Locally Final Coalgebras in Denotational Semantics

Sergey Goncharov[1], Marco Peressotti[2], Stelios Tsampas[3], Henning Urbat[4], and Stefano Volpe[5]

[1] Birmingham University, Birmingham, United Kingdom
s.goncharovbham@bham.ac.uk
[2] Southern Denmark University, Odense, Denmark
peressotti@imada.sdu.dk
[3] Friedrich-Alexander-Universität Erlangen-Nürnberg, Erlangen, Germany
henning.urbat@fau.de
[4] Southern Denmark University, Odense, Denmark
stelios@imada.sdu.dk
[5] Southern Denmark University, Odense, Denmark
stefano@imada.sdu.dk

Formal semantics aims to assign a precise mathematical meaning to every program of a given language. Two widely used flavours are *operational semantics*, which specifies how a program p behaves when it is executed, and *denotational semantics*, which associates to a program p some object $[\![p]\!]$ representing its abstract content. A prerequisite for a denotational semantics to be useful and well-behaved is *adequacy:* if two programs p and q are *denotationally equivalent*, i.e. $[\![p]\!] = [\![q]\!]$, then they are *behaviourally equivalent*, that is, their observable behaviour w.r.t. the *operational* semantics is indistinguishable. Adequacy ensures that the denotational and operational semantics are compatible, yet proving it tends to be cumbersome. For this reason, there has been longstanding interest in *categorical* accounts which mitigate this complexity via the power of abstraction.

Abstract GSOS Turi and Plotkin [TP97] proposed an elegant *bialgebraic* approach to operational and denotational semantics. It rests on the key observation that the operational rules of many programming or process languages can be presented as a *GSOS law*, a natural transformation

$$\varrho_X \colon \Sigma(X \times BX) \to B\Sigma^\star X$$

parametric in a *syntax functor* $\Sigma \colon \mathbb{C} \to \mathbb{C}$ (with free monad $\Sigma^\star$) modelling the constructors of the language, and a *behaviour functor* $B \colon \mathbb{C} \to \mathbb{C}$ modelling the type of transitions that programs can make during their execution. For example, to model a simple process algebra, one takes Σ to be a polynomial set functor representing the process constructors, like nil or $- \parallel -$, and B to be the functor $BX = (\mathcal{P}_\omega X)^L$ representing the behaviour of image-finite labelled transition systems. Every GSOS law has a canonical *operational model*, which equips the initial algebra $\Sigma(\mu\Sigma) \xrightarrow{\cong} \mu\Sigma$ of program terms with the structure of a *coalgebra* $\gamma \colon \mu\Sigma \to B(\mu\Sigma)$. Intuitively, γ is the transition system that runs programs according to the operational rules encoded by the law. Dually, there is a canonical *denotational model* obtained by equipping the final coalgebra $\nu B \xrightarrow{\cong} B(\nu B)$ (the domain of abstract program behaviours) with the structure of an *algebra* $\alpha \colon \Sigma(\nu B) \to \nu B$. The two models form the *initial* and the *final* bialgebra for the given GSOS law, respectively, and the unique bialgebra morphism $[\![-]\!] \colon \mu\Sigma \to \nu B$ gives a denotational semantics of programs; see the first diagram in Figure 1. The main feature of this framework is that the denotational semantics is always *compositional*: the denotation map $[\![-]\!]$ is, by construction, a morphism of Σ-algebras and thus denotational equivalence is respected by all constructors of the modelled language. Moreover, adequacy of the denotational semantics comes for free because denotational and behavioural equivalence simply coincide.

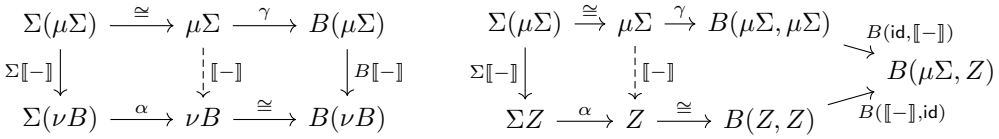

Figure 1: Denotational semantics in first-order and higher-order abstract GSOS.

Higher-Order Abstract GSOS A key limitation of Turi and Plotkin's framework is that it is not expressive enough to capture *higher-order* languages such as the λ-calculus. Recently, Goncharov et al. [Gon+23] have removed this limitation by developing the *higher-order abstract GSOS* framework. The idea is simple: move from behaviour *endo*functors to *bi*functors $B\colon \mathbb{C}^{\mathsf{op}} \times \mathbb{C} \to \mathbb{C}$ of mixed variance, accounting for the fact that in higher-order languages, programs can occur both as inputs (contravariantly) and as outputs (covariantly) of programs. For example, an untyped λ-calculus can be modelled by a behaviour bifunctor like $B(X, Y) = Y + Y^X$, which reflects that a λ-term either β-reduces to a λ-term, or acts as a function mapping λ-terms to λ-terms. The operational rules of a higher-order language are presented as a *higher-order GSOS law*, a (di)natural transformation

$$\varrho_{X,Y}\colon \Sigma(X \times B(X, Y)) \to B(X, \Sigma^\star(X + Y)).$$

As in the first-order case, every higher-order GSOS law has an *operational model*, which extends the initial algebra $\mu\Sigma$ of programs (e.g. λ-terms) to the initial *higher-order bialgebra*; see the top of the second diagram in Figure 1. This bialgebraic framework captures the operational semantics of a wide range of higher-order languages, and operational reasoning methods such as Howe's method [Urb+23] and logical relations [Gon+24] have been developed at this generality. In contrast, a bialgebraic treatment of higher-order *denotational* semantics along the lines of Turi and Plotkin's original vision has been missing so far.

Bialgebraic Denotational Semantics via Locally Final Coalgebras In recent work [Gon+26], we have introduced denotational semantics to the world of higher-order abstract GSOS. The main challenge is to identify the proper denotational models. Following Turi and Plotkin's ideas, the obvious approach is to use *final* higher-order coalgebras (and derived bialgebras). However, this does not work: the mixed-variance bifunctors modelling higher-order languages usually fail to have a final higher-order coalgebra, even for the simplest kinds of deterministic languages. Unlike in the first-order setting, non-existence of final higher-order coalgebras is not merely due to size issues, which could be circumvented by suitably restricting the behaviour bifunctors, but inherent to the nature of higher-order coalgebras.

The core insight of our work is to take *locally final coalgebras* instead of final ones. A higher-order coalgebra $Z \xrightarrow{\cong} B(Z, Z)$ for a bifunctor B is locally final if it is final as a coalgebra for the *endo*functor $B(Z, -)\colon \mathbb{C} \to \mathbb{C}$. We think of a locally final coalgebra as a domain of abstract higher-order behaviours. Unlike final coalgebras, locally final coalgebras exist for many higher-order languages, and are often (but not always) unique up to isomorphism. Thus, they form a natural foundation for bialgebraic denotational models. This is substantiated by several results:

1. We develop a general theory of existence, construction, and uniqueness of locally final coalgebras for bifunctors. Our approach is based on the ultrametric-enriched framework of *M-categories* [BST10], which provides a natural setting for the construction of fixed points of functors. We give a sufficient criterion for a behaviour bifunctor $B\colon \mathbb{C}^{\mathsf{op}} \times \mathbb{C} \to \mathbb{C}$

on an M-category to admit a (unique) locally final coalgebra, and provide an iterative construction method for it.

2. We show that every locally final coalgebra Z canonically extends to a bialgebra for a given higher-order GSOS law, provided that the latter satisfies a syntactic restriction called *relative flatness*. We thus obtain a unique denotation morphism from the initial higher-order bialgebra $\mu\Sigma$ to the bialgebra Z (second diagram in Figure 1). This denotational semantics is adequate: denotational implies behavioural equivalence w.r.t. the final coalgebra for $B(\mu\Sigma, -)$. Unlike in the first-order case, where adequacy holds by definition, this is a non-trivial result, as denotational and behavioural equivalence no longer coincide.

These results induce a generic denotational semantics for languages modelled in higher-order abstract GSOS. For instance, the untyped λ-calculus is captured by a higher-order GSOS law in $\mathbf{Set}^{\omega^{\mathrm{op}}}$ (the *topos of trees*), and its generic denotational model is given by a form of *guarded interaction trees* [FTB24], while behavioural equivalence is strong *applicative bisimilarity* [Abr90].

References

[Abr90] Samson Abramsky. "The lazy lambda calculus". In: *Research Topics in Functional Programming*. USA: Addison-Wesley Longman Publishing Co., Inc., 1990, 65–116. ISBN: 0201172364.

[BST10] Lars Birkedal, Kristian Støvring, and Jacob Thamsborg. "The category-theoretic solution of recursive metric-space equations". en. In: *Theoretical Computer Science* 411 (47 Oct. 2010), pp. 4102–4122. DOI: 10.1016/j.tcs.2010.07.010. URL: https://doi.org/10.1016/j.tcs.2010.07.010.

[FTB24] Dan Frumin, Amin Timany, and Lars Birkedal. "Modular Denotational Semantics for Effects with Guarded Interaction Trees". en. In: *Proceedings of the ACM on Programming Languages* 8 (POPL Jan. 2024), pp. 332–361. DOI: 10.1145/3632854.

[Gon+23] Sergey Goncharov, Stefan Milius, Lutz Schröder, Stelios Tsampas, and Henning Urbat. "Towards a Higher-Order Mathematical Operational Semantics". In: *Proc. ACM Prog. Lang.* 7, Issue POPL (2023), pp. 632–658. DOI: 10.1145/3571215.

[Gon+24] Sergey Goncharov, 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)*. IEEE Computer Society Press, 2024, 39:1–39:15. DOI: 10.1145/3661814.3662099.

[Gon+26] Sergey Goncharov, Marco Peressotti, Stelios Tsampas, Henning Urbat, and Stefano Volpe. *Towards a Higher-Order Bialgebraic Denotational Semantics*. 2026. arXiv: 2602.18295 [cs.PL].

[TP97] Daniele Turi and Gordon Plotkin. "Towards a mathematical operational semantics". In: *Twelfth Annual IEEE Symposium on Logic in Computer Science* (Warsaw, Poland). Vol. 1. IEEE Comput. Soc, 1997, pp. 280–291. DOI: 10.1109/lics.1997.614955. URL: https://doi.org/10.1109/lics.1997.614955.

[Urb+23] Henning Urbat, Stelios Tsampas, Sergey Goncharov, Stefan Milius, and Lutz Schröder. "Weak Similarity in Higher-Order Mathematical Operational Semantics". In: *38th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS 2023)*. IEEE Computer Society Press, 2023. DOI: 10.1109/LICS56636.2023.10175706.