

A categorical framework for congruence of applicative bisimilarity in higher-order languages

Tom Hirschowitz¹ Ambroise Lafont²

¹Univ. Grenoble Alpes, Univ. Savoie Mont Blanc, CNRS, LAMA
Chambéry, France

²UNSW, Sydney

Journée LHC
February 2021

Based on previous work with Peio Borthelle.

Motivation: generalisation of theorem statements

- Often, theorems are stated for **one** “typical” programming language.
- Goal: provide high-level tools for stating them **for all suitable languages and models**.

State of the art

- **Formats:** Tyft/tyxt, GSOS, PATH,...
- **Bialgebraic semantics** (Turi and Plotkin '97).
Functional languages only starting to be investigated (Peressotti '17).
- **Reduction monads** (Ahrens et al. '20).

Main contribution

1. Abstract setting for specifying operational semantics from **signatures**.
2. Abstract analogue of Abramsky's **applicative** bisimilarity in cbn λ -calculus, called **substitution-closed bisimilarity**.
3. A **semantic format** for congruence of substitution-closed bisimilarity:

Main theorem

If the signature complies with the format, then substitution-closed bisimilarity is a congruence.

Proof: abstract analogue of Howe's method.

This talk

Sketch main ideas on **one example**, big-step, cbn λ -calculus.

Call-by-name λ -calculus

Slightly non-standard presentation.

$$\frac{}{\lambda x.e \Downarrow e} \qquad \frac{e_1 \Downarrow e_3 \quad e_3[e_2] \Downarrow e_4}{e_1 e_2 \Downarrow e_4}$$

Typing: $\Downarrow \subseteq$ closed terms \times terms with 1 free variable.

Syntactic graphs

Objects of interest:

- graphs with typed vertices,
- types = natural numbers, morally numbers of potential free variables,
- sources are closed,
- targets have one potential free variable,
- vertices support the operations of untyped λ -calculus, including substitution.

Syntactic graphs

Definition

A **syntactic graph** consists of

- a model X_0 of syntax ($X_0(n)$ means n potential free variables), including

$$\lambda_n : X_0(n+1) \rightarrow X_0(n) \qquad \text{app}_n : X_0(n)^2 \rightarrow X_0(n)$$

$$\text{subst}_{p,n} : X_0(p) \times X_0(n)^p \rightarrow X_0(n),$$

- a set X_{\Downarrow} of edges, and
- a source and target map $X_{\Downarrow} \rightarrow X_0(0) \times X_0(1)$.

They form a category Σ_0 -**Gph**.

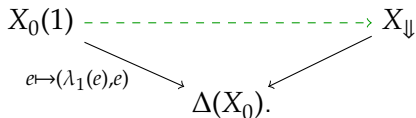
Notation: $X_{\Downarrow} \rightarrow \Delta(X_0)$

Transition rules

A syntactic graph is a model of the rule

$$\overline{\lambda x.e \Downarrow e}$$

when it is equipped with



Intuition

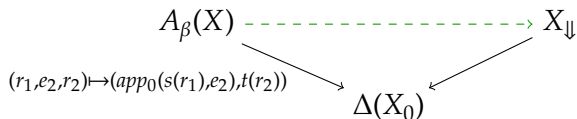
For all potential parameters, there is a transition with expected source and target.

Transition rules

A syntactic graph is a model of the rule

$$\frac{\frac{r_1}{e_1 \Downarrow e_3} \quad \frac{r_2}{e_3[e_2] \Downarrow e_4}}{e_1 e_2 \Downarrow e_4} .$$

when it is equipped with



where $A_\beta(X) = \{(r_1, e_2, r_2) \mid t(r_1)[e_2] = s(r_2)\}$.

Intuition

For all potential parameters, there is a transition with expected source and target.

Models

Definition

A **model** is a syntactic graph that is a model of both rules.

Informal Proposition

The initial model is a proof-relevant variant of the standard, syntactic graph.

Main result

Sketch: one defines

- substitution-closed relations,
- bisimulation,
- substitution-closed bisimilarity.

Theorem

Substitution-closed bisimilarity is a congruence.

Focus today: what makes the general theorem applicable to $\text{cbn } \lambda$.

Representable arities

Main steps:

- bisimulation by lifting and
- representable arities.

Bisimulation by lifting I

“Small” syntactic graphs:

- $\mathcal{L}(\mathbf{y}_0)$:
 - vertices generated from one closed constant, say k_0 ,
 - no transition.
- $\mathcal{L}(\mathbf{y}_\Downarrow)$:
 - vertices generated from

$$k'_0 \in \mathcal{L}(\mathbf{y}_\Downarrow)(0) \quad \text{and} \quad k'_1 \in \mathcal{L}(\mathbf{y}_\Downarrow)(1),$$

- one transition $r : k_0 \Downarrow k_1(x)$.

Proposition (\approx Yoneda)

- $\Sigma_0\text{-Gph}(\mathcal{L}(\mathbf{y}_0), X) \cong X_0(0)$.
- $\Sigma_0\text{-Gph}(\mathcal{L}(\mathbf{y}_\Downarrow), X) \cong X_\Downarrow$.
- $\mathcal{L}(\mathbf{y}_0) \xrightarrow{\mathcal{L}(\mathbf{y}_s)} \mathcal{L}(\mathbf{y}_\Downarrow) \xrightarrow{e} X$ corresponds to $s(e)$.

Bisimulation by lifting II

Definition

$f : X \rightarrow Y$ is a **functional bisimulation** iff

$$\begin{array}{ccc}
 \mathcal{L}(\mathbf{y}_0) & \xrightarrow{x} & X \\
 \mathbf{y}_s \downarrow & \nearrow e & \downarrow f \\
 \mathcal{L}(\mathbf{y}_{\downarrow}) & \xrightarrow{e'} & Y
 \end{array}
 \quad
 \begin{array}{ccc}
 x & \longmapsto & f(x) \\
 e \Downarrow & & \Downarrow e' \\
 x' & \longmapsto & y'
 \end{array}
 \quad
 \text{(for all / exists)}$$

Notation

$f \in \{\mathcal{L}(\mathbf{y}_s)\}^{\square}$, generalises to \mathbb{J}^{\square} .
 $\mathcal{L}(\mathbf{y}_s) \in^{\square} \{f\}$, generalises to ${}^{\square}\mathbb{K}$.

Definition

- **fibration**: $\{\mathcal{L}(\mathbf{y}_s)\}^{\square}$.
- **cofibration**: ${}^{\square}(\{\mathcal{L}(\mathbf{y}_s)\}^{\square})$.

Representable operation arities

By example: $e_1 e_2$, **head** operation of $\frac{e_1 \Downarrow e_3 \quad e_3[e_2] \Downarrow e_4}{e_1 e_2 \Downarrow e_4}$.

Goal

Find E_{app} such that $X(0)^2 \cong \Sigma_0\text{-Gph}(E_{app}, X)$, naturally in X .

Solution (merely saying that application has 2 arguments)

$$E_{app} = \mathcal{L}(\mathbf{y}_0) + \mathcal{L}(\mathbf{y}_0).$$

By $X(0)^2 \cong \Sigma_0\text{-Gph}(\mathcal{L}(\mathbf{y}_0), X)^2$
 $\cong \Sigma_0\text{-Gph}(\mathcal{L}(\mathbf{y}_0) + \mathcal{L}(\mathbf{y}_0), X)$.

Remark: \mathcal{L} preserves coproducts, so $\mathcal{L}(\mathbf{y}_0) + \mathcal{L}(\mathbf{y}_0) \cong \mathcal{L}(\mathbf{y}_0 + \mathbf{y}_0)$, etc.

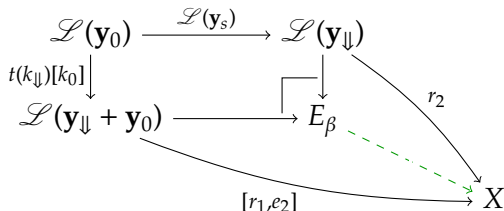
Representable rule arities

By example:

$$\frac{\frac{r_1}{e_1 \Downarrow e_3} \quad \frac{r_2}{e_3[e_2] \Downarrow e_4}}{e_1 e_2 \Downarrow e_4} .$$

Goal

Find E_β such that $A_\beta(X) \cong \Sigma_0\text{-Gph}(E_\beta, X)$, naturally in X .



Indeed, $A_\beta(X) \cong$ cocones to X
 $\cong \Sigma_0\text{-Gph}(E_\beta, X)$.

Semantic format

Proposition

Each rule yields a boundary morphism

head operation arity \rightarrow rule arity.

$$\begin{array}{ccc}
 \mathcal{L}(\mathbf{y}_0) & \longrightarrow & \mathcal{L}(\mathbf{y}_\Downarrow) \\
 \downarrow & & \lrcorner \downarrow \\
 \mathcal{L}(\mathbf{y}_0 + \mathbf{y}_0) & \longrightarrow & \mathcal{L}(\mathbf{y}_\Downarrow + \mathbf{y}_0) \longrightarrow E_\beta
 \end{array}$$

Condition

Each (head operation arity \rightarrow rule arity) is a cofibration.

For $\frac{e_1 \Downarrow e_3 \quad e_3[e_2] \Downarrow e_4}{e_1 e_2 \Downarrow e_4}$: stability under pushouts and composition.

Conclusion

Semantic format for congruence of substitution-closed bisimilarity

Representable arities should form cofibrations.

- Shown here: example of $\text{cbn } \lambda$.
- In the paper: general framework + more examples.

Short-term future work

Languages with terms as Labels.