Towards a Metalanguage for Corecursive Definitions

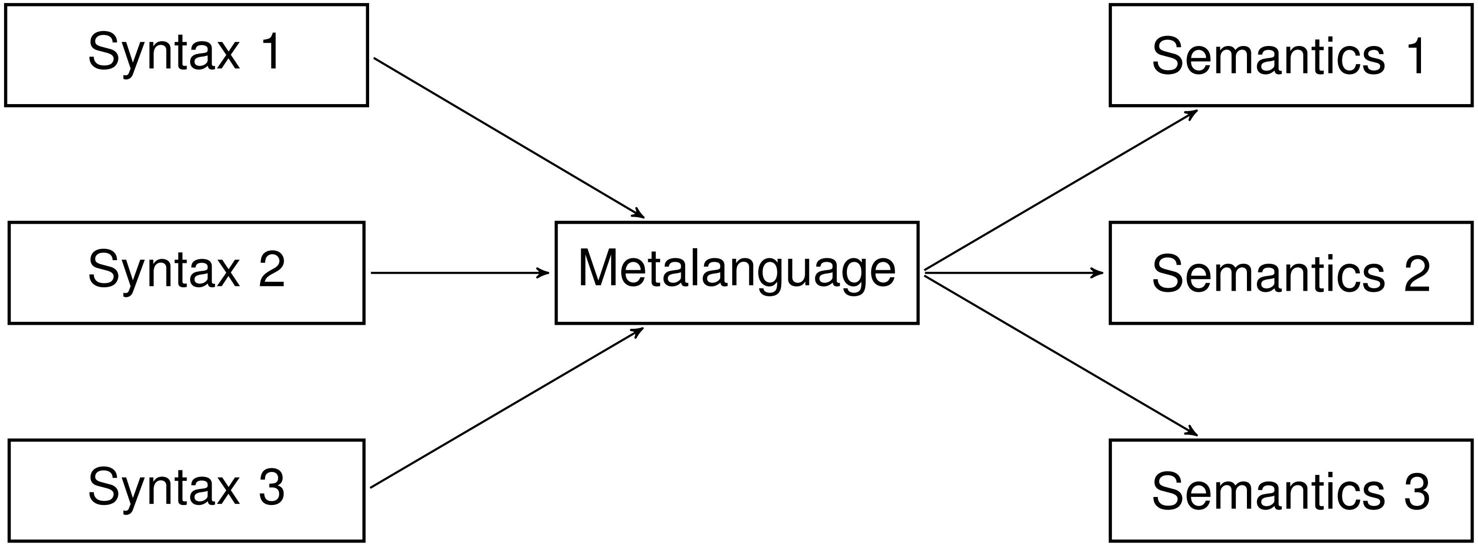
Sergey Goncharov (joint effort with Christoph Rauch & Lutz Schröder)

IFIP WG1.3 Meeting, Nijmegen, 27.06.2015

Chair 8 (Theoretical Computer Science)

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

The Idea of Metalanguage



Our Goal: Metalanguage for Effects + (Co)Recursion

Potential sources:

- Automata theory
- Process algebra
- (Coalgebraic) games
- Functional programming

Our Goal: Metalanguage for Effects + (Co)Recursion

Potential sources:

- Automata theory
- Process algebra
- (Coalgebraic) games
- Functional programming

Potential targets:

Various categories and monads

Effects

Moggi's Computational Metalanguage

- $\mathsf{Type}_W ::= W \mid 1 \mid \mathsf{Type}_W \times \mathsf{Type}_W \mid T(\mathsf{Type}_W)$
- Term construction (Cartesian operators omitted):

$$\frac{x : A \in \Gamma}{\Gamma \vdash x : A} \qquad \frac{\Gamma \vdash t : A}{\Gamma \vdash f(t) : B}\ (f : A \to B \in \Sigma)$$

$$\frac{\Gamma \vdash t : A}{\Gamma \vdash \mathsf{ret}\, t : TA} \qquad \frac{\Gamma \vdash p : TA \qquad \Gamma, x : A \vdash q : TB}{\Gamma \vdash \mathsf{do}\ x \leftarrow p; q : TB}$$

That is interpreted over a strong monad T: Underlying category $\mathcal{C}$, endofunctor $T : \mathcal{C} \to \mathcal{C}$, unit: $\eta : \mathrm{Id} \to T$ and Kleisli star

$$\_^{\star} : \mathrm{hom}(A, TB) \to \mathrm{hom}(TA, TB)$$

plus strength: $\tau_{A,B} : A \times TB \to T(A \times B)$.

Moggi's Metalanguage in Use

- **Syntax/Effectful Operations:** Divergence, Nondeterminism, Exeptions, States, . . .
- **Models/Monads:** Lifting monad (over predomains), powerset monad (over Sets), state monad (over any Cartesian closed category), . . .

Monads for Operations

Alternatively, a monad T is an algebraic theory, that is:

- TX is a set of Σ-terms over variables from X modulo Σ-equations;
- ret x is the variable x seen as a term;
- do $x \leftarrow p; q$ is the substitution $p[x \mapsto q]$.

Thanks to [Plotkin and Power, 2001] we know that algebraic operations $f : n \to 1 \in \Sigma$ are dual to generic effects $1 \to Tn$.

Monads for Operations

Alternatively, a monad T is an algebraic theory, that is:

- TX is a set of Σ-terms over variables from X modulo Σ-equations;
- $\mathsf{ret}\, x$ is the variable x seen as a term;
- $\mathsf{do}\; x \leftarrow p; q$ is the substitution $p[x \mapsto q]$.

Thanks to [Plotkin and Power, 2001] we know that algebraic operations $f : n \to 1 \in \Sigma$ are dual to generic effects $1 \to Tn$.

Example: Finite powerset monad $\mathcal{P}_\omega$ is generated by $\{\emptyset : 0 \to 1, + : 2 \to 1\}$, equivalently by $\{\mathit{abort} : 1 \to \mathcal{P}_\omega 0, \mathit{toss} : 1 \to \mathcal{P}_\omega 2\}$:

$$\emptyset = \mathsf{do}\; \mathit{abort}; \mathsf{ret}\, \star,$$

$$p + q = \mathsf{do}\; x \leftarrow \mathit{toss}; \mathsf{case}\; x\; \mathsf{of}\; \mathsf{inl}\, \star \mapsto p; \mathsf{inr}\, \star \mapsto q.$$

Our Agenda: Effects + Recursion

What are the general settings allowing for solving systems of equations

$$f_i = t_i(f_1, \dots, f_n)$$

where f is a function and t_i is a term constructed from interpreted and uninterpreted functions (including the f_i)?

- Interpreted means: satisfies an equational axiomatization, e.g.

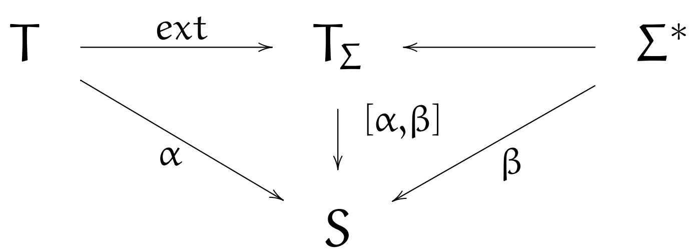
$$\emptyset + p = p + \emptyset = p, \;\; p + q = q + p, \;\; (p + q) + r = p + (q + r).$$

  This induces a monad: TX are terms over X modulo provable equivalence; $(f : X \to TY)^\star : TX \to TY$ is the substitution operation.

- Uninterpreted means: satisfies no equations.

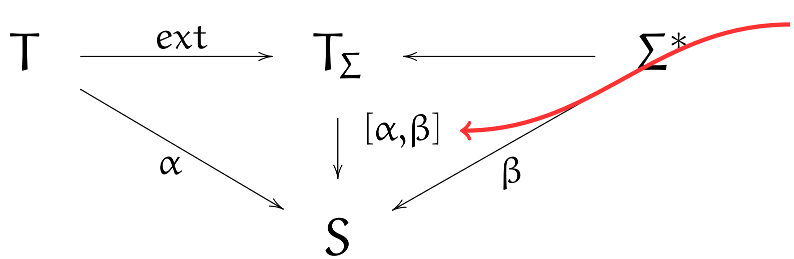
Free Completion: Finite Case

- Recall that TX is the object of terms over a signature of operations modulo equations.
- Given a signature Σ, $\Sigma^* X = \mu\gamma.\,(X + \Sigma\gamma)$ is the free monad over Σ.
- By [Hyland, Levy, Plotkin, and Power, 2007], $T_\Sigma X = \mu\gamma.\,T(X + \Sigma\gamma)$ is the coproduct in the category of monads:



Free Completion: Finite Case

- Recall that $\mathsf{T}X$ is the object of terms over a signature of operations modulo equations.
- Given a signature Σ, $\Sigma^*X = \mu\gamma.\,(X + \Sigma\gamma)$ is the free monad over Σ.
- By [Hyland, Levy, Plotkin, and Power, 2007], $\mathsf{T}_\Sigma X = \mu\gamma.\,\mathsf{T}(X + \Sigma\gamma)$ is the coproduct in the category of monads:

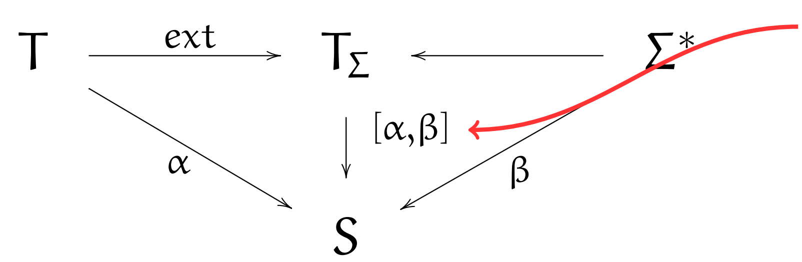


Equivalently a pair (α, g):

- $\alpha : \mathsf{T} \to S$ is a monad morphism;
- $g : \Sigma \to S$ is a nat. transformation.

Free Completion: Finite Case

- Recall that TX is the object of terms over a signature of operations modulo equations.
- Given a signature Σ, $\Sigma^* X = \mu\gamma.\,(X + \Sigma\gamma)$ is the free monad over Σ.
- By [Hyland, Levy, Plotkin, and Power, 2007], $T_\Sigma X = \mu\gamma.\, T(X + \Sigma\gamma)$ is the coproduct in the category of monads:



If $\Sigma = a \times -^b$ then

- $\alpha : T \to S$ is a monad morphism;
- $g : a \to Sb$ (Yoneda).

Recursion

Free Completion: Infinite Case

Complete Elgot monad is a monad, equipped with an iteration operator:

$$-^{\dagger} : \mathrm{hom}(A, T(B + A)) \to \mathrm{hom}(A, TB)$$

satisfying some axioms [Bloom and Ésik, 1993] .

Free Completion: Infinite Case

(Complete) Elgot monad is a monad, equipped with an iteration operator:

$$-^{\dagger} : \mathrm{hom}(A, T(B + A)) \to \mathrm{hom}(A, TB)$$

satisfying some axioms [Bloom and Ésik, 1993] .

Free Completion: Infinite Case

(Complete) Elgot monad is a monad, equipped with an iteration operator:

$$-^{\dagger} : \mathsf{hom}(A, T(B + A)) \to \mathsf{hom}(A, TB)$$

satisfying some axioms [Bloom and Ésik, 1993] such as dinaturality:

$$([\eta \circ \mathsf{inl}, h]^{\star} \circ g)^{\dagger} = [\eta, ([\eta \circ \mathsf{inl}, g]^{\star} \circ h)^{\dagger}]^{\star} \circ g.$$

Free Completion: Infinite Case

(Complete) Elgot monad is a monad, equipped with an iteration operator:

$$-^{\dagger} : \mathsf{hom}(A, T(B + A)) \to \mathsf{hom}(A, TB)$$

satisfying some axioms [Bloom and Ésik, 1993] .

Free Completion: Infinite Case

(Complete) Elgot monad is a monad, equipped with an iteration operator:

$$\_^{\dagger} : \mathrm{hom}(A, T(B + A)) \to \mathrm{hom}(A, TB)$$

satisfying some axioms [Bloom and Ésik, 1993] .

Examples include pointed monads over order-enriched categories, powerset, nontermination, their combinations with other effects.

Since $\bot = (\eta\,\mathsf{inr} : X \to T(\emptyset + X))^{\dagger}$, any Elgot monad is pointed, e.g. $\mathrm{I}X = X + 1$ is the initial Elgot monad on Sets.

Free Completion: Infinite Case

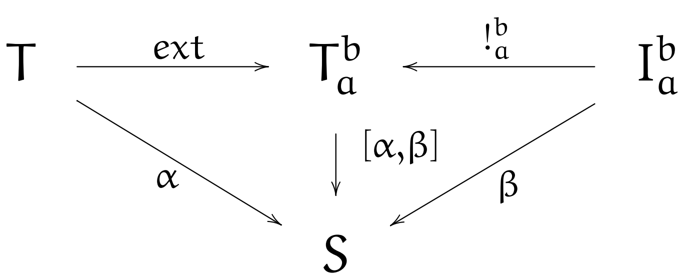
(Complete) Elgot monad is a monad, equipped with an iteration operator:

$$-^{\dagger} : \mathrm{hom}(A, T(B + A)) \to \mathrm{hom}(A, TB)$$

satisfying some axioms [Bloom and Ésik, 1993] .

Examples include pointed monads over order-enriched categories, powerset, nontermination, their combinations with other effects.

By [Goncharov, Rauch, and Schröder, 2015], $T^b_a \cong T + I^b_a$
where $T^b_a X = \nu\gamma.\, T(X + a \times \gamma^b)$:



Free Completion: Infinite Case

Verified in Coq (~5000 lines)

(Complete) Elgot monad is a monad, equipped with an iteration operator:

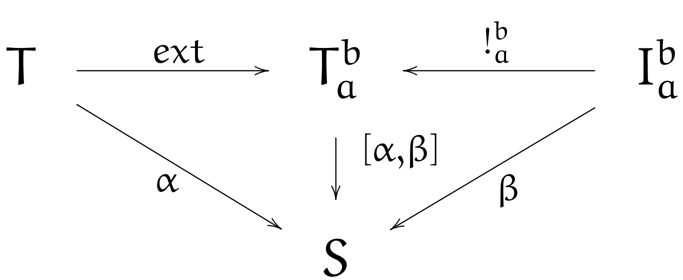
$$-^{\dagger} : \mathrm{hom}(A, T(B + A)) \to \mathrm{hom}(A, TB)$$

satisfying some axioms [Bloom and Ésik, 1993] .

Examples include pointed monads over order-enriched categories, powerset, nontermination, their combinations with other effects.

By [Goncharov, Rauch, and Schröder, 2015], $T^b_a \cong T + I^b_a$

where $T^b_a X = \nu\gamma.\, T(X + a \times \gamma^b)$:



Free Completion: Infinite Case

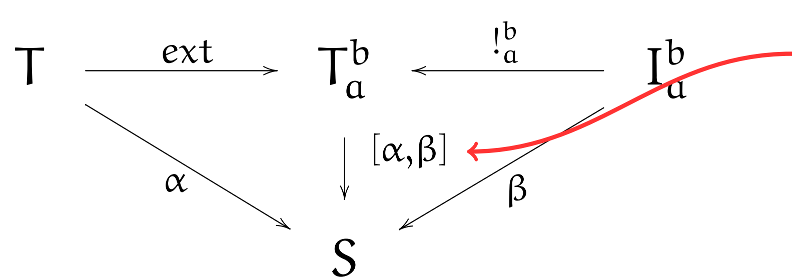
(Complete) Elgot monad is a monad, equipped with an iteration operator:

$$-^{\dagger} : \mathrm{hom}(A, T(B + A)) \to \mathrm{hom}(A, TB)$$

satisfying some axioms [Bloom and Ésik, 1993] .

Examples include pointed monads over order-enriched categories, powerset, nontermination, their combinations with other effects.

By [Goncharov, Rauch, and Schröder, 2015], $T^b_a \cong T + I^b_a$ where $T^b_a X = \nu\gamma.\, T(X + a \times \gamma^b)$:



Equivalently a pair (α, g):

- $\alpha : T \to S$ is a monad morphism;
- $g : a \to Sb$.

Free Elgot monad over Σ

Why $\Sigma X = a \times X^b$?

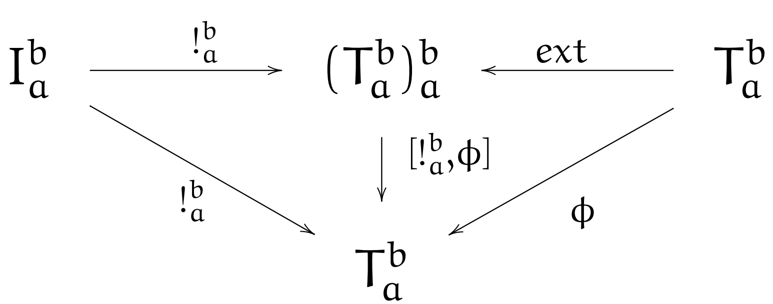
Althought we believe that $\nu\gamma.\, T(\text{--} + \Sigma\gamma) \cong T + \Sigma^\infty$ for any Σ,

still $\Sigma = a \times \text{--}^b$ is versatile, for

- we can iterate the coproduct construction to obtain

$$T + I_{a_1}^{b_1} + \cdots + I_{a_n}^{b_n} \cong \nu\gamma.\, T(\text{--} + a_1 \times \gamma^{b_1} + \cdots + a_n \times \gamma^{b_n})$$

- we can model recursion:



Free Elgot monad over Σ

Instead of $(a \to Tb) \to (a \to Tb)$

Why $\Sigma X = a \times X^b$?

Althought we believe that $\nu\gamma.\,T(-+\Sigma\gamma) \cong T + \Sigma^\infty$ for any Σ,

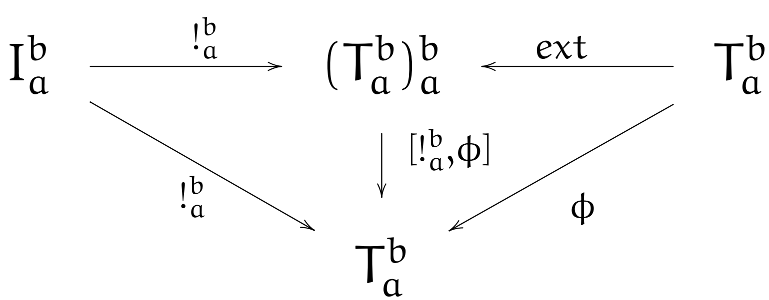
still $\Sigma = a \times -^b$ is versatile, for

- we can iterate the coproduct construction to obtain

$$T + I_{a_1}^{b_1} + \cdots + I_{a_n}^{b_n} \cong \nu\gamma.\,T(-+a_1 \times \gamma^{b_1} + \cdots + a_n \times \gamma^{b_n})$$

- we can model recursion:

1. Start with $f : a \to T_a^b b$.



Free Elgot monad over Σ

Instead of $(a \to Tb) \to (a \to Tb)$

Why $\Sigma X = a \times X^b$?

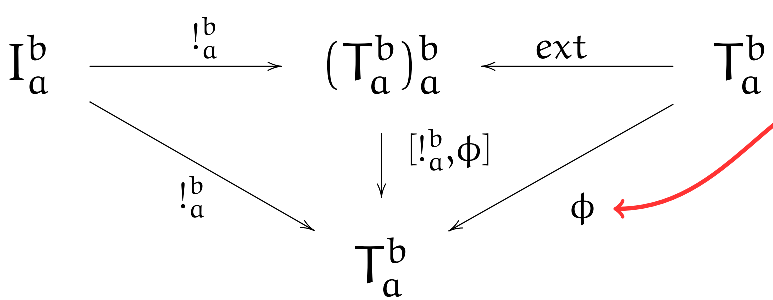
Althought we believe that $\nu\gamma.\, T(- + \Sigma\gamma) \cong T + \Sigma^\infty$ for any Σ,

still $\Sigma = a \times -^b$ is versatile, for

- we can iterate the coproduct construction to obtain

$$T + I^{b_1}_{a_1} + \cdots + I^{b_n}_{a_n} \cong \nu\gamma.\, T(- + a_1 \times \gamma^{b_1} + \cdots + a_n \times \gamma^{b_n})$$

- we can model recursion:



1. Start with $f : a \to T^b_a b$.
2. Construct $\phi : T^b_a \to T^b_a$.

Free Elgot monad over Σ

Instead of $(a \to Tb) \to (a \to Tb)$

Why $\Sigma X = a \times X^b$?

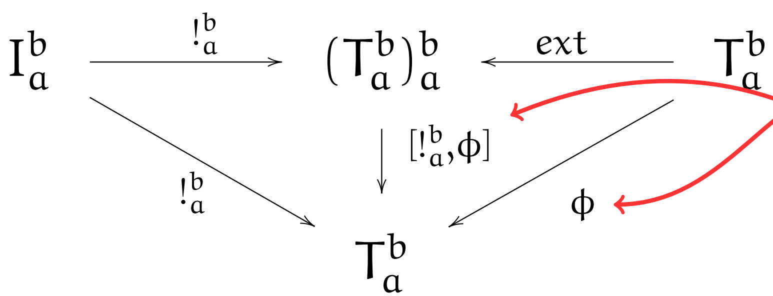
Althought we believe that $\nu\gamma.\, T(-+\Sigma\gamma) \cong T + \Sigma^\infty$ for any Σ,

still $\Sigma = a \times -^b$ is versatile, for

- we can iterate the coproduct construction to obtain

$$T + I^{b_1}_{a_1} + \cdots + I^{b_n}_{a_n} \cong \nu\gamma.\, T(-+a_1 \times \gamma^{b_1} + \cdots + a_n \times \gamma^{b_n})$$

- we can model recursion:



1. Start with $f : a \to T^b_a b$.
2. Construct $\phi : T^b_a \to T^b_a$.
3. Obtain $(T^b_a)^b_a \to T^b_a$.

Free Elgot monad over Σ

Instead of $(a \to Tb) \to (a \to Tb)$

Why $\Sigma X = a \times X^b$?

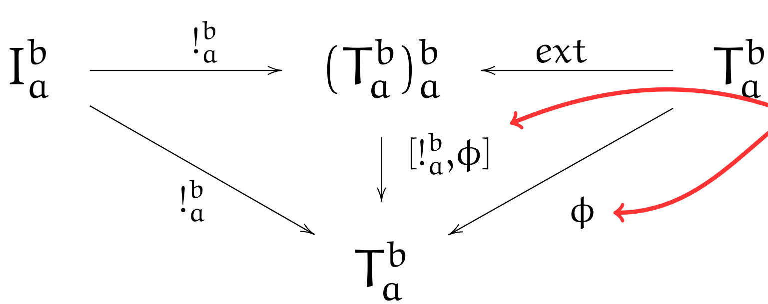
Althought we believe that $\nu\gamma.\, T(- + \Sigma\gamma) \cong T + \Sigma^\infty$ for any Σ,

still $\Sigma = a \times -^b$ is versatile, for

- we can iterate the coproduct construction to obtain

$$T + I_{a_1}^{b_1} + \cdots + I_{a_n}^{b_n} \cong \nu\gamma.\, T(- + a_1 \times \gamma^{b_1} + \cdots + a_n \times \gamma^{b_n})$$

- we can model recursion:



1. Start with $f : a \to T_a^b b$.
2. Construct $\phi : T_a^b \to T_a^b$.
3. Obtain $(T_a^b)_a^b \to T_a^b$.
4. Extract $\mathtt{fix}(f) : a \to Tb$.

Construction of Solutions

- $f : X \to T_a^b(Y + X)$ is guarded iff there exists a u such that

$$\begin{array}{ccccc} X & \xrightarrow{f} & T_a^b(Y+X) & \xrightarrow{\mathrm{out}} & T((Y+X) + a \times T_a^b(Y+X)^b) \\ & \searrow^{u} & & & \uparrow{\scriptstyle T(\mathsf{inl}+\mathrm{id})} \\ & & & & T(Y + a \times T_a^b(Y+X)^b) \end{array}$$

Construction of Solutions

- $f : X \to T_a^b(Y + X)$ is guarded iff there exists a u such that

$$\begin{array}{ccccc} X & \xrightarrow{f} & T_a^b(Y+X) & \xrightarrow{\mathsf{out}} & T((Y+X) + a \times T_a^b(Y+X)^b) \\ & & & \searrow^{u} & \uparrow{\scriptstyle T(\mathsf{inl}+\mathrm{id})} \\ & & & & T(Y + a \times T_a^b(Y+X)^b) \end{array}$$

- By [Uustalu, 2003], guarded morphisms admitt unique solutions.

Construction of Solutions

- $f : X \to T_a^b(Y + X)$ is guarded iff there exists a u such that

$$\begin{array}{ccccc} X & \xrightarrow{f} & T_a^b(Y+X) & \xrightarrow{\mathrm{out}} & T((Y+X) + a \times T_a^b(Y+X)^b) \\ & \searrow^{u} & & & \uparrow{\scriptstyle T(\mathsf{inl} + \mathrm{id})} \\ & & & & T(Y + a \times T_a^b(Y+X)^b) \end{array}$$

- By [Uustalu, 2003], guarded morphisms admitt unique solutions.
- Any $f : X \to T_a^b(Y + X)$ gives rise to

$$X \xrightarrow{f} T_a^b(Y+X) \xrightarrow{\mathrm{out}} T((Y+X) + a \times T_a^b(Y+X)^b)$$

Construction of Solutions

- $f : X \to T_a^b(Y + X)$ is guarded iff there exists a u such that

$$\begin{array}{ccc} X \xrightarrow{f} T_a^b(Y+X) & \xrightarrow{\mathsf{out}} & T((Y+X) + a \times T_a^b(Y+X)^b) \\ & \searrow^{u} & \uparrow{\scriptstyle T(\mathsf{inl}+\mathrm{id})} \\ & & T(Y + a \times T_a^b(Y+X)^b) \end{array}$$

- By [Uustalu, 2003], guarded morphisms admitt unique solutions.
- Any $f : X \to T_a^b(Y + X)$ gives rise to

$$X \xrightarrow{f} T_a^b(Y+X) \xrightarrow{\mathsf{out}} T((Y+X) + a \times T_a^b(Y+X)^b)$$
$$\xrightarrow{T\pi} T((Y + a \times T_a^b(Y+X)^b) + X)$$

Construction of Solutions

- $f : X \to T_a^b(Y + X)$ is guarded iff there exists a u such that

$$
\begin{array}{ccc}
X \xrightarrow{f} T_a^b(Y+X) & \xrightarrow{\mathrm{out}} & T((Y+X) + a \times T_a^b(Y+X)^b) \\
 & \searrow^{u} & \uparrow T(\mathsf{inl} + \mathrm{id}) \\
 & & T(Y + a \times T_a^b(Y+X)^b)
\end{array}
$$

- By [Uustalu, 2003], guarded morphisms admitt unique solutions.
- Any $f : X \to T_a^b(Y + X)$ gives rise to

$$X \xrightarrow{T\pi \circ \mathrm{out} \circ f} T((Y + a \times T_a^b(Y+X)^b) + X)$$

Construction of Solutions

- $f : X \to T_a^b(Y + X)$ is guarded iff there exists a u such that

$$X \xrightarrow{f} T_a^b(Y + X) \xrightarrow{out} T((Y + X) + a \times T_a^b(Y + X)^b)$$

$$X \xrightarrow{u} T(Y + a \times T_a^b(Y + X)^b) \xrightarrow{T(\mathsf{inl} + id)} T((Y + X) + a \times T_a^b(Y + X)^b)$$

- By [Uustalu, 2003], guarded morphisms admitt unique solutions.
- Any $f : X \to T_a^b(Y + X)$ gives rise to

$$\big(X \xrightarrow{T\pi \circ out \circ f} T((Y + a \times T_a^b(Y + X)^b) + X)\big)^\dagger$$

Construction of Solutions

- $f : X \to T_a^b(Y + X)$ is guarded iff there exists a u such that

$$X \xrightarrow{f} T_a^b(Y+X) \xrightarrow{\mathsf{out}} T((Y+X) + a \times T_a^b(Y+X)^b)$$

$$X \xrightarrow{u} T(Y + a \times T_a^b(Y+X)^b) \xrightarrow{T(\mathsf{inl} + \mathrm{id})} T((Y+X) + a \times T_a^b(Y+X)^b)$$

- By [Uustalu, 2003], guarded morphisms admitt unique solutions.
- Any $f : X \to T_a^b(Y + X)$ gives rise to

$$X \xrightarrow{(T\pi \circ \mathsf{out} \circ f)^\dagger} T(Y + a \times T_a^b(Y+X)^b)$$

Construction of Solutions

- $f : X \to T_a^b(Y + X)$ is guarded iff there exists a u such that

$$\begin{array}{ccccc} X & \xrightarrow{f} & T_a^b(Y+X) & \xrightarrow{\mathsf{out}} & T((Y+X) + a \times T_a^b(Y+X)^b) \\ & \searrow^{u} & & & \uparrow {\scriptstyle T(\mathsf{inl}+\mathrm{id})} \\ & & & & T(Y + a \times T_a^b(Y+X)^b) \end{array}$$

- By [Uustalu, 2003], guarded morphisms admit unique solutions.
- Any $f : X \to T_a^b(Y + X)$ gives rise to

$$X \xrightarrow{(T\pi \circ \mathsf{out} \circ f)^{\dagger}} T(Y + a \times T_a^b(Y+X)^b)$$

Construction of Solutions

- $f : X \to T_a^b(Y+X)$ is guarded iff there exists a u such that

$$\begin{array}{ccccc} X & \xrightarrow{f} & T_a^b(Y+X) & \xrightarrow{out} & T((Y+X)+a\times T_a^b(Y+X)^b) \\ & \searrow^{u} & & & \uparrow{\scriptstyle T(\mathsf{inl}+\mathrm{id})} \\ & & & & T(Y+a\times T_a^b(Y+X)^b) \end{array}$$

- By [Uustalu, 2003], guarded morphisms admit unique solutions.
- Any $f : X \to T_a^b(Y+X)$ gives rise to

$$X \xrightarrow{(T\pi\circ \mathrm{out}\circ f)^\dagger} T(Y+a\times T_a^b(Y+X)^b) \xrightarrow{\mathsf{tuo}\circ T(\mathsf{inl}+\mathrm{id})} T_a^b(Y+X)$$

Construction of Solutions

- $f : X \to T_a^b(Y + X)$ is guarded iff there exists a u such that

$$X \xrightarrow{f} T_a^b(Y+X) \xrightarrow{out} T((Y+X) + a \times T_a^b(Y+X)^b)$$

$$X \xrightarrow{u} T(Y + a \times T_a^b(Y+X)^b) \xrightarrow{T(inl + id)} T((Y+X) + a \times T_a^b(Y+X)^b)$$

- By [Uustalu, 2003], guarded morphisms admitt unique solutions.
- Any $f : X \to T_a^b(Y + X)$ gives rise to

$$X \xrightarrow{(T\pi \circ out \circ f)^\dagger} T(Y + a \times T_a^b(Y+X)^b) \xrightarrow{tuo \circ T(inl + id)} T_a^b(Y+X)$$

which we denote $\rhd f : X \to T_a^b(Y+X)$ and put $f^\dagger := (\rhd f)^\dagger$.

Guardedness

Guardedness: Example from Process Algebra

Basic Process Algebra (BPA) terms are given by the grammar

$$P, Q ::= X \in \mathit{Vars} \mid a \in \mathit{Act} \mid \emptyset \mid P + Q \mid P \cdot Q$$

Guardedness: Example from Process Algebra

Sequential composition

Basic Process Algebra (BPA) terms are given by the grammar

$$P, Q ::= X \in \mathit{Vars} \mid a \in \mathit{Act} \mid \emptyset \mid P + Q \mid P \cdot Q$$

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Guardedness: Example from Process Algebra

Sequential composition

Basic Process Algebra (BPA) terms are given by the grammar

$$P, Q ::= X \in \mathit{Vars} \mid a \in \mathit{Act} \mid \emptyset \mid P + Q \mid P \cdot Q$$

A term is guarded if it is either an action, or $\emptyset$ or a sum of guarded terms or a composition $P \cdot Q$ with guarded P.

Guardedness: Example from Process Algebra

Basic Process Algebra (BPA) terms are given by the grammar

$$P, Q ::= X \in \mathit{Vars} \mid a \in \mathit{Act} \mid \emptyset \mid P + Q \mid P \cdot Q$$

A term is guarded if it is either an action, or $\emptyset$ or a sum of guarded terms or a composition $P \cdot Q$ with guarded P.

Theorem [Bergstra and Klop, 1984]. A system of equations $X_i \equiv P_i$ with $\{X_i\}_i = \bigcup_i \mathit{Vars}(P_i)$ and guarded P_i uniquelly determines a solution $(S_i)_i$ w.r.t. the semantics of proceses as finitely-branching trees with edges labelled in Act.

Example from Process Algebra (Continued)

Example: $\{X \equiv a \cdot X + b \cdot Y, Y \equiv (a + b \cdot Y) \cdot X\}$.

Example from Process Algebra (Continued)

Example: $\{X \equiv a \cdot X + b \cdot Y, Y \equiv (a + b \cdot Y) \cdot X\}$.

(Non-Genuine) Non-Example: $\{X \equiv a \cdot X + \ Y\ , Y \equiv (a + b \cdot Y) \cdot X\}$
(Solution still exists and unique).

(Genuine) Non-Example: $\{X \equiv a \cdot X + b \cdot Y, Y \equiv (a + \ Y\ ) \cdot X\}$
(If (P, Q) is a solution then any $(P, Q + R)$ with $R = c \cdot R$ is a solution).

Example from Process Algebra (Continued)

Unguarded call

Example: $\{X \equiv a \cdot X + b \cdot Y, Y \equiv (a + b \cdot Y) \cdot X\}$.

(Non-Genuine) Non-Example: $\{X \equiv a \cdot X + Y, Y \equiv (a + b \cdot Y) \cdot X\}$ (Solution still exists and unique).

(Genuine) Non-Example: $\{X \equiv a \cdot X + b \cdot Y, Y \equiv (a + Y) \cdot X\}$ (If (P, Q) is a solution then any $(P, Q + R)$ with $R = c \cdot R$ is a solution).

This is a perfect example for T_a^b: T — finite powerset monad; a — actions; $b = 1$; composition is Kleisli composition, etc. Guardedness is guardedness.

Example from Process Algebra (Continued)

Example: $\{X \equiv a \cdot X + b \cdot Y, Y \equiv (a + b \cdot Y) \cdot X\}$.

(Non-Genuine) Non-Example: $\{X \equiv a \cdot X + \ Y\ , Y \equiv (a + b \cdot Y) \cdot X\}$ (Solution still exists and unique).

(Genuine) Non-Example: $\{X \equiv a \cdot X + b \cdot Y, Y \equiv (a + \ Y\ ) \cdot X\}$ (If (P, Q) is a solution then any $(P, Q + R)$ with $R = c \cdot R$ is a solution).

This is a perfect example for T^b_a: T — finite powerset monad; a — actions; $b = 1$; composition is Kleisli composition, etc. Guardedness is guardedness.

Example from Process Algebra (Continued)

Example: $\{X \equiv a \cdot X + b \cdot Y, Y \equiv (a + b \cdot Y) \cdot X\}$.

(Non-Genuine) Non-Example: $\{X \equiv a \cdot X + \ \ Y\ , Y \equiv (a + b \cdot Y) \cdot X\}$
(Solution still exists and unique).

(Genuine) Non-Example: $\{X \equiv a \cdot X + b \cdot Y, Y \equiv (a + \ \ Y\ ) \cdot X\}$
(If (P, Q) is a solution then any $(P, Q + R)$ with $R = c \cdot R$ is a solution).

This is a perfect example for T_a^b: T — finite powerset monad; a — actions; $b = 1$; composition is Kleisli composition, etc. Guardedness is guardedness.

Question: can we define parallel composition this way?

Example from Process Algebra (Continued)

Example: $\{X \equiv a \cdot X + b \cdot Y, Y \equiv (a + b \cdot Y) \cdot X\}$.

(Non-Genuine) Non-Example: $\{X \equiv a \cdot X + Y, Y \equiv (a + b \cdot Y) \cdot X\}$ (Solution still exists and unique).

(Genuine) Non-Example: $\{X \equiv a \cdot X + b \cdot Y, Y \equiv (a + Y) \cdot X\}$ (If (P, Q) is a solution then any $(P, Q + R)$ with $R = c \cdot R$ is a solution).

This is a perfect example for T^b_a: T — finite powerset monad; a — actions; $b = 1$; composition is Kleisli composition, etc. Guardedness is guardedness.

Question: can we define parallel composition this way?

Answer: Well..

Coalgebra Way

An F-coalgebra is a map $\xi : X \to FX$. E.g. the semantic domain for BPA-processes form a $\mathcal{P}_\omega(A \times -)$-coalgebra. In fact, the final one:

$$\begin{array}{ccc} X & \overset{\widehat{\xi}}{\dashrightarrow} & P_{Act} \\ {\scriptstyle \xi}\downarrow & & \downarrow{\scriptstyle \iota} \\ \mathcal{P}_\omega(A \times X) & \underset{\mathcal{P}_\omega(A \times \widehat{\xi})}{\dashrightarrow} & \mathcal{P}_\omega(A \times P_{Act}) \end{array}$$

Since $\mathcal{P}_\omega(A \times X) \subseteq \mathcal{P}_\omega(X)^A$, we have $\mathcal{P}_\omega(A \times -)$-coalgebra on X whenever we we know all derivatives $\partial_a : X \to \mathcal{P}_\omega(X)$.

Coalgebra Way

An F-coalgebra is a map $\xi : X \to FX$. E.g. the semantic domain for BPA-processes form a $\mathcal{P}_\omega(A \times -)$-coalgebra. In fact, the final one:

$$\begin{array}{ccc} X & \overset{\widehat{\xi}}{\dashrightarrow} & P_{Act} \\ \downarrow{\scriptstyle \xi} & & \downarrow{\scriptstyle \iota} \\ \mathcal{P}_\omega(A \times X) & \underset{\mathcal{P}_\omega(A \times \widehat{\xi})}{\dashrightarrow} & \mathcal{P}_\omega(A \times P_{Act}) \end{array}$$

Since $\mathcal{P}_\omega(A \times X) \subseteq \mathcal{P}_\omega(X)^A$, we have $\mathcal{P}_\omega(A \times -)$-coalgebra on X whenever we we know all derivatives $\partial_a : X \to \mathcal{P}_\omega(X)$.

E.g. for BPA-terms:

$$\partial_a(a) = \{\emptyset\}, \quad \partial_a(P + Q) = \partial_a(P) \cup \partial_a(Q), \quad \partial_a(P \cdot Q) = \partial_a(P) \cdot Q$$

where $\emptyset \cdot Q = \emptyset$, $(s \cup t) \cdot Q = s \cdot Q \cup t \cdot Q$, $\{\emptyset\} \cdot Q = \{Q\}$, $\{P\} \cdot Q = \{P \cdot Q\}$.

Coalgebra Way: Parallel Composition

We can extend our grammar by adding the parallel composition:

$$P, Q ::= .. \mid (P \mid Q),$$

for we can define

$$\partial_a(P \mid Q) = \{(S \mid Q) \mid S \in \partial_a(P)\} \cup \{(P \mid S) \mid S \in \partial_a(Q)\}$$

and the like.

The general pattern here is: The derivative of a function is expressed via a function of derivatives.

Bialgebraic Way, aka. abstract GSOS-Semantics

Instead of recursive equations we write operational semantic rules, e.g.

$$\frac{P \xrightarrow{a} P'}{P \mid Q \xrightarrow{a} P' \mid Q} \qquad \frac{Q \xrightarrow{a} Q'}{P \mid Q \xrightarrow{a} P \mid Q'}$$

with the same meaning: the behaviour of a function is expressed via a function of behaviours.

Bialgebraic Way, aka. abstract GSOS-Semantics

Instead of recursive equations we write operational semantic rules, e.g.

$$\frac{P \xrightarrow{a} P'}{P \mid Q \xrightarrow{a} P' \mid Q} \qquad \frac{Q \xrightarrow{a} Q'}{P \mid Q \xrightarrow{a} P \mid Q'}$$

with the same meaning: the behaviour of a function is expressed via a function of behaviours.

Theorem [Turi and Plotkin, 1997]: Given a signature of operations Σ (such as $+, \cdot$, etc), a behaviour functor B (such as $\mathcal{P}_\omega(A \times -)$) and a natural transformation $\Sigma(B \times \mathrm{Id}) \to B\Sigma^*$ there is a canonical Σ-algebra structure on the B-coalgebra.

How Can We Cope with Guarded Corecursion?

We can extend the BPA-grammar yet further:

$$P, Q ::= .. \mid \mathsf{rec}\ X.\ P$$

where P is a guarded (!) term. Then we can put

$$\partial_a(\mathsf{rec}\ X.\ P) = \partial_a P[(\mathsf{rec}\ X.\ P)/X].$$

The argument that this is well-defined critically depends on P being guarded.

It does not seem possible to convert this kind of arguments into GSOS-rules in a natural and general way.

Higher-Order Behavioral Differential Equations

Definitions in terms of derivatives are also called behavioral differential equations [Rutten, 2003].

Example (Zipping Infinite Lists): For infinite lists $\nu\gamma.\,(A \times \gamma) = A^\omega$,

$$o(zip(p, q)) = o(p) \qquad\qquad (zip(p, q))' = zip(q, p')$$

This corresponds to a GSOS-rule $\Sigma(B \times \mathrm{Id}) \to BT$ with $B = (A \times \text{-})$, $\Sigma X = X^2$

Example (Dropping Even Elements):

$$o(drop2(p)) = o(p) \qquad\qquad (drop2(p))' = drop2(p'')$$

Here we would need a “GSOS-rule” $\Sigma(B^2 \times \mathrm{Id}) \to BT$.

Example (Tail Function): $o(tail(p)) = o(p')$.

Related Work

In [Milius, Moss, and Schwencke, 2013] the authors faced a similar kind of challenge.

The proposed solution is to partition the set of definable operations into those defined via (abstract) GSOS and those defined via guarded corecursion and iterate this process.

A Syntax for Free Operations

We postulate a signature Ξ for free operations $B \to A$.

$$\frac{\Gamma \vdash p : [C]_f \qquad f : B \to A \in \Xi}{\Gamma \vdash \mathsf{pr}_1\, p : A}$$

$$\frac{\Gamma \vdash p : [C]_f \qquad \Gamma \vdash q : B \qquad f : B \to A \in \Xi}{\Gamma \vdash p \mathbin{\$} q : C}$$

$$\frac{\Gamma \vdash p : A \qquad \Gamma, x : B \vdash q : C \qquad f : B \to A \in \Xi}{\Gamma \vdash \langle p, x.\, q \rangle_f : [C]_f}$$

The type $[C]_f$ with $f : B \to A$ models the object $A \times C^B$.

A Syntax for Coalgebras

The final coalgebra structure $\iota : T_f C \to T(C + [T_f C]_f)$ and the final coalgebra morphism are mimicked as follows:

$$\frac{\Gamma \vdash p : T_f C}{\Gamma \vdash \mathsf{out}\, p : T(C + [T_f C]_f)}$$

$$\frac{\Gamma \vdash p : D \qquad \Gamma, x : D \vdash q : T(C + [D]_f)}{\Gamma \vdash \mathsf{init}\, x \Leftarrow p\ \mathsf{coit}\ q : T_f C}$$

The corresponding complete quasi-equational axiomatization is easy to obtain.

Syntactic Notion of Guardedness

Summarized, our type system is as follows:

$$A, B \ldots ::= V \mid 0 \mid 1 \mid A \times B \mid A + B \mid [A]_f \mid TA \qquad (V \in \mathcal{V})$$
$$T, S \ldots ::= U \mid T_f \qquad (U \in \mathcal{U}, f \in \Xi)$$

Definiton. Let s be a nonemty string from $\{1, 2\}^*$. A term $\Gamma \vdash p : TC$ is s-guarded if one of the following recursive clauses apply:

- $s = 1s'$ and $p = \text{do } z \leftarrow p'; \text{ret inr } z$ with some s' and p';
- $s = 1s'$ and $p = \text{do } z \leftarrow p'; \text{ret inl } z$ with some s' and s'-guarded p';
- symmetrically for $s = 2s'$;
- $p = \text{match}[x, y] \leftarrow q; x \mapsto p_1; y \mapsto p_2$ with some s-guarded p_1 and p_2;
- T is of the form S_f and $\Gamma \vdash \text{out } p : S(C + [TC]_f)$ is $1s$-guarded.

Solution Theorem

Let us write match $[x, y] \leftarrow p; x \mapsto q; y \mapsto r$ for

$$\text{do } z \leftarrow p; \text{case } z \text{ of inl } x \mapsto q; \text{inr } y \mapsto r.$$

Theorem. For any 2-guarded term $\Gamma, x : D \vdash p : T(C + D)$, there exists, up to semantic equality, a unique term $\Gamma, x : D \vdash p^{\dagger} : TC$ satisfying the equation

$$p^{\dagger} = \text{match } [y, x] \leftarrow p; y \mapsto \text{ret } y; x \mapsto p^{\dagger}.$$

Intuitivelly, we obtain a solution $p^{\dagger}$ of a guarded specification p.

Examples

Consider $L = I^1_A$ where I is the identity monad. The adjoined free operations are list constructors $\mathit{cons}_a : 1 \to 1$ ($a \in A$) and $LX = \nu\gamma.\,(X + A \times \gamma) \cong A^\omega + A^* \times X$. Then

- $\mathit{head}(p) = \mathsf{match}\,[o, \langle x, xs\rangle] \leftarrow \mathsf{out}\,p; o \mapsto !; \langle x, xs\rangle \mapsto \mathsf{ret}\,x;$
- $\mathit{tail}(p) = \mathsf{out}^{-1}(\mathsf{match}\,[o, \langle x, xs\rangle] \leftarrow \mathsf{out}\,p; o \mapsto !; \langle x, xs\rangle \mapsto \mathsf{out}\,xs);$
- $\mathit{zip} = (\lambda\langle p, q\rangle\,.\,\mathsf{out}^{-1}(\mathsf{match}\,[o, \langle x, xs\rangle] \leftarrow \mathsf{out}\,p; o \mapsto !;$
  $\langle x, xs\rangle \mapsto \mathsf{ret}\,\mathsf{inr}\langle x, \mathsf{ret}\,\mathsf{inr}\langle q, xs\rangle\rangle) : (A^\omega)^2 \to L(\emptyset + (A^\omega)^2))^\dagger;$
- $\mathit{drop2} = (\lambda p.\ \mathsf{out}^{-1}(\mathsf{match}\,[o, \langle x, xs\rangle] \leftarrow \mathit{tail}(p); o \mapsto !;$
  $\langle x, xs\rangle \mapsto \mathsf{ret}\,\mathsf{inr}\langle x, \mathsf{ret}\,\mathsf{inr}\,xs\rangle : A^\omega \to L(\emptyset + A^\omega))^\dagger.$

Connection to GSOS

- Starting from $\Sigma(\mathrm{Id} \times B) \to B\Sigma^*$,
- we obtain $\alpha : \Sigma^*(\mathrm{Id} \times B) \xrightarrow{\text{[Klin, 2011]}} \Sigma^* \times B\Sigma^* \xrightarrow{pr_2} B\Sigma^*$;
- and then $f : \Sigma^* B^\infty \emptyset \xrightarrow{\Sigma^*\ \iota} \Sigma^*(BB^\infty\emptyset)$
$\xrightarrow{\Sigma^*\langle \iota^{-1}, id\rangle} \Sigma^*(B^\infty\emptyset \times BB^\infty\emptyset) \xrightarrow{\alpha} B\Sigma^* B^\infty\emptyset$;
- hence $\Sigma^* B^\infty\emptyset$ is a B-coalgebra and we obtain universal map

$$g : \Sigma^* \nu B \cong \Sigma^* B^\infty \emptyset \to \nu B.$$

Theorem.

$$\Sigma^* B^\infty\emptyset \xrightarrow{f} B\Sigma^* B^\infty\emptyset \xrightarrow{B(\eta\, \mathsf{inr})} B(B^\infty(\emptyset + \Sigma^* B^\infty\emptyset)) \xrightarrow{\mathsf{out}^{-1}\, \mathsf{inr}} B^\infty(\emptyset + \Sigma^* B^\infty\emptyset)$$

is guarded and $g = (\mathsf{out}^{-1}\, \mathsf{inr}\, B(\eta\, \mathsf{inr}) \circ f)^\dagger$.

Connection to GSOS

Final coalgebra $\iota : B^\infty\emptyset \longrightarrow BB^\infty\emptyset$.

- Starting from $\Sigma(\mathrm{Id} \times B) \longrightarrow B\Sigma^*$,
- we obtain $\alpha : \Sigma^*(\mathrm{Id} \times B) \xrightarrow{\text{[Klin, 2011]}} \Sigma^* \times B\Sigma^* \xrightarrow{pr_2} B\Sigma^*$;
- and then $f : \Sigma^*B^\infty\emptyset \xrightarrow{\Sigma^*\iota} \Sigma^*(BB^\infty\emptyset)$
  $\xrightarrow{\Sigma^*\langle\iota^{-1},id\rangle} \Sigma^*(B^\infty\emptyset \times BB^\infty\emptyset) \xrightarrow{\alpha} B\Sigma^*B^\infty\emptyset$;
- hence $\Sigma^*B^\infty\emptyset$ is a B-coalgebra and we obtain universal map

  $g : \Sigma^*\nu B \cong \Sigma^*B^\infty\emptyset \longrightarrow \nu B$.

Theorem.

$$\Sigma^*B^\infty\emptyset \xrightarrow{f} B\Sigma^*B^\infty\emptyset \xrightarrow{B(\eta\,\mathsf{inr})} B(B^\infty(\emptyset + \Sigma^*B^\infty\emptyset)) \xrightarrow{\mathsf{out}^{-1}\,\mathsf{inr}} B^\infty(\emptyset + \Sigma^*B^\infty\emptyset)$$

is guarded and $g = (\mathsf{out}^{-1}\,\mathsf{inr}\,B(\eta\,\mathsf{inr}) \circ f)^\dagger$.

Connection to GSOS

Final coalgebra $\iota : B^\infty\emptyset \longrightarrow BB^\infty\emptyset$.

- Starting from $\Sigma(\mathrm{Id} \times B) \longrightarrow B\Sigma^*$,
- we obtain $\alpha : \Sigma^*(\mathrm{Id} \times B) \xrightarrow{\text{[Klin, 2011]}} \Sigma^* \times B\Sigma^* \xrightarrow{pr_2} B\Sigma^*$;
- and then $f : \Sigma^* B^\infty\emptyset \xrightarrow{\Sigma^*\,\iota} \Sigma^*(BB^\infty\emptyset)$
  $\xrightarrow{\Sigma^*\langle\iota^{-1},id\rangle} \Sigma^*(B^\infty\emptyset \times BB^\infty\emptyset) \xrightarrow{\alpha} B\Sigma^* B^\infty\emptyset$;
- hence $\Sigma^* B^\infty\emptyset$ is a B-coalgebra and we obtain universal map

  $g : \Sigma^*\nu B \cong \Sigma^* B^\infty\emptyset \longrightarrow \nu B$.

Theorem.

$$\Sigma^* B^\infty\emptyset \xrightarrow{f} B\Sigma^* B^\infty\emptyset \xrightarrow{B(\eta\,\mathsf{inr})} B(B^\infty(\emptyset + \Sigma^* B^\infty\emptyset)) \xrightarrow{\mathsf{out}^{-1}\,\mathsf{inr}} B^\infty(\emptyset + \Sigma^* B^\infty\emptyset)$$

is guarded and $g = (\mathsf{out}^{-1}\,\mathsf{inr}\,B(\eta\,\mathsf{inr}) \circ f)^\dagger$.

Further Work

- Better syntax.
- Simulataneous recursion.
- Typing rules for guardedness.
- Implementation.
- Connecting to work on guarded recursion.

Lambek's Lemma

Let us write

$$\text{match}\,[x, y] \leftarrow p; y \mapsto q; z \mapsto r$$

for $(\text{do}\ z \leftarrow p; \text{case}\ z\ \text{of}\ \text{inl}\, x \mapsto q; \text{inr}\, y \mapsto r)$.

Let

$$\text{tuo}\, p = \text{init}\ t \Leftarrow p\ \text{coit}\big(\text{match}\,[x, c] \leftarrow t; x \mapsto \text{ret inl}\, x; \\ c \mapsto \text{ret inr}\, c[s \mapsto \text{out}\ s]_f\big).$$

where $p[x \mapsto q]_f = \langle \text{pr}_1\, p, y.\, q[p \,\$\, y/x]\rangle_f$.

Lemma (Lambek's Lemma). For any suitably typed p and q, $\text{out}(\text{tuo}\, p) = p$ and $\text{tuo}(\text{out}\ q) = q$.

References

J.A. Bergstra and J.W. Klop. The algebra of recursively defined processes and the algebra of regular processes. In Jan Paredaens, editor, *Automata, Languages and Programming*, volume 172 of *Lecture Notes in Computer Science*, pages 82–94. Springer Berlin Heidelberg, 1984. URL `http://dx.doi.org/10.1007/3-540-13345-3_7`.

Stephen L. Bloom and Zoltán Ésik. *Iteration theories: the equational logic of iterative processes*. Springer-Verlag New York, Inc., New York, NY, USA, 1993.

Sergey Goncharov, Christoph Rauch, and Lutz Schröder. Unguarded recursion on coinductive resumptions. In *Proc. Mathematical Foundations of Programming Semantics XXXI, MFPS 2015*, ENTCS, 2015. URL `https://www8.cs.fau.de/_media/research:papers:mfps15-elgot.pdf`.

Martin Hyland, Paul Blain Levy, Gordon D. Plotkin, and John Power. Combining algebraic effects with continuations. *Theor. Comput. Sci.*, 375(1-3):20–40, 2007.

Bartek Klin. Bialgebras for structural operational semantics: An introduction. *Theor. Comput. Sci.*, 412(38):5043–5069, 2011.

Stefan Milius, Lawrence S. Moss, and Daniel Schwencke. *Logical Methods in Computer Science*, 9(3), 2013.

Gordon Plotkin and John Power. Semantics for algebraic operations. In *Mathematical Foundations of Programming Semantics, MFPS 2001*, volume 45 of *ENTCS*, pages 332–345. Elsevier, 2001.

Jan J. M. M. Rutten. Behavioural differential equations: A coinductive calculus of streams, automata, and power series. *Theor. Comput. Sci.*, 308(1-3):1–53, 2003.

D. Turi and G. Plotkin. Towards a mathematical operational semantics. In *Logic in Computer Science*, pages 280–291. IEEE, 1997.

Tarmo Uustalu. Generalizing substitution. *ITA*, 37(4):315–336, 2003.