Familial monads and structural operational semantics

Tom Hirschowitz

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

CHoCoLa Seminar
December 13, 2018

Structural operational semantics

- Notes by Plotkin (1981): method rather than theory, by example.
- Describe dynamics of programming languages, syntactically.
  - Terms from algebraic signature.
  - Dynamics as a (labelled) transition system.
  - Basic idea: describe behaviour of each operation.

$$\frac{\dots \qquad x_i \xrightarrow{a_i} y_i \qquad \dots}{f(x_1, \dots, x_n) \xrightarrow{a} M(y_1, \dots, y_n)}$$

Structural: behaviour of system determined by its components.

- Disturbing observation: bisimilarity in π not a congruence!

Structural $\not\Longrightarrow$ compositional.

Formats

De Simone (1985): rule format.

- Algebraic signature + transition system specification

  $\rightsquigarrow$ transition system.
- Specification complies with format $\Longrightarrow$ transition system behaves well.
- E.g.,
  - (weak) bisimilarity is a congruence,
  - conservative extension,
  - bisimulation up to X is sound.

A wealth of formats

Since then, lots of different formats, combining:

- negative premises,
- predicates,
- look ahead,
- terms as labels,
- variable binding,...

Functorial operational semantics

- Attempt to tame the diversity of formats and define models.
- Appealing simplicity:
  - terms = monad T,
  - labels = comonad L,
  - rules = distributive law $TL \to LT$.
- But not so widely adopted.
- Possible reasons:
  - too abstract,
  - not expressive enough (e.g., no negative premises or higher-order afaik),
  - does not scale well to variable binding.
- Simplifying attempt by Staton (2008):
  - SOS = monad on labeled relations.
  - Better treatment of variable binding.
  - But no clear improvement on expressiveness, and not better adopted.

Proposal

Two distinct goals:

1. Find the right language for describing
   - what goes on in proofs of congruence of bisimilarity, etc,...,
   - under which hypotheses.
2. Generate instances satisfying the hypotheses.

Here: focus on 1.

Plan

New abstract framework:

Main example	General case
cat. of transition systems functional bisimulation	transition category by lifting (cf. presheaf models)
SOS specification model partial transition proof	monad $\mathscr{T}$ $\mathscr{T}$-algebra Kleisli morphism

- Morally, monad = saturation by the given rules.
- Congruence proof $\leftarrow\!\!\!\sim$ familiality of $\mathscr{T}$,...
- Example with binding: π-calculus.

My first transition category

Categories that look like transition systems and functional simulations.

Baby example

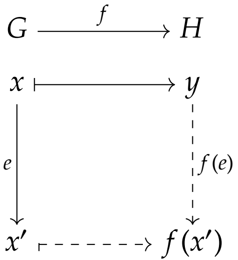
(Directed, multi-)graphs, **Gph**.

- Intuitively: untyped + one label ≈ rewrite systems.
- Presheaves over Γ, the category consisting of $s, t \colon [0] \rightrightarrows [1]$.
- Presheaf = contravariant functor to **Set** (Notation: $\widehat{\Gamma}$!):

$$\begin{array}{rcl} G \colon \Gamma^{op} & \to & \mathbf{Set} \\ [0] & \mapsto & V_G \\ [1] & \mapsto & E_G \\ (s \colon [0] \to [1]) & \mapsto & (V_G \xleftarrow{\sigma} E_G) \\ (t \colon [0] \to [1]) & \mapsto & (V_G \xleftarrow{\tau} E_G). \end{array}$$

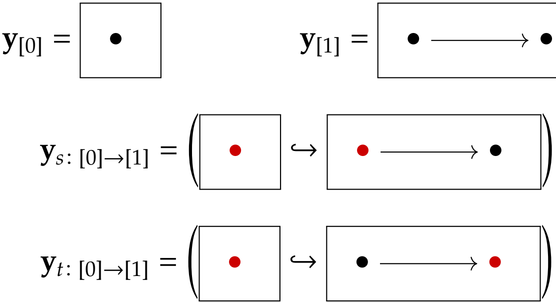
Graph morphisms

Graph morphism = natural transformation = functional simulation.



Yoneda embedding for graphs

Yoneda embedding $\mathbf{y}\colon \Gamma \hookrightarrow \mathbf{Gph}$:



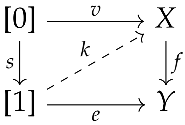
Notation

Yoneda embedding often kept implicit!

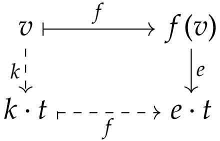
Functional bisimulation by lifting

Definition (Functional bisimulation)

f functional bisimulation iff $\forall v, e, \exists k$:



i.e.



- v picks a vertex in X.
- e picks an edge in Y.
- Commutation says $s(e) = f(v)$.
- Existence of k:
    - antecedent of e by f,
    - whose source is v.

Interlude: lifting

Definition

- In any category, $s \boxslash f$ iff $\forall v, e, \exists k$:

$$\begin{array}{ccc} A & \xrightarrow{v} & X \\ {\scriptstyle s}\downarrow & \overset{k}{\nearrow} & \downarrow{\scriptstyle f} \\ B & \xrightarrow[e]{} & Y. \end{array}$$

- Similarly:

$$S^{\boxslash} = \{f \mid \forall s \in S, s \boxslash f\} \qquad {}^{\boxslash}F = \{s \mid \forall f \in F, s \boxslash f\}.$$

Proposition (obvious)

f functional bisimulation $\Leftrightarrow s \boxslash f \Leftrightarrow f \in \{s\}^{\boxslash}$.

A transition category with basic labels

- Let A be the considered set of labels.
- Presheaves over Ω_A:

$$\ldots \qquad [a] \qquad \ldots \qquad (a \in A)$$
$$s^a \uparrow\uparrow t^a$$
$$[0]$$

- Any $X \in \widehat{\Omega_A}$ has
  - a set of vertices $X[0]$,
  - a set of a-transitions $X[a]$ for all $a \in A$, each with its source and target.

A transition category with basic labels

Definition (Functional bisimulation)

$$
\begin{array}{ccc}
[0] & \xrightarrow{v} & X \\
{\scriptstyle s^a}\downarrow & \overset{k}{\nearrow} & \downarrow{\scriptstyle f} \\
[a] & \xrightarrow[e]{} & Y
\end{array}
\qquad \text{i.e.} \qquad
\begin{array}{ccc}
v & \overset{f}{\longmapsto} & f(v) \\
{\scriptstyle k:a}\downarrow & & \downarrow{\scriptstyle e:a} \\
k \cdot t^a & \underset{f}{\longmapsto} & e \cdot t^a
\end{array}
$$

Equivalently, $f \in \{s^a \mid a \in A\}^{\boxslash}$.

- Transition types in $\mathbf{Gph}$: just $[1]$.
- Here: all $[a]$'s.

Transition categories

Definition

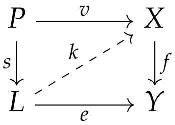
Category with distinguished cospans

$$P \xrightarrow{s} L \xleftarrow{t} Q$$

+ finite completeness, cocompleteness, well-poweredness, images, and tininess of all $P \in \mathbf{P}$.

Let $\mathbf{T}_s$ denote the set of all such $s \colon P \to L$.

Definition (Functional bisimulation)



i.e. $f \in \mathbf{T}_s^{\boxslash}$.

Interlude: quick introduction to monads

On **Set** for concreteness.

Definition

A monad is an endofunctor $\mathscr{T} : \mathbf{Set} \to \mathbf{Set}$ with natural transformations

$$X \xrightarrow{\eta_X} \mathscr{T}(X) \qquad \mathscr{T}(\mathscr{T}(X)) \xrightarrow{\mu_X} \mathscr{T}(X)$$

satisfying axioms.

Notation

$$\begin{array}{ccc} X & \xrightarrow{\eta_X} & \mathscr{T}(X) \\ x & \mapsto & (\!|x|\!). \end{array}$$

Intuition: $\mathscr{T}(X)$ = terms with free variables in X.

Interlude: quick introduction to monads

Example

CCS syntax:

$$\mathscr{T}_{CCS}(X) \ni P, Q ::= (\!|x|\!) \mid (P|Q) \mid a.P \mid \bar{a}.P \mid \ldots \qquad (x \in X).$$

Interlude: monads

Let $\langle n \rangle = \{1, \ldots, n\}$.

Intuition

- Morphism $e \colon \langle n \rangle \to \mathscr{T}(X)$ = n-tuple of terms with free variables in X.
- $e_i = e(i) \in \mathscr{T}(X)$.
- On morphisms, $\mathscr{T}(f) \colon \mathscr{T}(X) \to \mathscr{T}(Y)$ = renaming of free variables.

Kleisli composition ≈ substitution

Definition (Kleisli composition)

Given $f: X \to \mathscr{T}(Y)$ and $g: Y \to \mathscr{T}(Z)$:

$$X \xrightarrow{f} \mathscr{T}(Y) \xrightarrow{\mathscr{T}(g)} \mathscr{T}(\mathscr{T}(Z)) \xrightarrow{\mu_Z} \mathscr{T}(Z).$$

- $\mathscr{T}(\mathscr{T}(Z))$ terms whose free variables are terms.
- Intuition for μ: remove outer $(\!|-|\!)$'s.

Example

$$b.\left((\!|\ a.(\!|x|\!)|(\!|y|\!)\ |\ (\!|z|\!)\ |\!)\ |\ (\!|\ \overline{a}.(\!|t|\!)\ |\!)\right) \quad \stackrel{\mu}{\longmapsto} \quad b.\left(a.(\!|x|\!)|(\!|y|\!)\ |\ (\!|z|\!)\ |\ \overline{a}.(\!|t|\!)\right)$$

SOS specifications as monads

Idea (Staton): view SOS rules as endofunctors.

Concrete idea, on $\widehat{\Omega_A}$

SOS specification $S \rightsquigarrow$ monad $\mathscr{T}_S$:

- $\mathscr{T}_S(X)[0]$: terms with constants in $X[0]$,
- $\mathscr{T}_S(X)[a]$: derivations with transition axioms in all $X[b]$'s,
- multiplication $\mathscr{T}_S^2(X)[a] \to \mathscr{T}_S(X)[a]$: plugging derivations.

Example: CCS

- Take
$$A := 1 + \mathscr{N} + \mathscr{N} = \{\tau\} \uplus \{a \mid a \in \mathscr{N}\} \uplus \{\bar{a} \mid a \in \mathscr{N}\}$$
for some set of names $\mathscr{N}$.
- Let $\alpha ::= \tau \mid a \mid \bar{a}$.
- $X \in \widehat{\Omega_A}$ any A-labelled transition system.
- Think of:
  - $X[0]$ as a set of term variables.
  - $X[\alpha]$ as a set of (typed) transition variables.

Example: CCS

- $\mathscr{T}_{CCS}(X)[0]$: CCS terms with constants in $X[0]$.

$$P, Q ::= (\!|x|\!) \mid (P|Q) \mid a.P \mid \bar{a}.P \mid \ldots \qquad (x \in X[0])$$

- $\mathscr{T}_{CCS}(X)[\alpha]$: α-transitions with constants in all $X[\beta]$'s.

$$\frac{x \xrightarrow{e\colon \alpha} y \text{ in } X}{(\!|e|\!)\colon (\!|x|\!) \xrightarrow{\alpha} (\!|y|\!)} \qquad \frac{}{in^a_P\colon a.P \xrightarrow{a} P} \qquad \frac{L\colon P \xrightarrow{\alpha} P'}{L|Q\colon (P|Q) \xrightarrow{\alpha} (P'|Q)}$$

$$\frac{L\colon P \xrightarrow{\bar{a}} P' \qquad R\colon Q \xrightarrow{a} Q'}{(L \rhd R)\colon (P|Q) \xrightarrow{\tau} (P'|Q')} \qquad \ldots$$

Deeper into CCS

Next few slides:

- concrete examples of transitions,
- operations upon them, e.g., substitution,

with categorical interpretation.

CCS: transition proofs as Kleisli morphisms

Consider the labelled transition system $X := [\bar{a}] + [a]$:

$$x \xrightarrow{e:\bar{a}} x' \qquad\qquad y \xrightarrow{f:a} y'.$$

Then the proof (omitting $(\!|-|\!)$'s)

$$\frac{\dfrac{}{e\colon x \xrightarrow{\bar{a}} x'} \qquad \dfrac{}{f : y \xrightarrow{a} y'}}{(e \rhd f)\colon (x|y) \xrightarrow{\tau} (x'|y')}$$

is in $\mathscr{T}_{CCS}([\bar{a}] + [a])[\tau]$ hence yields a morphism $[\tau] \to \mathscr{T}([\bar{a}] + [a])$.

Remark

Linear use of variables.

CCS: transition proofs as Kleisli morphisms

Other example:

$$\dfrac{\dfrac{\dfrac{}{e\colon x \xrightarrow{\bar{a}} x'}}{(e|y)\colon (x|y) \xrightarrow{\bar{a}} (x'|y)}}{(e|y|y)\colon (x|y|y) \xrightarrow{\bar{a}} (x'|y|y)}$$

yields a morphism $[\bar{a}] \to \mathscr{T}([\bar{a}] + [a])$.

Remark

Non-linear use of variables.

CCS: substitution

Plugging the proof below left in e on the right

$$\frac{}{out^a_z \colon \bar{a}.z \xrightarrow{\bar{a}} z} \qquad \frac{\dfrac{}{e \colon x \xrightarrow{\bar{a}} x'} \qquad \dfrac{}{f \colon y \xrightarrow{a} y'}}{(e \rhd f) \colon (x|y) \xrightarrow{\tau} (x'|y')}$$

where $z \in X[0]$, yields

$$\frac{\dfrac{}{out^a_z \colon \bar{a}.z \xrightarrow{\bar{a}} z} \qquad \dfrac{}{f \colon y \xrightarrow{a} y'}}{(out^a_z \rhd f) \colon (\bar{a}.z|y) \xrightarrow{\tau} (z|y')} .$$

CCS: substitution, categorically?

The proofs

$$\frac{}{out^a_z \colon \bar{a}.z \xrightarrow{\bar{a}} z} \qquad \frac{\dfrac{}{e \colon x \xrightarrow{\bar{a}} x'} \qquad \dfrac{}{f \colon y \xrightarrow{a} y'}}{(e \rhd f) \colon (x|y) \xrightarrow{\tau} (x'|y')}$$

are morphisms

$$[\tau] \xrightarrow{e \rhd f} \mathscr{T}_{CCS}([\bar{a}] + [a]) \qquad [\bar{a}] \xrightarrow{out^a_z} \mathscr{T}_{CCS}[0].$$

How to model substitution?

Kleisli composition.

CCS: substitution, categorically?

The substitution from before

$$\frac{}{out^a_z \colon \bar{a}.z \xrightarrow{\bar{a}} z} \qquad \frac{\dfrac{}{e \colon x \xrightarrow{\bar{a}} x'} \qquad \dfrac{}{f \colon y \xrightarrow{a} y'}}{(e \rhd f) \colon (x|y) \xrightarrow{\tau} (x'|y')}$$

amounts to

$$\begin{array}{ccc} \mathscr{T}_{CCS}([\bar{a}] + [a]) & \xrightarrow{\mathscr{T}_{CCS}([\text{“}out^a_z\text{”}, \eta_{[a]}])} & \mathscr{T}_{CCS}(\mathscr{T}_{CCS}([0] + [a])) \\ \uparrow{\scriptstyle e \rhd f} & & \downarrow{\scriptstyle \mu} \\ [\tau] & & \mathscr{T}_{CCS}([0] + [a]) \end{array}$$

Proof substitution = Kleisli composition.

Bonus: models

Bonus

Models for a SOS specification = Algebras for the monad.

Definition

A $\mathscr{T}$-algebra is a map of the form $\mathscr{T}(X) \to X$ satisfying axioms.

Intuition: interpretation of the syntactic operations and SOS rules.

Summary

Basic set up

- Transition category $\mathscr{C}$ (≈ transition systems and simulations)
- Monad $\mathscr{T} : \mathscr{C} \to \mathscr{C}$ (≈ SOS specification = set of SOS rules)
- Morphisms in Kleisli category of $\mathscr{T}$: transition proofs.
- Kleisli composition: proof substitution.
- $\mathscr{T}$-algebras: models.

Congruence of bisimilarity

Will follow from:

Theorem

Up to suitable hypotheses, if $a\colon \mathscr{T}(X) \to X$ *is a* $\mathscr{T}$*-algebra and* $f : R \to X$ *is a functional bisimulation, then so is*

$$\mathscr{T}(R) \xrightarrow{\mathscr{T}(f)} \mathscr{T}(X) \xrightarrow{a} X.$$

Intuition

- If $R[0] = \{(x, y) \in X[0] \times X[0] \mid ...\}$,
- then $\mathscr{T}(R)[0]$ = terms with free variables in R, i.e.,

$$\{C[(x_1, y_1), ..., (x_n, y_n)] \mid \forall i, (x_i, y_i) \in R[0]\}.$$

- $\rightsquigarrow \mathscr{T}(R)$ is the context closure of R.

Hypothesis 1: compositionality

Generally a vague concept (thanks for asking!).

Definition

An algebra $a \colon \mathscr{T}(X) \to X$ is compositional iff it is a functional bisimulation.

$$\begin{array}{ccc} P & \xrightarrow{v} & \mathscr{T}(X) \\ {\scriptstyle s}\downarrow & \overset{k}{\nearrow} & \downarrow{\scriptstyle a} \\ L & \xrightarrow[e]{} & X \end{array}$$

Morally: any transition $C[x_1, \ldots, x_n] \xrightarrow{\alpha} x'$ decomposes as

$$\frac{\ldots \qquad x_i \xrightarrow{\alpha_i} y_i \qquad \ldots}{C[x_1, \ldots, x_n] \xrightarrow{\alpha} E[y_1, \ldots, y_n]}\ .$$

Congruence of bisimilarity

Will follow from:

Theorem

If X is a compositional $\mathscr{T}$-algebra and $f : R \to X$ is a functional bisimulation, then so is

$$\mathscr{T}(R) \xrightarrow{\mathscr{T}(f)} \mathscr{T}(X) \xrightarrow{a} X,$$

up to further hypotheses.

It now suffices to prove that $\mathscr{T}(f)$ is a functional bisimulation.

Standard proof method (simplified)

- f is just first projection.
- Consider any $C[(x_1, y_1), ..., (x_n, y_n)] \in \mathscr{T}(R)$.
- Given transition from $C[(x_1, y_1), ..., (x_n, y_n)]$ decomposes as in

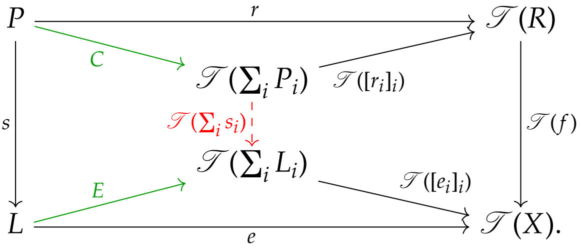
$$\begin{array}{ccc} C[(x_1, y_1), ..., (x_n, y_n)] & \xmapsto{\mathscr{T}(f)} & C[x_1, ..., x_n] \\ {\scriptstyle E[(e_1,f_1),...,(e_n,f_n)]:L} \downarrow & & \downarrow {\scriptstyle E[e_1,...,e_n]:L} \\ D[(x'_1, y'_1), ..., (x'_n, y'_n)] & \xmapsto[\mathscr{T}(f)]{} & D[x'_1, ..., x'_n]. \end{array}$$

- R is a bisimulation, so find $(f_1, ..., f_n)$ as shown.
- That's the intuition. In practice:
  - transition contexts E are not first-class citizens,
  - $\rightsquigarrow$ induction on C.

In the abstract framework

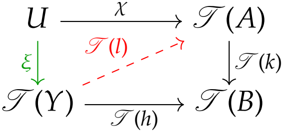
$$
\begin{array}{ccc}
P & \xrightarrow{\quad r \quad} & \mathscr{T}(R) \\
{\scriptstyle s}\big\downarrow & & \big\downarrow{\scriptstyle \mathscr{T}(f)} \\
L & \xrightarrow[\quad e \quad]{} & \mathscr{T}(X).
\end{array}
$$

In the abstract framework



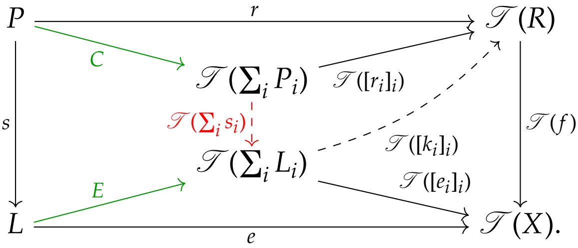
Familiality! Intuition: generic = linear context

Any $U \to \mathscr{T}(X)$ factors as $U \xrightarrow{\xi} \mathscr{T}(Y) \xrightarrow{\mathscr{T}(f)} \mathscr{T}(X)$ with ξ generic:

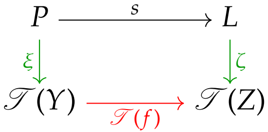


(meaning $k \circ l = h$)

In the abstract framework



$\mathbf{T}_s$-familiality



If ξ and ζ are generic, then $f \in {}^{\boxslash}(\mathbf{T}_s^{\boxslash})$.

Example of $\mathbf{T}_s$-familiality: $\mathscr{T}_{CCS}$

Why would $\mathscr{T}_{CCS}$ be $\mathbf{T}_s$-familial?

Example

$$\begin{array}{ccc} [0] & \xrightarrow{\quad s \quad} & [\tau] \\ {\scriptstyle (\square_1 | \square_2)} \downarrow & & \downarrow {\scriptstyle (\square_1 \rhd \square_2)} \\ \mathscr{T}_{CCS}([0] + [0]) & \xrightarrow[\mathscr{T}_{CCS}(s^{\bar{a}} + s^{a})]{} & \mathscr{T}_{CCS}([\bar{a}] + [a]) \end{array}$$

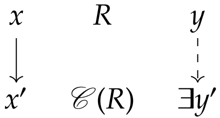
$\mathbf{T}_s$-familiality here: $s^{\bar{a}}, s^{a} \in \mathbf{T}_s$ so $s^{\bar{a}} + s^{a} \in {}^{\boxslash}(\mathbf{T}_s^{\boxslash})$.

Progression

Standard definition of bisimulation up to context

R progresses to $\mathscr{C}(R)$, where

$$\mathscr{C}(R) := \{(C[P_1, ..., P_n], C[Q_1, ..., Q_n]) \mid P_i \; R \; Q_i\}.$$



Progression

Standard definition of bisimulation up to context

R progresses to $\mathscr{C}(R)$, where

$$\mathscr{C}(R) := \{(C[P_1, ..., P_n], C[Q_1, ..., Q_n]) \mid P_i \; R \; Q_i\}.$$

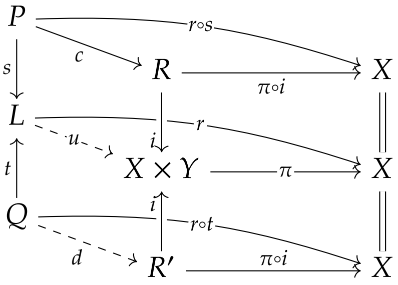
$$\begin{array}{ccc} x & R & y \\ \downarrow & & \downarrow \\ x' & R' & \exists y' \end{array}$$

Generalises to R progresses to R'.

Progression in the abstract framework

Definition

- Relations $R, R' \hookrightarrow X \times Y$ in transition category.
- $R \rightsquigarrow R'$ iff



and symmetrically for Y.

Example: bisimulation up to context

$R \rightsquigarrow \mathscr{T}(R)$.

But wait...

Question

Does $R \rightsquigarrow R$ iff R is a bisimulation?

- Not quite, but artefact of formalism.
- Reason: in $R \rightsquigarrow R$, $R \hookrightarrow X \times Y$ may have no transition.
- Good news, we can add them:

Proposition

Under mild hypotheses, factors as

$$R \to \overline{R} \to X \times Y$$

with $\overline{R}$ a bisimulation.

Soundness of bisimulation up to context

Theorem

Under hypotheses, $R \rightsquigarrow \mathscr{T}(R)$ *entails* $\mathscr{T}(R) \rightsquigarrow \mathscr{T}(R)$.

Corollary

Any bisimulation up to context embeds into some bisimulation.

Example with binding: π-calculus syntax

- Need to refine the base category.
- Let's start with the part about syntax:

$$\ldots \qquad [a] \underset{t^a}{\overset{s^a}{\leftleftarrows}} \textcolor{red}{[0]}, \qquad \ldots \qquad (a \in A)$$

  i.e., the terminal category $\textcolor{red}{1}$.
- $\widehat{1} = \mathbf{Set}$: CCS syntax is a monad over sets

$$\mathscr{T}_{CCS}(X) = \{\text{terms with free variables in } X\}.$$

- Known solution (Fiore?): index over finite sets of names.

$$\mathscr{T}_{\pi}(X)(\gamma) = \{\text{terms with free variables in } X \text{ and free names in } \gamma\}.$$

Example with binding: π-calculus syntax

Base category for π-calculus syntax

Morally: opposite of $\mathscr{P}_f(\mathscr{N})$.

- Objects: all $\gamma \subseteq_f \mathscr{N}$.
- Morphisms $\gamma \to \gamma'$: bijections $\gamma' \to \gamma$ (think contravariance).

Monad for π-calculus syntax:

$$\mathscr{T}_\pi(X)(\gamma) = \{\gamma \vdash_X P\}$$

where $X \in \widehat{\mathscr{P}_f(\mathscr{N})}$.

$$\frac{x \in X(\gamma) \qquad f \in \mathbf{Set}(\gamma, \gamma')}{\gamma' \vdash_X (\!|x|\!)(f)} \qquad \frac{\gamma, b \vdash_X P \qquad a \in \gamma}{\gamma \vdash_X a(b).P}$$

$$\frac{\gamma \vdash_X P \qquad \gamma \vdash_X Q}{\gamma \vdash_X P|Q} \qquad \frac{}{\gamma \vdash_X 0} \qquad \frac{\gamma, a \vdash_X P}{\gamma \vdash_X \nu a.P} \qquad \frac{\gamma \vdash_X P \qquad a, b \in \gamma}{\gamma \vdash_X \overline{a}\langle b\rangle.P}$$

Example with binding: π-calculus syntax

Meaning of

$$\frac{x \in X(\gamma) \qquad f \in \mathbf{Set}(\gamma, \gamma')}{\gamma' \vdash_X (\!|x|\!)(f)}:$$

renaming

- isn't required from all presheaves,
- in particular not from bisimulation relations,
- but is required from $\mathscr{T}_\pi$-algebras.

In other words: it is a syntactic operation.

Example with binding: π-calculus transitions

- Formally add objects for transition types:
  - τ_γ: silent transition between processes with names in γ,
  - $\bar{a}\langle b\rangle_\gamma$: output b on a, with $a, b \in \gamma$,
  - $\bar{a}(\nu c)_\gamma$: output new c on a, with $a \in \gamma$ but $c \notin \gamma$,
  - ...
- and source and target morphisms

$$\gamma \xrightarrow{s} \tau_\gamma \xleftarrow{t} \gamma \qquad \gamma \xrightarrow{s} \bar{a}\langle b\rangle_\gamma \xleftarrow{t} \gamma \qquad \gamma \xrightarrow{s} a(b)_\gamma \xleftarrow{t} \gamma$$

$$\gamma \xrightarrow{s} \bar{a}(\nu c)_\gamma \xleftarrow{t} \gamma, c \qquad \gamma \xrightarrow{s} a(\nu c)_\gamma \xleftarrow{t} \gamma, c$$

- plus compatible morphisms for bijective renaming (next slide).

Example with binding: a few rules

$$\frac{e \in X(\alpha) \qquad \gamma \xrightarrow{s} \alpha \xleftarrow{t} (\gamma, \delta) \qquad h\colon \gamma \xrightarrow{\simeq} \gamma' \qquad k\colon \delta \xrightarrow{\simeq} \delta' \qquad \gamma' \cap \delta' = \varnothing}{(\!|e|\!)(h,k) : \gamma' \vdash_X (\!|e \cdot s|\!)(h) \xrightarrow{(h,k)\cdot\alpha} \gamma', \delta' \vdash_X (\!|e \cdot t|\!)(h+k)}$$

Example: renaming a bound name, $k\colon \{c\} \xrightarrow{\simeq} \{d\}$

$$\begin{array}{ccccc} \gamma & \xrightarrow{s} & \bar{a}(\nu c) & \xleftarrow{t} & \gamma, c \\ {\scriptstyle h=id_\gamma}\uparrow & & \uparrow{\scriptstyle (id_\gamma, k)} & & \uparrow{\scriptstyle id_\gamma + k} \\ \gamma' & \xrightarrow[s]{} & (id_\gamma, k)\cdot \bar{a}(\nu c) = \bar{a}(\nu d) & \xleftarrow[t]{} & \gamma', d. \end{array}$$

Example with binding: a few rules

$$\frac{\gamma, b \vdash_X P \qquad a, c \in \gamma}{in^{a,c}_{b.P} : \gamma \vdash_X a(b).P \xrightarrow{a(c)_\gamma} \gamma \vdash_X [b \mapsto c] \cdot P}$$

- Renaming defined inductively.
- $[b \mapsto c]$ means $\gamma, b \xrightarrow{[id,c]} \gamma$.
- Base case:

$$g \cdot (\!|x|\!)(f) = (\!|x|\!)(g \circ f).$$

Example with binding: a few rules

$$\frac{R : \gamma \vdash_X P \xrightarrow{\bar{a}\langle b\rangle_\gamma} \gamma \vdash_X P' \qquad S : \gamma \vdash_X Q \xrightarrow{a(b)_\gamma} \gamma \vdash_X Q'}{R \rhd S : \gamma \vdash_X (P|Q) \xrightarrow{\tau_\gamma} \gamma \vdash_X (P'|Q')}$$

Familiality

Proposition

$\mathscr{T}_\pi$ is a $\mathbf{T}_s$-familial monad.

Example

$$\begin{array}{ccccc}
\gamma & \xrightarrow{\;s\;} & a(c) & \xleftarrow{\;t\;} & \gamma \\
{\scriptstyle a(b).(\!|\square|\!)(id)}\big\downarrow & & \big\downarrow{\scriptstyle in^{a,c}_{b.(\!|\square|\!)(id)}} & & \big\downarrow{\scriptstyle (\!|\square|\!)[b\mapsto c]} \\
\mathscr{T}_\pi(\gamma, b) & \xrightarrow[\mathscr{T}_\pi(id)]{} & \mathscr{T}_\pi(\gamma, b) & \xleftarrow[\mathscr{T}_\pi(id)]{} & \mathscr{T}_\pi(\gamma, b)
\end{array}$$

with $b \notin \gamma$.

Familiality

Proposition

$\mathscr{T}_\pi$ is a $\mathbf{T}_s$-familial monad.

Example

$$
\begin{array}{ccccc}
\gamma & \xrightarrow{\quad s \quad} & \tau_\gamma & \xleftarrow{\quad t \quad} & \gamma \\
{\scriptstyle (\!|\square_1|\!)(id) | (\!|\square_2|\!)(id)}\downarrow & & \downarrow{\scriptstyle (\!|\square_1|\!)(id,id) \rhd (\!|\square_2|\!)(id,id)} & & \downarrow{\scriptstyle (\!|\square_1|\!)(id) | (\!|\square_2|\!)(id)} \\
\mathscr{T}_\pi(\gamma + \gamma) & \xrightarrow[\mathscr{T}_\pi(s+s)]{} & \mathscr{T}_\pi(\overline{a}\langle b \rangle + a(b)) & \xleftarrow[\mathscr{T}_\pi(t+t)]{} & \mathscr{T}_\pi(\gamma + \gamma)
\end{array}
$$

Failure of congruence

- But bisimilarity is not a congruence in π: what fails?
- Congruence theorem holds under the

Crucial hypothesis

The considered algebra $\mathscr{T}(X) \to X$ is compositional, i.e., is a functional bisimulation.

- For π, the considered algebra is

$$\mu_0 \colon \mathscr{T}_\pi(\mathscr{T}_\pi(0)) \to \mathscr{T}_\pi(0),$$

which is not a functional bisimulation.

Failure of congruence

Adapting a well-known counterexample

- Let $p = (\overline{a}|b)$ (for $a \neq b$) and $f : \{a, b\} \to \{a\}$.
- Unmatched transition:

$$
\begin{array}{ccc}
(\!|p|\!)[b \mapsto a] & \stackrel{\mu_0}{\longmapsto} & \overline{a}|a \\
 & & \downarrow{\scriptstyle \tau_{\{a\}}} \\
 & & 0|0.
\end{array}
$$

Summary

Main example	General case
cat. of transition systems functional bisimulation	transition category $\mathbf{T}_s^{\boxslash}$
SOS specification model partial transition proof	monad algebra Kleisli morphism

- Hypotheses $\Longrightarrow$
  - congruence of bisimilarity,
  - soundness of bisimulation up to context.
- Example with binding: π.

Perspectives

- Existing formats $\rightsquigarrow$ instances?
- More general format along the lines of free monads.

Questions

- "Free" familial monads?
- Why would a free μ be a functional bisimulation?

- Other up-to techniques.
- Related questions, e.g., weak bisimulation, process equations, environmental bisimulation, Howe.
- Broader scope: analytic monads, to accomodate structural congruence.
- Go quantitative?

Progression by lifting in cospans