Bialgebraic Methods for Programming Languages

Sergey Goncharov

University of Birmingham

UNIVERSITY OF BIRMINGHAM

CMCS 2026, April 11–12, Turin

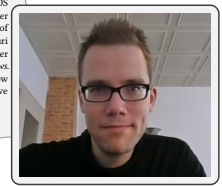
Higher-Order Mathematical Operational Semantics

Towards a Higher-Order Mathematical Operational Semantics*

SERGEY GONCHAROV, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany
STEFAN MILIUS†, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany
LUTZ SCHRÖDER‡, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany
STELIOS TSAMPAS§, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany
HENNING URBAT¶, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany

Compositionality proofs in higher-order languages are notoriously involved, and general semantic frameworks guaranteeing compositionality are hard to come by. In particular, Turi and Plotkin's bialgebraic abstract GSOS framework, which has been successfully applied to obtain off-the-shelf compositionality results for first-order languages, so far does not apply to higher-order languages. In the present work, we develop a theory of abstract GSOS specifications for higher-order languages, in effect transferring the core principles of Turi and Plotkin's framework to a higher-order setting. In our theory, the operational semantics of higher-order languages is represented by certain dinatural transformations that we term *pointed higher-order GSOS laws*. We give a general compositionality result that applies to all systems specified in this way and discuss how compositionality of the SKI calculus and the λ-calculus w.r.t. a strong variant of Abramsky's applicative bisimilarity are obtained as instances.

CCS Concepts: • **Theory of computation → Categorical semantics**; **Operational semantics**.



HO Mathematical Operational Semantics Project

- Goncharov, Milius, Schröder, Tsampas, and Urbat, "Towards a Higher-Order Mathematical Operational Semantics", POPL 2023
- Urbat, Tsampas, Goncharov, Milius, and Schröder, "Weak Similarity in Higher-Order Mathematical Operational Semantics", LICS 2023
- Goncharov, Santamaria, Schröder, Tsampas, and Urbat, "Logical Predicates in Higher-Order Mathematical Operational Semantics", FoSSaCS 2024
- Goncharov, Milius, Tsampas, and Urbat, "Bialgebraic Reasoning on Higher-Order Program Equivalence", LICS 2024
- Goncharov, Tsampas, and Urbat, "Abstract Operational Methods for Call-by-Push-Value", POPL 2025
- Goncharov, Milius, Schröder, Tsampas, and Urbat, "Bialgebraic Reasoning on Stateful Languages", ICFP 2025
- Goncharov, Partow, and Tsampas, "Big Steps in Higher-Order Mathematical Operational Semantics", ICFP 2025
- Urbat, "Higher-Order Behavioural Conformances via Fibrations", POPL 2026

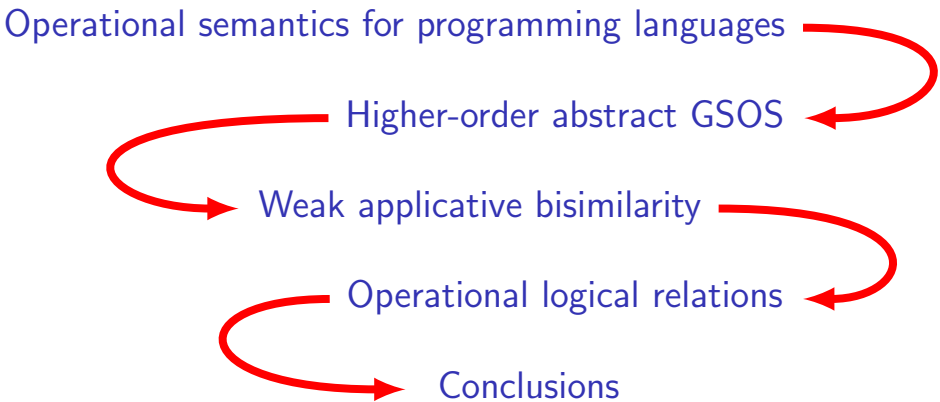
.. and rolling

Our Categorical Operational Semantics Project

- **Main Idea:** Given operational specification (set of O/S rules), devise methods/properties of the language, using category theory as leverage

  - **Methods:** Abstract logical relations, Abstract Howe's method, ...
  - **Properties:** Compositionality, safety, adequacy, equivalence of programs, semantic styles
- **Side-effect:** categorical methods $\rightsquigarrow$ functional implementation (Haskell, Agda, Rocq)

Outline


Operational semantics for programming languages
Higher-order abstract GSOS
Weak applicative bisimilarity
Operational logical relations
Conclusions

Abstract (HO-)GSOS

Operational v.s. Denotational

$$s(n) + m \to s(n + m)$$

$$0 + n \to n$$

- Operational Semantics (how programs behave?)

$$s(s(0)) + s(s(0)) \to s(s(0) + s(s(0)))$$
$$\to s(s(0 + s(s(0)))) \to s(s(s(s(0))))$$

- Denotational Semantics (what programs denote?)

$$[\![s(s(0)) + s(s(0))]\!] = [\![s(s(0))]\!] + [\![s(s(0))]\!] = 2 + 2 = 4$$

Semantics in Use

Denotational:

- ☺ Compositional by design:

$$[\![p]\!] = [\![q]\!] \Rightarrow [\![C[p]]\!] = [\![C[q]]\!]$$

for any program context C

- ☺ Mathematically rigorous and precise
- ☹ Ease to define: from hard to impossible

Operational:

- ☺ Lightweight and easy to define even for complex languages
- ☹ Nonuniform and fragile
- ☹ Hard to reason about (because of lack of compositionality)

Semantics in Use

Denotational:

☺ Compositional by design:

$$[\![p]\!] = [\![q]\!] \Rightarrow [\![C[p]]\!] = [\![C[q]]\!]$$

for any program context C

☺ Mathematically rigorous and precise

☹ Ease to define: from hard to impossible

Operational:

☺ Lightweight and easy to define even for complex languages !

☹ Nonuniform and fragile

☹ Hard to reason about (because of lack of compositionality)

Example: Call-by-Name (Lazy) λ-calculus

- Operational (small-step, call-by-name) semantics rules

$$\frac{}{(\lambda x.\, p)q \to p[q/x]} \qquad \frac{p \to p'}{pq \to p'q}$$

- Values v are irreducible closed terms (=programs): $v \not\to t$ for any t
- **Examples:**
  - $I \coloneqq \lambda x.\, x$ — value
  - $\Omega \coloneqq (\lambda x.\, xx)(\lambda x.\, xx)$ — non-value
  - $I\,(\lambda y.\, y) \;\to\; \lambda y.\, y$ — one step to a value
  - $\Omega \;\to\; \Omega \;\to\; \Omega \;\to\; \cdots$ — diverges

Contextual Equivalence

- Termination: $t\downarrow$ = "$t \to^\star v$ for some value v"
- Contextual preorder: $s \lesssim_{ctx} t$ if $C[s]\downarrow$ implies $C[t]\downarrow$ for all contexts C
  - **Example:**

    $$f \lesssim_{ctx} \lambda x.\, fx$$

    — how to prove it?
- Contextual equivalence: $s \simeq_{ctx} t$ if $s \lesssim_{ctx} t$ and $t \lesssim_{ctx} s$
  - **Example:**

    $$f \not\simeq_{ctx} \lambda x.\, fx$$

    — because $\lambda x.\, \Omega x \not\lesssim_{ctx} \Omega$: $\lambda x.\, \Omega x\downarrow$, but not $\Omega\downarrow$
- Ground contextual equivalence: in typed settings, restrict C to ground types, e.g. bool. Then again

  $$f \simeq_{ctx}^{bool} \lambda x.\, fx\ !$$

Scaling Up: Higher-Order Operational Semantics

Aspects that add/vary:

- Evaluation strategy (call-by-name v.s. call-by-value)
- Notion of observation (via types of contexts)
- Types $\rightsquigarrow$ other notions of contextual equivalence
- Computational effects (non-determinism, exceptions, store, ...)
- Other language features (recursive types, type constructors, polymorphism)

Big Challenge: Sound methods for contextual preorders (eliminating "$\forall C$")

Operational Methods are complicated, fragile and boilerplate

? Can we build a mathematical theory of (higher-order) operational semantics, abstracting and unifying these methods?

First-Order Abstract GSOS

First-Order Abstract GSOS

Turi and Plotkin's abstraction of GSOS rule format[1]:

- Signature endo-functor Σ
- Behaviour endo-functor B
- GSOS law – natural transformation $\rho_X \colon \Sigma(X \times BX) \to B(\Sigma^\star X)$

Example (Process Algebra):

- $\Sigma = \{\ |\colon 2, \emptyset\colon 0\} \cup \{a.(-)\colon 1 \mid a \in A\}$
- $BX = \mathcal{P}(A \times X)$
- GSOS law encodes rules like:

$$\frac{p \xrightarrow{a} p'}{p \mid q \xrightarrow{a} p' \mid q}$$

[1] Turi and Plotkin, "Towards a Mathematical Operational Semantics", 1997.

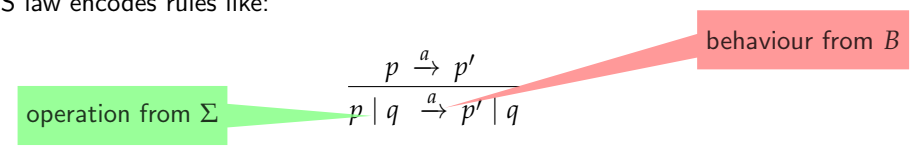
First-Order Abstract GSOS

Turi and Plotkin's abstraction of GSOS rule format[1]:

- Signature endo-functor Σ
- Behaviour endo-functor B
- GSOS law – natural transformation $\rho_X \colon \Sigma(X \times BX) \to B(\Sigma^\star X)$

Example (Process Algebra):

- $\Sigma = \{\,|\colon 2, \emptyset\colon 0\} \cup \{a.(-)\colon 1 \mid a \in A\}$
- $BX = \mathcal{P}(A \times X)$
- GSOS law encodes rules like:



[1] Turi and Plotkin, "Towards a Mathematical Operational Semantics", 1997.

First-Order Abstract GSOS

Turi and Plotkin's abstraction of GSOS rule format[1]:

- Signature endo-functor Σ
- Behaviour endo-functor B
- GSOS law – natural transformation $\rho_X \colon \Sigma(X \times BX) \to B(\Sigma^\star X)$

Example (Process Algebra):

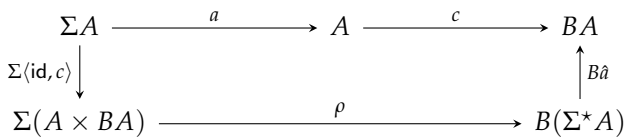
- $\Sigma = \{\ |\colon 2, \emptyset\colon 0\} \cup \{a.(-)\colon 1 \mid a \in A\}$
- $BX = \mathcal{P}(A \times X)$
- GSOS law encodes rules like:

$$\frac{p \xrightarrow{a} p'}{p \mid q \xrightarrow{a} p' \mid q}$$

[1] Turi and Plotkin, "Towards a Mathematical Operational Semantics", 1997.

Bialgebras

- ρ-bialgebra interprets operations by an algebra $a\colon \Sigma A \to A$ and provides a behaviour via a coalgebra $c\colon A \to BA$ such that

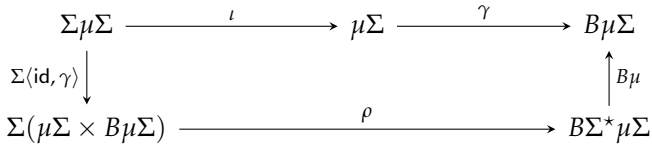


  where $\hat{a}\colon \Sigma^\star A \to A$ is inductive extension of a

- ρ-bialgebras form a category

Operational Model

- Initial algebra $(\mu\Sigma, \iota\colon \Sigma\mu\Sigma \to \mu\Sigma)$ extends to initial bi-algebra



  where $\mu\Sigma \xrightarrow{\gamma} B\mu\Sigma$ — operational model (= Coalgebraic semantics)
- Analogously: final coalgebra νB extends to final bialgebra = denotational model
- Abstract behaviour: unique bialgebra morphism $[\![-]\!]_\rho\colon \mu\Sigma \to \nu B$
- Full abstraction: p and q are behaviourally equal iff $[\![p]\!]_\rho = [\![q]\!]_\rho$
  - **Corollary:** Behavioral equivalence is congruence

First-Order GSOS: Summary

Theory of first-order GSOS takes Σ, B, ρ as input parameters, and produces

- ☺ operational semantics $\gamma\colon \mu\Sigma \to B(\mu\Sigma)$ (operational model)
- ☺ notion of program equivalence $\sim \subseteq \mu\Sigma \times \mu\Sigma$ (strong bisimilarity)
- ☺ generic compositionality: $p \sim q \Rightarrow C[p] \sim C[q]$ for any context

But

- ☹ $\sim$ is too fine-grained for programming languages
- ☹ first-order $\subsetneq$ higher-order $\rightsquigarrow$ no λ-calculus

First-Order GSOS: Summary

Theory of first-order GSOS takes Σ, B, ρ as input parameters, and produces

- ☺ operational semantics $\gamma\colon \mu\Sigma \to B(\mu\Sigma)$ (operational model)
- ☺ notion of program equivalence $\sim \subseteq \mu\Sigma \times \mu\Sigma$ (strong bisimilarity)
- ☺ generic compositionality: $p \sim q \Rightarrow C[p] \sim C[q]$ for any context

But

- ☹ $\sim$ is too fine-grained for programming languages
- ☹ first-order $\subsetneq$ higher-order $\rightsquigarrow$ no λ-calculus ←

Higher-Order Abstract GSOS

Extended Combinatory Logic xCL

xCL language (variant of combinatory logic)

- K $(=\lambda p.\lambda q.\, p)$
- S $(=\lambda p.\lambda q.\lambda r.\, (p \cdot r) \cdot (q \cdot r))$
- plus S', S'' and K' for partially reduced terms

$$K \xrightarrow{p} K'(p) \qquad K'(p) \xrightarrow{q} p \qquad S \xrightarrow{p} S'(p) \qquad S'(p) \xrightarrow{q} S''(p,q)$$

$$S''(p,q) \xrightarrow{r} (p \cdot r) \cdot (q \cdot r) \qquad \frac{p \to p'}{p \cdot q \to p' \cdot q} \qquad \frac{p \xrightarrow{q} p'}{p \cdot q \to p'}$$

  - **Example:** $Spqr \to S'(p)qr \to S''(p,q)r \to (pr)(qr)$
- This is very similar to original GSOS, but it is not

Extended Combinatory Logic xCL

xCL language (variant of combinatory logic)

- K $(=\lambda p.\lambda q.\, p)$
- S $(=\lambda p.\lambda q.\lambda r.\, (p \cdot r) \cdot (q \cdot r))$
- plus S', S'' and K' for partially reduced terms

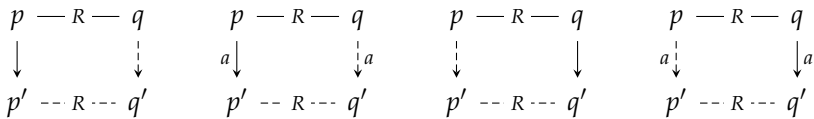
$$K \xrightarrow{p} K'(p) \qquad K'(p) \xrightarrow{q} p \qquad S \xrightarrow{p} S'(p) \qquad S'(p) \xrightarrow{q} S''(p,q)$$

$$S''(p,q) \xrightarrow{r} (p \cdot r) \cdot (q \cdot r) \qquad \frac{p \to p'}{p \cdot q \to p' \cdot q} \qquad \frac{p \xrightarrow{q} p'}{p \cdot q \to p'} \quad \text{!}$$

  - **Example:** $Spqr \to S'(p)qr \to S''(p,q)r \to (pr)(qr)$
- This is very similar to original GSOS, but it is not

Congruence for xCL

- $\Sigma = \{S, K, S', S'', K'\}$, $\mu\Sigma$ is the set of xCL-terms
- Strong applicative bisimilarity $\sim$ on $\mu\Sigma$: greatest relation $R \subseteq \mu\Sigma \times \mu\Sigma$ such that



Proposition: $\sim$ is a Σ-congruence

Proof Idea: For any $R \subseteq \mu\Sigma \times \mu\Sigma$, define

$$\widehat{R} = \{(C[s], C[t]) \in \mu\Sigma \times \mu\Sigma \mid C \text{ a (linear) context}, sRt\}$$

Show that $\widehat{\sim}^\star \subseteq \sim$. Essentially, this is up-to-transitive-congruence

Higher-Order Abstract GSOS

Definition: Higher-order GSOS law in category $\mathbf{C}$ consists of

- Signature functor $\Sigma\colon \mathbf{C} \to \mathbf{C}$
- Behaviour functor $B\colon \mathbf{C}^{\mathrm{op}} \times \mathbf{C} \to \mathbf{C}$
- Family $\rho_{X,Y}\colon \Sigma(X \times B(X, Y)) \to B(X, \Sigma^\star(X + Y))$ natural in Y and dinatural in X

For xCL:

- $\mathbf{C}$ – category of sets
- $\Sigma X = \coprod_{f \in Ops} X^{arity(f)}$, $Ops = \{S, S', S'', K, K', I, \cdot\}$
- $B(X, Y) = Y^X + Y$
- ρ is induced by rules of operational semantics (next slide)

For λ-calculus: $\mathbf{C} = [\mathbb{F}, \mathbf{Set}]$, $\Sigma = V + \delta X + X^2$, and ρ must be V-pointed[2]

[2] Fiore, Plotkin, and Turi, "Abstract Syntax and Variable Binding", 1999.

xCL as HO-GSOS

Rules

$$\frac{p \to p'}{p \cdot q \to p' \cdot q} \qquad \frac{p \xrightarrow{q} p'}{p \cdot q \to p'}$$

correspond to

$$\rho((p, p') \cdot (q, \_)) = p' \cdot q$$
$$\rho((p, f) \cdot (q, \_)) = f(q)$$

$$\left(\rho_{X,Y} \colon \Sigma(X \times (Y + Y^X)) \to \Sigma^\star(X + Y) + (\Sigma^\star(X + Y))^X\right)$$

Representing Rules

For $\mathbf{C} = Set$, $\Sigma X = \coprod_{f \in Ops} X^{arity(f)}$, $B(X,Y) = Y^X + Y$, HO-GSOS precisely correspond to sets of rules of the form:

$$\frac{(x_j \to y_j)_{j \in W} \qquad (x_i \xrightarrow{z} y_i^z)_{i \in \overline{W}, z \in \{x_1, \ldots, x_n\}}}{f(x_1, \ldots, x_n) \to t}$$

or

$$\frac{(x_j \to y_j)_{j \in W} \qquad (x_i \xrightarrow{z} y_i^z)_{i \in \overline{W}, z \in \{x, x_1, \ldots, x_n\}}}{f(x_1, \ldots, x_n) \xrightarrow{x} t}$$

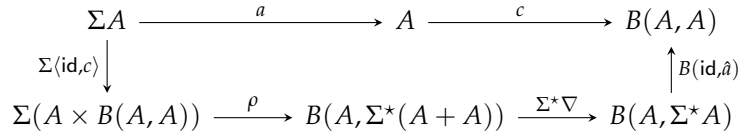
($W \subseteq \{1, \ldots, n\}$, $\overline{W} = \{1, \ldots, n\} \setminus W$)

Proof Idea: Yoneda-style argument

Generally, HO-GSOS vastly abstract this situation

Higher-Order Bialgebras

- Higher-order ρ-bialgebra is triple $(A, a\colon \Sigma A \to A, c\colon A \to B(A,A))$ such that diagram



  commutes

- Initial bialgebra $(\mu\Sigma, \iota, \gamma)$ exists with operational model $\gamma\colon \mu\Sigma \to B(\mu\Sigma, \mu\Sigma)$
- Operational equivalence is kernel of map

$$\mathrm{coit}(\gamma)\colon \mu\Sigma \to \nu\gamma.\, B(\mu\Sigma, \gamma)$$

- But final bialgebra need not exist!

Abstract Congruence for HO-GSOS

Theorem[3]**:** Given that

1. $\mathbf{C}$ is regular (roughly, $\mathbf{C}$ admits a good notion of image factorization)
2. Σ preserves reflexive coequalizers (in particular, if Σ is finitary)
3. B preserves monomorphisms in both arguments (it sends epis on the first argument to monos)

the kernel pair $\mu\Sigma \to \mu\Sigma_{\sim}$ of the final coalgebra map

$$\mathrm{coit}(\gamma)\colon \mu\Sigma \to \nu\gamma.\, B(\mu\Sigma, \gamma)$$

is a congruence

Proof Idea: Abstraction of up-to-transitive-congruence proof

[3] Goncharov, Milius, Schröder, Tsampas, and Urbat, "Towards a Higher-Order Mathematical Operational Semantics", 2023.

Weak Applicative Bisimilarity

Weak v.s. Strong

- Strong applicative bisimilarity is sound for contextual equivalence:

$$t \sim s \quad \Longrightarrow \quad \forall C.\, C[t] \sim C[s] \quad \Longrightarrow \quad t \simeq_{ctx} s$$

- But it is too fine-grained, e.g. $I \approx II$, because $I \not\to$, but $II \to I$
- Remedy: weak applicative bisimilarity:
  1. Let $\Longrightarrow \coloneqq (\to^\star)$, $\overset{t}{\Longrightarrow} \coloneqq (\Longrightarrow ; \overset{t}{\to})$
  2. Derive weak similarity/bisimilarity $\lesssim / \approx$ as strong similarity/bisimilarity for $\Longrightarrow$
- **Challenge:** Prove that $\approx$ and $\lesssim$ are congruences

Howe's Method

Howe's closure of relation $R \subseteq \mu\Sigma \times \mu\Sigma$ is defined inductively with rule

$$\frac{p\ (R^H)^\Sigma\ r \qquad r\ R\ q}{p\ R^H\ q}$$

where R^Σ — Σ-closure of R

Properties: If R is preorder then

$$R \subseteq R^H \qquad (R^H)^\Sigma \subseteq R^H \quad \text{(congruence)} \qquad \Delta \subseteq R^H \quad \text{(reflexivity)}$$

(but, R^H need not be transitive!)

Howe's Method: show that $\lesssim^H$ is simulation. Then $\lesssim$ is congruence:

$$\lesssim^\Sigma \quad \subseteq \quad (\lesssim^H)^\Sigma \quad \subseteq \quad \lesssim^H \quad \subseteq \quad \lesssim$$

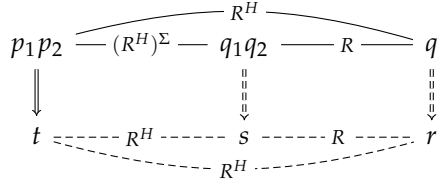
Howe's Method for xCL

Proposition: $\lesssim^H$ is simulation

Proof Idea: Strengthen coinduction invariant to:

Whenever	$pR^Hq,$	if	$p \Longrightarrow p'$	then	$\exists q'.q \Longrightarrow q' \wedge p' \ R^H \ q',$ and
		if	$p \overset{r}{\Longrightarrow} t$ and	$r \ R^H \ r'$	then $\exists s.q \overset{r'}{\Longrightarrow} s \wedge t \ R^H \ s$

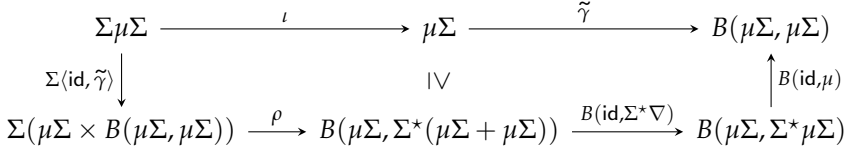
Example clause:



where $p_1 \overset{p_2}{\Longrightarrow} t$ and $q_1 \overset{q_2}{\Longrightarrow} s$

From Bialgebras to Lax Bialgebras

- Recall: $(\mu\Sigma, \iota, \gamma)$ is initial ρ-bialgebra, where $\gamma\colon \mu\Sigma \to B(\mu\Sigma, \mu\Sigma)$ encodes strong transitions $\to$
- Let $\tilde{\gamma}\colon \mu\Sigma \to B(\mu\Sigma, \mu\Sigma)$ correspond to weak transitions $\Rightarrow$
- However, $(\mu\Sigma, \iota, \tilde{\gamma})$ is not ρ-bialgebra, but only a lax ρ-bialgebra (!)



— originally introduced in first-order setting[4]

[4] Bonchi, Petrisan, Pous, and Rot, "Lax Bialgebras and Up-To Techniques for Weak Bisimulations", 2015.

Howe's Method Abstractly[5]

Setting:

- HO-GSOS law $\rho \rightsquigarrow \rho$-bialgeba $(\mu\Sigma, \iota, \gamma)$ for strong transitions
- Lax ρ-bialgebra $(\mu\Sigma, \iota, \tilde{\gamma})$ for weak transitions

Key Assumptions:

- Factorization system on $\mathbf{C}$, enabling suitable calculus of relations
- Relational lifting of B:

$$(R \subseteq X \times X, S \subseteq Y \times Y) \quad \mapsto \quad \bar{B}(R,S) \subseteq B(X,Y) \times B(X,Y)$$

- HO-GSOS law ρ lifts to relations

Key Lemma: Abstract Howe's closure of weak similarity is simulation

Corollary: Weak similarity is congruence

[5] Urbat, Tsampas, Goncharov, Milius, and Schröder, "Weak Similarity in Higher-Order Mathematical Operational Semantics", 2023.

Operational Logical Relations

Invariants: Algebraic and Coalgebraic

- Given $F\colon \mathcal{C} \to \mathcal{C}$, lifting $\overline{F}\colon \mathbf{Pred}(\mathcal{C}) \to \mathbf{Pred}(\mathcal{C})$, and algebra $a\colon FA \to A$ algebraic invariant[6] is predicate $P \rightarrowtail A$ that satisfies

$$\overline{F}P \leq a^{-1}[P] \qquad (\iff a[\overline{F}P] \leq P)$$

- Analogously, for relations: $\overline{F}\colon \mathbf{Rel}(\mathcal{C}) \to \mathbf{Rel}(\mathcal{C})$

$$\overline{F}R \leq (a \times a)^{-1}[R] \qquad (\iff (a \times a)[\overline{F}R] \leq R)$$

- **Example:** $R \subseteq A \times A$ is congruence for Σ-algebra if it is Σ-invariant
- Dually, coinductive invariant: given coalgebra $c\colon C \to FC$, and $\overline{F}\colon \mathbf{Pred}(\mathcal{C}) \to \mathbf{Pred}(\mathcal{C})$

$$P \leq c^{-1}[\overline{F}(P)]$$

[6] Jacobs, *Introduction to Coalgebra: Towards Mathematics of States and Observation*, 2016.

Coalgebraic Invariants under Mixed Variance

Let $B\colon \mathcal{C}^{\mathrm{op}} \times \mathcal{C} \to \mathcal{C}$ with a relation lifting $\overline{B}$, and let $c, \widetilde{c}\colon X \to B(X,X)$ be coalgebras

- Applicative (bi)simulation for $(c, \widetilde{c})$ is a relation $R \rightarrowtail X \times X$ such that

$$R \leq (c \times \widetilde{c})^{-1}[\overline{B}(\Delta, R)]$$

- Logical relation for $(c, \widetilde{c})$ is a relation $R \rightarrowtail X \times X$ such that

$$R \leq (c \times \widetilde{c})^{-1}[\overline{B}(R, R)]$$

- Logical predicate for c is a predicate $P \rightarrowtail X$ such that

$$P \leq c^{-1}[\overline{B}(P, P)]$$

In contrast to applicative bisimulation, it is not clear how to define logical relation/predicate

Strong Normalization via Logical Predicates

- xCL is not strongly normalizing, but a typed version in $\mathbf{Set}^{Ty}$ is! How can we prove it?
- **Proof Strategy:**
  1. Start with strong normalization predicate $P \rightarrowtail \mu\Sigma$
  2. Strengthen P to $\Box P$ (Jacobs' henceforth operator)
  3. Prove that $\overline{\Sigma}(\Box P) \leq \iota^{-1}[P]$ implies $\overline{\Sigma}(\Box P) \leq \iota^{-1}[\Box P]$ ($\Box P$ is $\overline{\Sigma}$-invariant)
  4. Note that $\Box P$ is $\overline{\Sigma}$ iff $\Box P = \mu\Sigma$ (induction) $\implies$ $\Box P$, in particular, P always true!
- Step 3. — local inspection of operational semantics rules. But how to define $\Box P$?
- Two-stage procedure[7]:
  1. Define $\Box(S, P)$ relative of S as greatest fixpoint:

$$\Box(S, P) = \nu G.\, P \wedge c^{-1}[\overline{B}(S, G)]$$

  2. $\Box P$ — unique solution of $\Box P = \Box(\Box P, P)$. Existence by Banach fixpoint theorem using metric properties of $\overline{B}$, induced by typing

[7] Goncharov, Santamaria, Schröder, Tsampas, and Urbat, "Logical Predicates in Higher-Order Mathematical Operational Semantics", 2024.

Step-Indexing

- To construct $\square$ we essentially abstracted terminating properties of semantics
- But we can use logical relations of different kind to reason about non-terminating programs: step-indexed logical relations
- Given $R \rightarrowtail X \times X$, define $\square R$ by transfinite recursion instead:

$$\begin{aligned} \square^0 R &= R \\ \square^{\alpha+1} R &= \square^\alpha R \wedge (c \times \widetilde{c})^{-1}[\overline{B}(\square^\alpha R, \square^\alpha R)] \\ \square^\alpha R &= \bigwedge_{\beta<\alpha} \square^\beta R \end{aligned}$$

  $(\square^\alpha R)_\alpha$ forms descending chain, so it must stabilize at some ordinal: $\square^\nu R = \bigcap_\alpha \square^\alpha R$
- No Knaster-Tarski because $\overline{B}$ is antitone in first argument!

Theorem:[8] Under general assumptions (liftable ρ, c is bialgebra, $\widetilde{c}$ is lax-bialgebra, ...), every $\square^\alpha \top$ is congruence

[8] Goncharov, Milius, Tsampas, and Urbat, "Bialgebraic Reasoning on Higher-Order Program Equivalence", 2024.

Step-Indexing for xCL

$\Box\top$ for combinatory logic is the inductively defined family $(\Box^\alpha\top \subseteq \mu\Sigma \times \mu\Sigma)_{\alpha\leq\omega}$:

$$\Box^0\top = \top, \qquad \Box^{n+1}\top = \Box^n\top \cap \mathcal{E}(\Box^n\top) \cap \mathcal{V}(\Box^n\top, \Box^n\top), \quad \Box^\omega\top = \bigcap\nolimits_{n<\omega} \Box^n\top$$

where $\mathcal{E}$ and $\mathcal{V}$ are relation transformers:

$$\begin{aligned}\mathcal{E}(R) &= \{(t,s) \mid \text{if } t \to t' \text{ then } \exists s'.\, s \Rightarrow s' \wedge (t',s') \in R\}\\ \mathcal{V}(Q,R) &= \{(t,s) \mid \text{for all } r_1, r_2,\ (r_1,r_2) \in Q,\\ &\qquad\qquad \text{if } t \xrightarrow{r_1} t' \text{ then } \exists s'.\, s \overset{r_2}{\Rightarrow} s' \wedge (t',s') \in R\}\end{aligned}$$

As a slogan: *"related programs applied to related arguments produce related results"*

Contextual Preorders in the Abstract

Given $O, R \subseteq \mu\Sigma \times \mu\Sigma$, R is O-adequate if $R \subseteq O$; Contextual preorder w.r.t. O:

$$\lesssim^O = \text{greatest } O\text{-adequate congruence}$$

Theorem: $\Box^\nu \top$ is O-adequate then $\Box^\nu \top \subseteq \lesssim^O$

Example: $f \lesssim_{ctx} \lambda x.\, fx \iff f \lesssim^O \lambda x.\, fx$ where $O = \{(t,s) \mid t{\downarrow} \Rightarrow s{\downarrow}\}$ follows from $(f, \lambda x.\, fx) \in \Box^\alpha \top$ (easy)

But we can also do more: we can redefine lax-bialgebra:

$$t \overset{r}{\Rightarrow} s : \qquad (\exists t'.\, t \Rightarrow t' \wedge t' \overset{r}{\rightarrow} s) \vee (\exists t'.\, t \Rightarrow t' \wedge s = t'\ r)$$

Then $\Box^\nu \top$ is O_{bool}-adequate for

$$O_{bool} = \{(t,s) \in \mu\Sigma_{bool} \times \mu\Sigma_{bool} \mid t{\downarrow} \Rightarrow s{\downarrow}\} \qquad \text{and} \qquad O_\tau = \top \quad \text{for} \quad \tau \neq bool$$

and we can recover $f \simeq_{ctx}^{bool} \lambda x.\, fx \iff f \simeq^{O^{bool}} \lambda x.\, fx$!

Conclusions

Weak Applicative Bisimilarity:

- Abstract framework via lax ρ-bialgebras and higher-order GSOS laws
- Congruence via abstract Howe's method; soundness for contextual preorder
- Instances: combinatory logic, call-by-name λ-calculus, ...

Operational Logical Relations:

- Abstract logical predicates $\Box P$ for HO-GSOS laws; application: strong normalization
- Abstract step-indexed logical relations $(\Box^\alpha \top)_\alpha$; each $\Box^\alpha \top$ is a congruence
- Limit $\Box^\nu \top$ sound for contextual preorder; flexible: ground contexts, recursive types, ...

Further Work:

- Polymorphic languages, and free theorems
- Improving sufficient conditions in terms of rule formats
- Connection to big-step/small-step equivalence, trace semantics
- Understanding relation between $\Box^\nu \top$, $\Box \top$ and $\Box(\Delta, \top)$ (applicative bisimilarity)

Thank You for Your Attention!

Higher-Order Abstract GSOS

Categorical Framework for Higher-Order Operational Semantics

Language

Signature $\cong$ Endofunctor $\Sigma\colon \mathbf{C} \to \mathbf{C}$ on a category $\mathbf{C}$, e.g.:

- $\mathbf{C} = \mathbf{Set}$, $\Sigma = \{0/0, a_i/0, +/2, \cdot/2\}$
- $\mathbf{C}$ = "nominal sets", $\Sigma X = \mathbb{A} + [\mathbb{A}]X + X \times X$

Behaviour

Behaviour = Mixed-variance functor $B\colon \mathbf{C}^{\mathrm{op}} \times \mathbf{C} \to \mathbf{C}$, e.g.:

- $B(X, Y) = Y^X + Y$ (deterministic)
- $B(X, Y) = \mathcal{P}_{\omega}(Y^X + Y)$ (non-deterministic)

HO Specification in GSOS Format

$\frac{t \xrightarrow{a} t' \quad s \xrightarrow{a} s'}{t\,\mathsf{l}\,s \xrightarrow{\mathsf{l}} t'\,\mathsf{l}\,s'} \qquad \frac{t \xrightarrow{a} t'}{t\,\mathsf{s} \longrightarrow t'} \qquad \frac{(\lambda x.\, t) \xrightarrow{s} t\{s/x\}}{}$

$\Longrightarrow$ **Distributive law** ρ of Σ over B

Higher-Order Bialgebraic Semantics

Transition semantics is a unique solution $\gamma\colon \mu\Sigma \to B(\mu\Sigma, \mu\Sigma)$:

$$\begin{array}{ccccc} \Sigma\mu\Sigma & \xrightarrow{\iota} & \mu\Sigma & \xrightarrow{\gamma} & B(\mu\Sigma, \mu\Sigma) \\ {\scriptstyle \Sigma(\mathrm{id},\gamma)}\downarrow & & & & \uparrow{\scriptstyle B(\mathrm{id},\iota)} \\ \Sigma(\mu\Sigma \times B(\mu\Sigma, \mu\Sigma)) & \xrightarrow{\rho} & B(\mu\Sigma, \Sigma^*(\mu\Sigma + \mu\Sigma)) & \xrightarrow{B(\mathrm{id}, \Sigma^* \nabla)} & B(\mu\Sigma, \Sigma^* \mu\Sigma) \end{array}$$

Generic Strong Applicative Bisimulation

Coalgebraic notion of **strong applicative bisimilarity** $\sim$ on initial Σ-algebra $\mu\Sigma$ (=algebra of programs) as a pullback

$$\begin{array}{ccc} \sim & \longrightarrow & \mu\Sigma \\ \downarrow & \lrcorner & \downarrow{\scriptstyle \mathrm{coit}\,\gamma} \\ \mu\Sigma & \xrightarrow{\mathrm{coit}\,\gamma} & \nu\gamma.\, B(\mu\Sigma, \gamma) \end{array}$$

Central Result: Compositionality for Free

Under certain general assumptions, $\sim$ is a congruence

References I


Bonchi, Filippo, Daniela Petrisan, Damien Pous, and Jurriaan Rot. "Lax Bialgebras and Up-To Techniques for Weak Bisimulations". In: *26th International Conference on Concurrency Theory, CONCUR 2015, Madrid, Spain, September 1.4, 2015*. 2015, pp. 240–253. DOI: 10.4230/LIPIcs.CONCUR.2015.240.


Fiore, Marcelo P., Gordon D. Plotkin, and Daniele Turi. "Abstract Syntax and Variable Binding". In: *14th Annual IEEE Symposium on Logic in Computer Science, Trento, Italy, July 2-5, 1999*. IEEE Computer Society, 1999, pp. 193–202. URL: https://doi.org/10.1109/LICS.1999.782615.


Goncharov, Sergey, Stefan Milius, Lutz Schröder, Stelios Tsampas, and Henning Urbat. "Bialgebraic Reasoning on Stateful Languages". In: *Proc. 30th ACM SIGPLAN International Conference on Functional Programming (ICFP 2025)*. Vol. 9. Association for Computing Machinery, 2025.


Goncharov, Sergey, Stefan Milius, Lutz Schröder, Stelios Tsampas, and Henning Urbat. "Towards a Higher-Order Mathematical Operational Semantics". In: *In 50th ACM SIGPLAN Symposium on Principles of Programming Languages (POPL 2023)* (2023).

References II

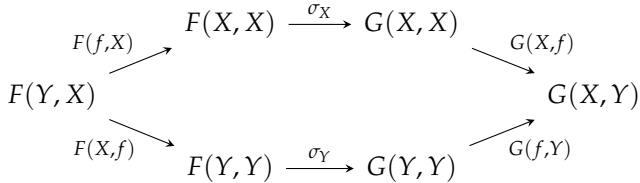
- Goncharov, Sergey, Stefan Milius, Stelios Tsampas, and Henning Urbat. "Bialgebraic Reasoning on Higher-Order Program Equivalence". In: *39th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS 2024)*. Association for Computing Machinery, 2024.
- Goncharov, Sergey, Pouya Partow, and Stelios Tsampas. "Big Steps in Higher-Order Mathematical Operational Semantics". In: *Proc. 30th ACM SIGPLAN International Conference on Functional Programming (ICFP 2025)*. Vol. 9. Association for Computing Machinery, 2025.
- Goncharov, Sergey, Alessio Santamaria, Lutz Schröder, Stelios Tsampas, and Henning Urbat. "Logical Predicates in Higher-Order Mathematical Operational Semantics". In: *Foundations of Software Science and Computation Structures (FoSSaCS 2024)*. Ed. by Naoki Kobayashi and James Worrell. Vol. 14575. LNCS. Springer, 2024, pp. 47–69.
- Goncharov, Sergey, Stelios Tsampas, and Henning Urbat. "Abstract Operational Methods for Call-by-Push-Value". In: *52th ACM SIGPLAN Symposium on Principles of Programming Languages (POPL 2025)*. Proc. ACM Program. Lang. (2025).
- Jacobs, Bart. *Introduction to Coalgebra: Towards Mathematics of States and Observation*. Vol. 59. Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 2016. DOI: 10.1017/CBO9781316823187.

References III

- Turi, D. and G. Plotkin. "Towards a Mathematical Operational Semantics". In: *Logic in Computer Science*. IEEE. 1997, pp. 280–291.
- Urbat, Henning. "Higher-Order Behavioural Conformances via Fibrations". In: *Proc. ACM Program. Lang.* 10.POPL (2026), pp. 1586–1614. DOI: 10.1145/3776697. URL: https://doi.org/10.1145/3776697.
- Urbat, Henning, Stelios Tsampas, Sergey Goncharov, Stefan Milius, and Lutz Schröder. "Weak Similarity in Higher-Order Mathematical Operational Semantics". In: *Logic in Computer Science (LICS 2023)*. Ed. by Igor Walukiewicz. (in press). IEEE Computer Society Press, 2023.

Dinaturality

A dinatural transformation from $F\colon \mathbf{C}^{\mathrm{op}} \times \mathbf{C} \to \mathbf{D}$ to $G\colon \mathbf{C}^{\mathrm{op}} \times \mathbf{C} \to \mathbf{D}$ is a family $(\sigma_X\colon F(X,X) \to G(X,X))_{X \in \mathbf{C}}$, such that



for every $f\colon X \to Y$

Example: apply$\colon X^Y \times Y \to X$

Lambda-Calculus: Way of Combinators

- Let $\Lambda(n)$ be λ-terms with free variables from $\{1,\dots,n\}$ modulo α-equivalence
- $\Sigma X = \coprod_{n\in\mathbb{N}} \Lambda(n+1)\times X^n + X^2$, so

  $(f, t_1, \dots, t_n)$ represents $\lambda(n+1).\, f[t_1/1, \dots, t_n/n]$
- $B(X,Y) = Y + Y^X$
- For every $f \in \Lambda(n+1)$ let $\lceil f \rceil \in \Sigma^\star(n+2)$ be obtained from f by recursively replacing topmost $\lambda x.\, t$ with $t[(n+2)/x] \in \Lambda(n+2)$

Examples: $S = \lambda yz.\,(1z)(yz) \in \Lambda(1)$, $S' = \lceil S \rceil = \lambda z.\,(1z)(2z) \in \Lambda(2)$,
$S'' = \lceil S' \rceil = (13)(23) \in \Sigma^\star(3)$

Way of Combinators, Cont'd

- The rules

$$\frac{}{f(x_1,\ldots,x_n) \xrightarrow{t} \lceil f \rceil[x_1/1\ldots,x_n/n,t/(n+1)]} \quad (f \in \Lambda(n+1))$$

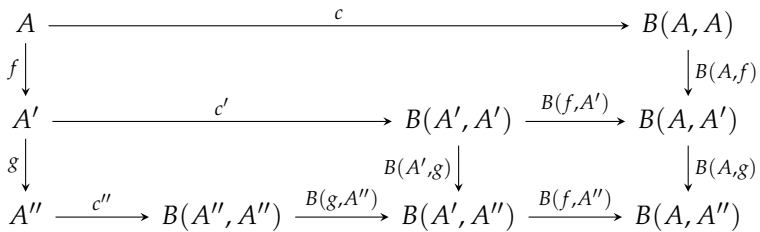
$$\frac{p \to p'}{pq \to p'q} \qquad \frac{p \xrightarrow{q} p'}{pq \to p'}$$

  then mimic the standard call-by-name semantics of untyped λ-calculus

- Hence, we obtain a congruence result for the lazy λ-calculus circumventing presheave semantics!

Bialgebras Form a Category

If $f\colon A \to A'$ and $g\colon A' \to A''$ are ρ-bialgebra morphisms then so is the composition gf, for the diagram



obviously commutes.

Lambda-Calculus

- Operational semantics rules

$$\frac{s \to s'}{s\ t \to s'\ t} \qquad\qquad \frac{}{(\lambda x.s)\ t \to s[t/x]}$$

- $\mathbf{C} = \mathbf{Set}^{\mathbb{F}}$, where $\mathbb{F}$ is the category of finite cardinals

$$\Sigma\colon \mathbf{C} \to \mathbf{C}, \qquad \Sigma X = V + \delta X + X \times X,$$
$$B\colon \mathbf{C}^{\mathrm{op}} \times \mathbf{C} \to \mathbf{C}, \qquad B(X,Y) = \langle\!\langle X, Y \rangle\!\rangle \times (Y + Y^X + 1)$$

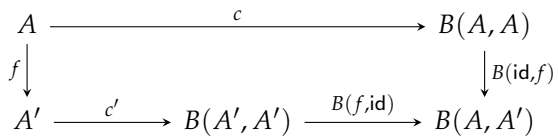
  where Y^X is exponent in $\mathbf{Set}^{\mathbb{F}}$, V is the presheaf of variables $\mathbf{Set}^{\mathbb{F}}(n) = n$, $(\delta X)(n) = X(n+1)$, $\langle\!\langle X, Y \rangle\!\rangle(n) = \mathbf{Set}^{\mathbb{F}}(X^n, Y)$

- $\mu\Sigma$ is the presheaf $\Lambda \in \mathbf{Set}^{\mathbb{F}}$ of λ-terms over n free variables
- H/O GSOS law is pointed: $\rho_{X,Y}\colon \Sigma(jX \times B(jX,Y)) \to B(jX, \Sigma^\star(jX + Y))$[9]

[9] Fiore, Plotkin, and Turi, "Abstract Syntax and Variable Binding", 1999.

Higher-Order Bialgebras Morphisms

A ρ-bialgeba morphism from (A, a, c) to (A', a', c') is a Σ-algebra morphism $f\colon A \to A'$, such that the diagram



commutes

Example: Proving "η-Law"

- Recall: $t \overset{r}{\Rightarrow} s = (\exists t'.\, t \Rightarrow t' \wedge t' \overset{r}{\rightarrow} s) \vee (\exists t'.\, t \Rightarrow t' \wedge s = t'\ r)$

$$\Box^{n+1}\top = \Box^n\top \cap \mathcal{E}(\Box^n\top) \cap \mathcal{V}(\Box^n\top, \Box^n\top)$$
$$\mathcal{E}(\Box^n\top) = \{(t,s) \mid \text{if } t \rightarrow t' \text{ then } \exists s'.\, s \Rightarrow s' \wedge (t',s') \in \Box^n\top\}$$
$$\mathcal{V}(\Box^n\top, \Box^n\top) = \{(t,s) \mid \text{for all } r_1, r_2,\ (r_1,r_2) \in \Box^n\top,$$
$$\text{if } t \overset{r_1}{\rightarrow} t' \text{ then } \exists s'.\, s \overset{r_2}{\Rightarrow} s' \wedge (t',s') \in \Box^n\top\}$$

- Proof of $(S \cdot (K \cdot I) \cdot f, f) \in \Box^n\top$ by induction on n, in particular:

$$\begin{array}{ccccccccc}
S\cdot(K\cdot I)\cdot f & \rightarrow & S'(K\cdot I)\cdot f & \rightarrow & S''(K\cdot I, f) & \overset{t}{\rightarrow} & (K\cdot I\cdot t)\cdot(f\cdot t) & \overset{\star}{\rightarrow} & f\cdot t \\
\Box^n\top & & \Box^{n-1}\top & & \Box^{n-2}\top & & \Box^{n-3}\top & & \Box^{n-6}\top \\
f & \Longrightarrow & f & \Longrightarrow & f & \overset{t'}{\longrightarrow} & f\cdot t' & \Longrightarrow & f\cdot t'
\end{array}$$