text{if } s \in BS. \end{cases}$$

☞ $$Accept_{BA}(A)_{state} =_{def} A_{state}^{BA_{symbol}} \times 2.$$

A Σ_{BA}-algebra $\Sigma_{BA}(A) \xrightarrow{\alpha} A$ is an S-sorted function and uniquely corresponds to a $\Sigma{\downarrow}BA$-algebra A: For all $s \in S \setminus BS$ and $f : s_1 \dots s_n \to s \in F \setminus BF$,



α_s is the coproduct extension

of the interpretations in A

of all constructors f of Σ

Since Σ_{BA} preserves colimits of increasing ω-chains, the category $Alg_{\Sigma_{BA}}$ of Σ_{BA}-algebras has an initial object $ini : \Sigma_{BA}(\mu\Sigma_{BA}) \xrightarrow{\sim} \mu\Sigma_{BA}$ and thus a fixpoint of Σ_{BA}.

For all $s \in S \setminus BS$, $\mu\Sigma_{BA,s} = \coprod_{t \in Gen_{\Sigma,s}} BA^{var(t)}$.

☞ The Reg-algebra $T_{Reg}(BA)$ of regular expressions over BA_{symbol} is initial in $Alg_{Reg\downarrow BA}$:

$$(\mu Reg_{BA})_{reg} = \coprod_{t \in Gen_{Reg,reg}} BA^{var(t)} = T_{Reg}(BA).$$

A Σ_{BA}-coalgebra $A \stackrel{\alpha}{\to} \Sigma^{BA}(A)$ is an S-sorted function and uniquely corresponds to a $\Sigma{\downarrow}BA$-algebra A: For all $s \in S \setminus BS$ and $f : s \to s_1 \dots s_n \in F \setminus BF$,

$$A_s \xrightarrow{\alpha_s = \langle f^A \rangle_{f:s \to s_1 \dots s_n \in F}} \Sigma_{BA}(A)_s \qquad \alpha_s \text{ is the product extension}$$

$$\Sigma_{BA}(A)_s \xrightarrow{\pi_f} A_{s_1+\dots+s_n} \qquad \text{of the interpretations in } A$$

$$f^A = \pi_f \circ \alpha_s : A_s \to A_{s_1+\dots+s_n} \qquad \text{of all destructors } f \text{ of } \Sigma$$

Since Σ_{BA} preserves limits of decreasing ω-chains, the category $coAlg_{\Sigma_{BA}}$ of Σ^{BA}-algebras has a final object $fin : \nu\Sigma_{BA} \xrightarrow{\sim} \Sigma_{BA}(\nu\Sigma_{BA})$ and thus a fixpoint of Σ_{BA}.

For all $s \in S \setminus BS$, $\nu\Sigma_{BA,s} \subseteq \prod_{t \in Obs_{\Sigma,s}} (BA \times cov(t))$.

☞ Let $BA_{bool} = 2$. The *Accept*-algebra $Lang(BA)$ of languages over BA_{symbol} is final in $Alg_{Accept\downarrow BA}$:

$$(\nu Accept_{BA})_{state} = \prod_{t \in Obs_{Accept,state}} (BA_{Bool} \times \{x\}) = \prod_{t \in Obs_{Accept,state}} BA_{Bool} = 2^{BA^*_{symbol}}$$
$$= \mathcal{P}(BA^*_{symbol}) = Lang(BA)_{state}.$$

Given $(A, g) \in Alg_{\Sigma\downarrow BA}$, the unique $\Sigma\downarrow BA$-homomorphism $fold^A : \mu\Sigma_{BA} \to A$ is defined as follows: For all $s \in S$ and $t \in T_\Sigma(BA)_s$,

$$fold_s^A(t) = \begin{cases} t^A(id_A) & \text{if } s \in S \setminus BS, \\ g^{-1}(t) & \text{if } s \in BS. \end{cases}$$

Given $(A, g) \in Alg_{\Sigma\downarrow BA}$, the unique $\Sigma\downarrow BA$-homomorphism $unfold^A : A \to \nu\Sigma_{BA}$ is defined as follows: For all $s \in S$ and $a \in A_s$,

$$unfold_s^A(a) = \begin{cases} (t^A(a))_{t \in Obs_{\Sigma,s}} & \text{if } s \in S \setminus BS, \\ g(a) & \text{if } s \in BS. \end{cases}$$

Invariants and induction, congruences and coinduction

Let $\Sigma = (S, F, B\Sigma)$ be a constructor signature and A be a $\Sigma{\downarrow}BA$-algebra.

An S-sorted subset inv of A is a **Σ-invariant** or **Σ-subalgebra of A** if for all $f : w \to s \in F \setminus BF$ and $a \in A_w$,

$$a \in inv \quad \text{implies} \quad f^A(a) \in inv,$$

and for all $s \in BS$, $inv_s = A_s$.

Let $A = \mu\Sigma_{BA}$ and $B \in Alg_{\Sigma\downarrow BA}$. Since A is initial,

(1) A is the only Σ-invariant of A,
(2) $image(fold^B)$ is the least Σ-invariant of B.

By (1), **induction is sound**: Let $R \subseteq A$.

$$A \subseteq R \iff inv \subseteq R \text{ for some } \Sigma\text{-invariant } inv \text{ of } A$$
$$\iff \text{least } \Sigma\text{-invariant of } A \;=\; \cap\,\{inv \mid inv \text{ is a } \Sigma\text{-invariant } inv \text{ of } A\} \subseteq R$$

☞ Since $T_{Reg}(BA) = \mu Reg_{BA}$, induction justifies the inductive definition of a function on regular expressions.

Let $\Sigma = (S, F, B\Sigma)$ be a destructor signature and A be a $\Sigma{\downarrow}BA$-algebra.

An S-sorted binary relation $\sim$ is a **Σ-congruence on** A if for all $f : s \rightarrow w \in F \setminus BF$ and $a, b \in A_s$,

$$a \sim b \quad \text{implies} \quad f^A(a) \sim f^A(b),$$

and for all $s \in BS$, $\sim_s = \Delta_{A,s}$.

Let $A = \nu\Sigma_{BA}$ and $B \in Alg_{\Sigma{\downarrow}BA}$. Since A is final,

(1) Δ_A is the only Σ-congruence on A,
(2) $kernel(unfold^B)$ is the greatest Σ-congruence on B.

By (1), **coinduction is sound**: Let $R \subseteq A \times A$.

$$\begin{aligned} R \subseteq \Delta_A &\iff R \subseteq \sim \text{ for some } \Sigma\text{-congruence} \sim \text{on } A \\ &\iff R \subseteq \text{greatest } \Sigma\text{-congruence on } A \\ &\qquad\qquad = \bigcup\{\sim \mid \sim \text{ is a } \Sigma\text{-congruence} \sim \text{on } A\} \end{aligned}$$

☞ Since $Lang(BA) = \nu Accept_{BA}$, coinduction provides a method for proving that two given languages agree with each other.

Context-free grammars are constructor signatures

A **context-free grammar (CFG)** $G = (S, Z, P, B\Sigma, BG)$ consists of

- a signature $B\Sigma = (BS, BF, B\Sigma')$,
- a $B\Sigma$-Algebra BG,
- a finite set S of **sorts (nonterminals)** including a set BS,
- a set Z of **terminals** that includes the carriers of BG,
- a finite set P of **rules (productions)** of the form $s \to w$ with $s \in S \setminus BS$ and $w \in (S \cup Z \setminus BG)^*$.

The constructor signature

$$\Sigma(G) = (S, F, B\Sigma)$$

with

$$F = \{f_p : s_1 \dots s_n \to s \mid \begin{array}{l} p = (s \to w_0 s_1 w_1 \dots s_n w_n) \in P, \\ w_0, \dots, w_n \in Z^*,\ s_1, \dots, s_n \in S\} \end{array} \}$$

is called the **abstract syntax of** G.

$\Sigma(G)$-terms are called **syntax trees of** G.

$Word(G)$, the word algebra of G

For all $s \in S$,

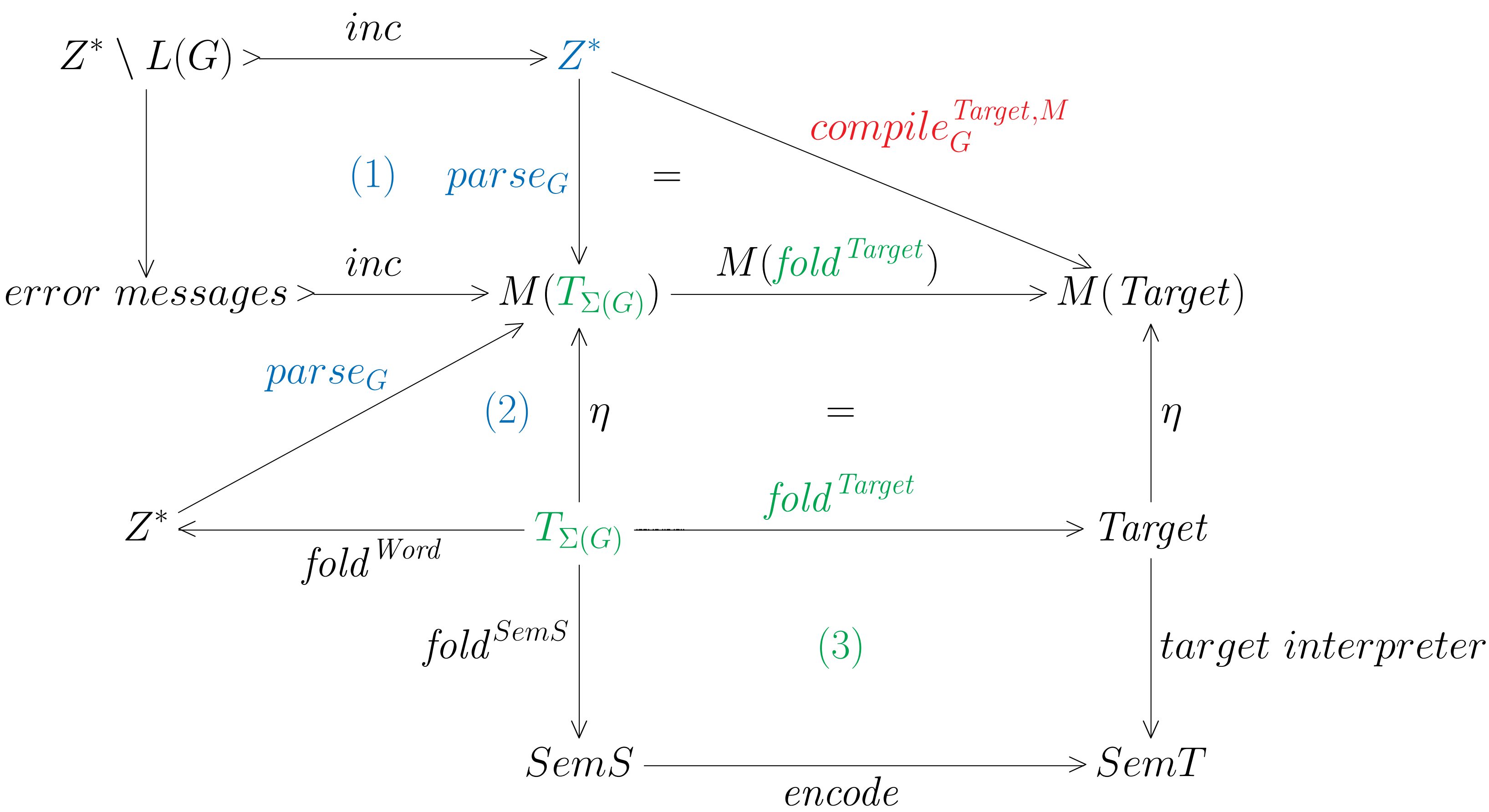
$$Word(G)_s =_{def} \begin{cases} BG_s & \text{if } s \in BS, \\ Z^* & \text{otherwise.} \end{cases}$$

For all $p = (s \to w_0 s_1 w_1 \ldots s_n w_n) \in P$ with $w_0, \ldots, w_n \in Z^*$ and $s_1, \ldots, s_n \in S$,

$$\begin{aligned} f_p^{Word(G)} : Word(G)_{s_1} \times \ldots \times Word(G)_{s_n} &\to Word(G)_s \\ (v_1, \ldots, v_n) &\mapsto w_0 v_1 w_1 \ldots v_n w_n \end{aligned}$$

The **language** $L(G)$ **of** G is the image of $T_{\Sigma(G)}$ under $fold^{Word(G)}$: For all $A \in S \setminus BS$,

$$L(G)_s =_{def} \{fold^{Word(G)}(t) \mid t \in T_{\Sigma(G),s}\}.$$



$compiler_G$ = generic compiler for G

M = monad with unit η M determines error messages and the number of results

$parse_G$ = parser for G (1) = completeness (2) = correctness

$fold^{Target}$ = syntax-tree compiler (3) = correctness

$SemS$ = source language semantics $SemT$ = target language semantics

Derivative parser for regular expressions

☞ **$T_{Reg}(BA)$ is an *Accept*-algebra:** For all $x, y \in Z$ and $R, R' \in T_{Reg,reg}$,

$$\begin{array}{rcl}
\delta^T(\emptyset, x) & = & \emptyset, \\
\delta^T(\epsilon, x) & = & \emptyset, \\
\delta^T(x, y) & = & \mathit{if}\ x = y\ \mathit{then}\ \epsilon\ \mathit{else}\ \emptyset, \\
\delta^T(R|R', x) & = & \delta^T(R, x) \mid \delta^T(R', x), \\
\delta^T(R \cdot R', x) & = & \delta^T(R, x) \cdot R' \mid \mathit{if}\ \mathit{final}^T(R)\ \mathit{then}\ \delta^T(R', x)\ \mathit{else}\ \emptyset, \\
\delta^T(\mathit{star}(R), x) & = & \delta^T(R, x) \cdot \mathit{star}(R),
\end{array}$$

$$\begin{array}{rcl}
\mathit{final}^T(\emptyset) & = & \mathit{False}, \\
\mathit{final}^T(\epsilon) & = & \mathit{True}, \\
\mathit{final}^T(x) & = & \mathit{False}, \\
\mathit{final}^T(R|R') & = & \mathit{final}^T(R) \vee \mathit{final}^T(R'), \\
\mathit{final}^T(R \cdot R') & = & \mathit{final}^T(R) \wedge \mathit{final}^T(R'), \\
\mathit{final}^T(\mathit{star}(R)) & = & \mathit{True}.
\end{array}$$

The derivate parser:

$$T_{Reg,reg} \times Z^* \xrightarrow{(\delta^*)^T} T_{Reg,reg} \xrightarrow{\mathit{final}^T} \mathit{Bool}$$

Moreover, the unique $Reg{\downarrow}BA$-homomorphism

$$fold^{Lang(BA)} : T_{Reg}(BA) \to Lang(BA)$$

is $Accept{\downarrow}BA$-homomorphic.

Hence $fold^{Lang(BA)}$ agrees with the unique $Accept{\downarrow}BA$-homomorphism

$$unfold^{T_{Reg}(BA)} : T_{Reg}(BA) \to Lang(BA).$$

Hence **the derivative parser is correct**, i.e., for all $R \in T_{Reg}(BA)$ and $w \in Z^*$,

$$parse^T(R, w) = True \iff w \in L(R) = fold^{Lang(BA)}(R),$$

and

the greatest $Accept$**-congruence on** $T_{Reg}(BA)$, $kernel(unfold^{T_{Reg}(BA)})$, **agrees with** $kernel(fold^{Lang(BA)})$ **and thus is a** Reg**-congruence**

and

the least Reg**-invariant of** $Lang(BA)$, $image(fold^{Lang(BA)})$, **agrees with** $image(unfold^{T_{Reg}(BA)})$ **and thus is an** $Accept$**-invariant**.

Context-free grammars are systems of *Reg*-equations

Let X be an S-sorted set of variables. An S-sorted function

$$E : X \to T_\Sigma(X)$$

is called a system of **recursive Σ-equations**.

E is **ideal** if for all $x \in X$ $E(x) \notin X$.

Let A be a Σ-algebra. E induces the **step function**

$$\begin{aligned} E_A : A^X &\to A^X \\ f &\mapsto \lambda x.E(x)^A(f) \end{aligned}$$

Fixpoints of E_A coincide with **solutions of E in A**.

Let $G = (S, Z, P, B\Sigma, BG)$ be a context-free grammar. We add the base sorts as *reg*-constants to *Reg*. The *Reg*-algebra *Lang* interprets $s : \epsilon \to reg$ by BG_s.

G can be represented as an ideal system of recursive *Reg*-equations:

$$\begin{aligned} E(G) : S \setminus BS &\to T_{Reg}(S \setminus BS) \\ s &\mapsto \textstyle\sum_{s \to \varphi \in P} \varphi \end{aligned}$$

$$\begin{aligned} \beta : S \setminus BS &\to Lang \\ s &\mapsto L(G)_s \end{aligned}$$

is the least solution of $E(G)$ in $Lang$.

If G is non-left-recursive ($s \not\to_G^+ sw$), then there is exactly one solution of $E(G)$ in $Lang$.

Is used for proving that a given language coincides with $L(G)$.

Extending the derivative parser to parsers for CFGs

Let $R\Sigma$ be the union of *Reg*, *Accept*, the sort *word*, S as additional *reg*-constants and the following function symbols:

$$\begin{aligned} parse &: reg\ word \rightarrow Bool \\ \delta^* &: reg\ word \rightarrow reg \\ [] &: \epsilon \rightarrow word \\ \_ : \_ &: symbol\ word \rightarrow word \\ reduce &: reg \rightarrow reg \\ ite &: Bool\ reg\ reg \rightarrow reg \\ eq, in &: symbol\ symbol \rightarrow symbol \\ \vee, \wedge &: Bool\ Bool \rightarrow Bool \end{aligned}$$

The parser is a set *Red* of rewrite rules between $R\Sigma$-terms over the set $X = \{R, R', w, x\}$ of variables:

$$\begin{aligned} parse(R, w) &\rightarrow final(\delta^*(R, w)) \\ \delta^*(R, x : w) &\rightarrow \delta^*(reduce(\delta(R, x)), w) \\ \delta^*(R, []) &\rightarrow R \\ \delta(\emptyset, x) &\rightarrow \emptyset \\ \delta(\epsilon, x) &\rightarrow \emptyset \end{aligned}$$

$$
\begin{aligned}
\delta(a, x) &\rightarrow ite(eq(a, x), \epsilon, \emptyset) \quad \text{for all } a \in Z \setminus BA \\
\delta(s, x) &\rightarrow ite(x \ in \ A, \epsilon, \emptyset) \quad \text{for all } s \in BS \\
\delta(s, x) &\rightarrow \delta(E(G)(s), x) \quad \text{for all } s \in S \setminus BS \\
\delta(R|R', x) &\rightarrow \delta(R, x) \mid \delta(R', x) \\
\delta(R \cdot R', x) &\rightarrow \delta(R, x) \cdot R' \mid ite(final(R), \delta(R', x), \emptyset) \\
\delta(star(R), x) &\rightarrow \delta(R, x) \cdot star(R) \\
final(\emptyset) &\rightarrow False \\
final(\epsilon) &\rightarrow True \\
final(a) &\rightarrow False \quad \text{for all } a \in BS \cup Z \setminus BA \\
final(s) &\rightarrow final(E(G)(s)) \quad \text{for all } s \in S \setminus BS \\
final(R|R') &\rightarrow final(R) \vee final(R') \\
final(R \cdot R') &\rightarrow final(R) \wedge final(R') \\
final(star(R)) &\rightarrow True \\
ite(True, R, R') &\rightarrow R \\
ite(False, R, R') &\rightarrow R' \\
eq(x, x) &\rightarrow True \\
eq(a, b) &\rightarrow False \quad \text{for all } a, b \in Z \text{ with } a \neq b \\
a \ in \ s &\rightarrow True \quad \text{for all } s \in BS \text{ and } a \in BA_s \\
a \ in \ s &\rightarrow False \quad \text{for all } s \in BS \text{ and } a \in Z \setminus BA_s
\end{aligned}
$$

Let A be an $R\Sigma$-algebra. A rewrite rule $t \to u$ is **correct w.r.t.** A if $t^A = u^A$.

We extend $Lang$ to an $R\Sigma$-algebra by defining for all $s \in S$:

$$s^{Lang} =_{def} \begin{cases} BG_s & \text{if } s \in BS, \\ L(G)_s = fold^{Lang}(E(G)(s)) & \text{otherwise,} \end{cases}$$

and interpreting the above function symbols in the obvious way.

All rewrite rules of Red **are correct w.r.t.** $Lang$**.**

If G **is non-left-recursive, then the parser given by** Red **is correct**, i.e., for all $s \in N \setminus BS$ und $w \in Z^*$,

$$parse(s,w) \xrightarrow{+}_{Red} \begin{cases} True & \text{if } u \in L(G)_s, \\ False & \text{otherwise.} \end{cases}.$$