From Abstract Higher-Order GSOS to Abstract Big-Step Semantics, Abstractly (Early Idea)

Sergey Goncharov
University of Birmingham, UK

Pouya Partow
University of Birmingham, UK

Stelios Tsampas
University of Southern Denmark, Denmark

Abstract

We propose an abstract (categorical) framework for relating small-step and big-step operational semantics (SOS). While small-step semantics defines a fine-grained reduction relation on programs, big-step semantics associates programs directly to their final results (values). Although the equivalence between these two semantics is critical for program analysis and reasoning, it is typically done on a case-by-case basis, lacking a unifying mathematical foundation. To address this, we leverage *higher-order mathematical operational semantics*, a recently introduced generalization of its first-order counterpart by Turi and Plotkin. We introduce a categorical abstraction of big-step SOS, complementing the existing abstract higher-order GSOS for small-step semantics. We then provide a general construction for deriving big-step semantics from small-step semantics to further obtain an abstract equivalence between them.

Keywords and phrases Operational semantics, Higher-order GSOS, Extended combinatory logic

Extended Combinatory Logic. For a quick illustration, consider the extended combinatory logic **xCL** [3] (cf. [2]), whose grammar, small-step and big-step operational semantics are given in Figure 1. **xCL** differs from the standard combinatory logic in three respects: (i) it involves labeled transitions, so that $s \xrightarrow{t} s'$ mimics the standard $st \to s'$, (ii) it features multi-ary "combinators" K', S', S'' for partial applications of K and S, and (iii) the semantics is properly operational call-by-name (lazy) semantics (akin to the lazy λ-calculus [1]).

Let $t{\downarrow}$ denote that $t \to t'$ for no t', equivalently: $t \xrightarrow{s} t'$ for some s and t'. It is easy to see that $t{\downarrow}$ iff t is a value. For **xCL**, as in many other settings, the equivalence

$$t \Downarrow s \iff t \to^\star s \wedge s{\downarrow} \qquad (\star)$$

is highly desirable, where $\to^\star$ is the transitive-reflexive closure of $\to$. Our goal is to abstract it and prove it on an abstract categorical level.

Abstract Higher-Order GSOS. Extending the original work of Turi and Plotkin [5], a (abstract) *higher-order GSOS law* in category $\mathbf{C}$ consists of a *signature functor* $\Sigma\colon \mathbf{C} \to \mathbf{C}$, a *behavior functor* $B\colon \mathbf{C}^{\mathsf{op}} \times \mathbf{C} \to \mathbf{C}$, and a family of maps

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

natural in Y and *dinatural* in X. Higher-order GSOS laws abstract a wide range of small-step operational semantic specifications, such as that for **xCL** (Σ and B for **xCL** are detailed further below), and induce *operational models* $\gamma\colon \mu\Sigma \to B(\mu\Sigma, \mu\Sigma)$, which are coalgebras abstracting derivable small-step transitions.

Separability. To abstractly formulate ($\star$), we refine ρ to what we call a *separated higher-order GSOS*. This is defined over a *signature of values* $\Sigma_{\mathsf{v}}\colon \mathbf{C} \to \mathbf{C}$, a *signature of computations* $\Sigma_{\mathsf{c}}\colon \mathbf{C} \times \mathbf{C} \to \mathbf{C}$, a *value behavior functor* $D\colon \mathbf{C}^{\mathsf{op}} \times \mathbf{C} \to \mathbf{C}$, an ω-continuous strong monad $\mathbf{T}$ [4] (e.g. $\mathbf{T}$ – powerset), and consists of the following ingredients: families of morphisms

$$\rho^{\mathsf{v}}_X\colon \Sigma_{\mathsf{v}}X \to D(X, \Sigma^\star X), \qquad \rho^{\mathsf{c}}_{X,Y}\colon \Sigma_{\mathsf{c}}(X \times (TD(X,Y) + TY), X) \to T\Sigma^\star(X+Y),$$

$$\textit{Values:} \quad v, w ::= I \mid K \mid S \mid K'(t) \mid S'(t) \mid S''(s,t)$$
$$\textit{Terms:} \quad s, t, r, q ::= v \mid st$$

$$S \xrightarrow{t} S'(t) \qquad S'(t) \xrightarrow{s} S''(t,s) \qquad S''(t,s) \xrightarrow{r} (tr)(sr) \qquad \frac{t \to t'}{ts \to t's} \qquad \frac{t \xrightarrow{s} t'}{ts \to t'}$$
$$K \xrightarrow{t} K'(t) \qquad K'(t) \xrightarrow{s} t \qquad I \xrightarrow{t} t$$

$$\frac{}{v \Downarrow v} \qquad \frac{s \Downarrow I \quad t \Downarrow v}{st \Downarrow v} \qquad \frac{s \Downarrow K \quad K'(t) \Downarrow v}{st \Downarrow v} \qquad \frac{s \Downarrow S \quad S'(t) \Downarrow v}{st \Downarrow v}$$
$$\frac{s \Downarrow K'(r) \quad r \Downarrow v}{st \Downarrow v} \qquad \frac{s \Downarrow S'(r) \quad S''(r,t) \Downarrow v}{st \Downarrow v} \qquad \frac{s \Downarrow S''(r,q) \quad (rt)(qt) \Downarrow v}{st \Downarrow v}$$

Figure 1 xCL: grammar, small-step operational semantics, big-step operational semantics.

dinatural in X and natural in Y, and a distributive law $\chi_{X,Y} \colon \Sigma_{\mathsf{c}}(TX, Y) \to T\Sigma_{\mathsf{c}}(X, Y)$ between $\mathbf{T}$ and $\Sigma_{\mathsf{c}}(-, Y)$ natural in Y. This yields $\Sigma X = \Sigma_{\mathsf{v}} X + \Sigma_{\mathsf{v}}(X, X)$, $B(X, Y) = T(D(X, Y) + Y)$ and a separation of the operational model into a value part $\gamma^{\mathsf{v}} \colon \Sigma_{\mathsf{v}}\mu\Sigma \to D(\mu\Sigma, \mu\Sigma)$ and a computation part $\gamma^{\mathsf{c}} \colon \Sigma_{\mathsf{c}}(\mu\Sigma, \mu\Sigma) \to T\mu\Sigma$. We then define (thanks to ω-continuity of $\mathbf{T}$) an abstraction of the multi-step semantics $\to^\star$ as the least fixpoint $\beta \colon \mu\Sigma \to T(\Sigma_{\mathsf{v}}\mu\Sigma)$ of the equation $\beta = [\eta, \beta^\sharp \cdot \gamma^{\mathsf{c}}] \cdot \iota^{-1}$ where ι is the isomorphism $\mu\Sigma \cong \Sigma_{\mathsf{v}}\mu\Sigma + \Sigma_{\mathsf{c}}(\mu\Sigma, \mu\Sigma)$ and $(-)^\sharp$ is the Kleisli lifting. Note that the purpose of making Σ_{c} a binary functor is to divide arguments into *strict* and *lazy*. This division is vital for the correctness of big-step semantics, since only the behavior of strict arguments can be inspected, e.g. $KI\Omega \Downarrow I$ only because we are not trying to evaluate $\Omega = (SII)(SII)$ (treating it lazily).

For **xCL**, Σ_{v}, Σ_{c} and D instantiate as follows over $\mathbf{C} = \mathbf{Set}$:

$$\Sigma_{\mathsf{v}} X = \underbrace{1 + 1 + 1}_{I,K,S} + \underbrace{X + X}_{K',S'} + \underbrace{X \times X}_{S''} \qquad \Sigma_{\mathsf{c}}(X, Y) = \underbrace{X \times Y}_{\text{application}} \qquad D(X, Y) = \underbrace{Y^X}_{\xrightarrow{-}-}$$

Abstract Big-Step Semantics and Equivalence. Under the above assumptions we abstract big-step semantics as natural transformations $\xi \colon \Sigma_{\mathsf{c}}(\Sigma_{\mathsf{v}} X, X) \to T(\Sigma^\star X)$, representing rules other than $\frac{}{v \Downarrow v}$ (those are determinied solely by Σ_{v}). A suitable notion of operational model can be defined in the big-step case. It turns out though that separability is not enough for proving ($\star$). Consider the following small-step specification:

$$g(x) \xrightarrow{y} f(y) \qquad \Omega \to \Omega \qquad \frac{x \to y}{f(x) \to g(y)} \qquad \frac{x \xrightarrow{x} y}{f(x) \to x}$$

The only way to define big-step rules would be as follows:

$$\frac{}{g(x) \Downarrow g(x)} \qquad \frac{x \Downarrow g(y) \qquad g(y) \Downarrow v}{f(x) \Downarrow v}$$

The equivalence ($\star$) then fails, because $f(f(g(\Omega))) \to g(g(\Omega))$, but $f(f(g(\Omega))) \Downarrow g(\Omega)$. Our idea to remedy this is to impose the *strong separation condition*, abstracting the following: if a rule has at least one premise of the form $x_k \to x'_k$ then the conclusion of the rule must be

$$f(x_1, \ldots, x_n, y_1, \ldots, y_m) \to f(x'_1, \ldots, x'_n, y_1, \ldots, y_m)$$

where either $x_i \to x'_i$ occurs in the premise, or $x'_i = x_i$.

References

1 S. Abramsky. The lazy λ-calculus. In *Research topics in Functional Programming*, pages 65–117. Addison Wesley, 1990.

2 Pietro Di Gianantonio, Furio Honsell, and Marina Lenisa. Rpo, second-order contexts, and λ-calculus. In Roberto Amadio, editor, *Foundations of Software Science and Computational Structures*, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

3 Sergey Goncharov, Stefan Milius, Lutz Schröder, Stelios Tsampas, and Henning Urbat. Towards a higher-order mathematical operational semantics. 7:632–658, 2023.

4 Sergey Goncharov, Lutz Schröder, Christoph Rauch, and Julian Jakob. Unguarded recursion on coinductive resumptions. *Log. Methods Comput. Sci.*, 14(3), 2018.

5 Daniele Turi and Gordon D. Plotkin. Towards a mathematical operational semantics. In *12th Annual IEEE Symposium on Logic in Computer Science (LICS 1997)*, pages 280–291, 1997.