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

Tom Hirschowitz\textsuperscript{1} \hspace{1cm} Ambroise Lafont\textsuperscript{2}

\textsuperscript{1}Univ. Grenoble Alpes, Univ. Savoie Mont Blanc, CNRS, LAMA Chambéry, France

\textsuperscript{2}UNSW, 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).
Main contribution

1. Abstract setting for specifying operational semantics from signatures.
2. Abstract analogue of Abramsky’s applicative bisimilarity in cbn \( \lambda \)-calculus, called substitution-closed bisimilarity.
3. A semantic format for congruence of substitution-closed bisimilarity:

<table>
<thead>
<tr>
<th>Main theorem</th>
</tr>
</thead>
<tbody>
<tr>
<td>If the signature complies with the format, then substitution-closed bisimilarity is a congruence.</td>
</tr>
</tbody>
</table>

Proof: abstract analogue of Howe’s method.
This talk

Sketch main ideas on one example, big-step, cbn $\lambda$-calculus.
Call-by-name $\lambda$-calculus

Slightly non-standard presentation.

$\lambda x. e \Downarrow e$

$e_1 \Downarrow e_3$
$e_3[e_2] \Downarrow e_4$
$e_1 \Downarrow e_4$
$e_2 \Downarrow e_4$

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 $\lambda$-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

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

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

- a set $X_{\downarrow}$ of edges, and

- a source and target map $X_{\downarrow} \to X_0(0) \times X_0(1)$.

They form a category $\Sigma_0 \text{-Gph}$. 

**Notation:** $X_{\downarrow} \to \Delta(X_0)$
Transition rules

A syntactic graph is a model of the rule

\[ \lambda x. e \Downarrow e \]

when it is equipped with

\[ X_0(1) \rightarrow \Delta(X_0) \rightarrow X_\downarrow \]

\[ e \mapsto (\lambda_1(e), e) \]

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

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

when it is equipped with

\[
A_\beta(X) \xrightarrow{\text{app}_0(s(r_1),e_2),t(r_2)} X_{\downarrow} \xrightarrow{\Delta(X_0)} \Delta(X_0)
\]

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 cbn $\lambda$. 
Representable arities

Main steps:

- bisimulation by lifting and
- representable arities.
Bisimulation by lifting 1

“Small” syntactic graphs:

- $\mathcal{L}(y_0)$:
  - vertices generated from one closed constant, say $k_0$,
  - no transition.

- $\mathcal{L}(y_\downarrow)$:
  - vertices generated from
    
    $k'_0 \in \mathcal{L}(y_\downarrow)(0)$ and $k'_1 \in \mathcal{L}(y_\downarrow)(1),$
  - one transition $r : k_0 \downarrow k_1(x)$.

Proposition ($\approx$ Yoneda)

<table>
<thead>
<tr>
<th>Proposition ($\approx$ Yoneda)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Sigma_0$-$\text{Gph}(\mathcal{L}(y_0), X) \cong X_0(0)$.</td>
</tr>
<tr>
<td>$\Sigma_0$-$\text{Gph}(\mathcal{L}(y_\downarrow), X) \cong X_\downarrow$.</td>
</tr>
<tr>
<td>$\mathcal{L}(y_0) \xrightarrow{\mathcal{L}(y_s)} \mathcal{L}(y_\downarrow) \xrightarrow{e} X$ corresponds to $s(e)$.</td>
</tr>
</tbody>
</table>
**Bisimulation by lifting II**

### Definition

\[ f : X \to Y \text{ is a functional bisimulation iff} \]

\[
\begin{align*}
\mathcal{L}(y_0) & \xrightarrow{x} X & x & \xrightarrow{} f(x) \\
y_s & \xleftarrow{e} f & e & \xleftarrow{} e' \\
\mathcal{L}(y_s) & \xrightarrow{e'} Y & x' & \xrightarrow{} y'.
\end{align*}
\]

(for all / exists)

### Notation

\[ f \in \{\mathcal{L}(y_s)\}^\square, \text{ generalises to } J^\square. \]
\[ \mathcal{L}(y_s) \in \square\{f\}, \text{ generalises to } \square K. \]

### Definition

- **fibration**: \( \{\mathcal{L}(y_s)\}^\square \).
- **cofibration**: \( \square(\{\mathcal{L}(y_s)\}^\square) \).
Representable operation arities

By example: $e_1 e_2$, head operation of

\[
\begin{array}{c}
e_1 \Downarrow e_3 \quad e_3[e_2] \Downarrow e_4 \\
\hline
\end{array}
\]

\[
e_1 e_2 \Downarrow e_4
\]

Goal

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

Solution (merely saying that application has 2 arguments)

\[
E_{app} = \mathcal{L}(y_0) + \mathcal{L}(y_0).
\]

By

\[
X(0)^2 \cong \Sigma_0 \dashv \text{Gph}(\mathcal{L}(y_0), X)^2 \\
\cong \Sigma_0 \dashv \text{Gph}(\mathcal{L}(y_0) + \mathcal{L}(y_0), X).
\]

Remark: $\mathcal{L}$ preserves coproducts, so $\mathcal{L}(y_0) + \mathcal{L}(y_0) \cong \mathcal{L}(y_0 + y_0)$, etc.
Representable rule arities

By example:

\[
\begin{array}{ccc}
  r_1 & & r_2 \\
  e_1 \downarrow e_3 & & e_3[e_2] \downarrow e_4 \\
  e_1 e_2 \downarrow e_4
\end{array}
\]

Goal

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

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

<table>
<thead>
<tr>
<th>Proposition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Each rule yields a boundary morphism</td>
</tr>
<tr>
<td>head operation arity $\to$ rule arity.</td>
</tr>
</tbody>
</table>

$$
\mathcal{L}(y_0) \to \mathcal{L}(y_\downarrow)
$$

$$
\mathcal{L}(y_0 + y_0) \to \mathcal{L}(y_\downarrow + y_0) \to E_\beta
$$

<table>
<thead>
<tr>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Each (head operation arity $\to$ rule arity) is a cofibration.</td>
</tr>
</tbody>
</table>

For $e_1 \downarrow e_3 \quad e_3[e_2] \downarrow e_4$ : stability under pushouts and composition.
Conclusion

Semantic format for congruence of substitution-closed bisimilarity

Representable arities should form cofibrations.

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

Short-term future work

Languages with terms as Labels.