Insegnamento di

Foundation of Computing

Pagina del corso: http:/pages.di.unipi.it/montanari/FOC.html

Note di

Pi Calcolo

a cura di
Ugo Montanari
Dipartimento di Informatica
Università degli Studi di Pisa
ugo@di.unipi.it
In the example there are only two possible states for the communication centre. A car can (recursively) talk on the channel assigned currently by the communication base, and send them to the car on channel \( cSB \). The system can be represented by the expression:

\[
\text{SYSTEM} \triangleq \text{CAR}(\text{talk}_1, \text{switch}_1) \mid \text{BASE}_1 \mid \text{IDLEBASE}_2 \mid \text{CENTRE}_1.
\]

Let us start with an example which illustrates how the general mechanism for handling name mobility makes the formalisation of the semantics of the -calculus more complicated than that of CCS, especially for allowing it to give \( \text{BASE}_i \) and send them to the car on channel \( \text{BASE}_i \) and Bob \( \text{BASE}_i \) and then uses \( \text{BASE}_i \) and Bob \( \text{BASE}_i \) and then uses

\[
\text{CAR}(\text{talk}, \text{switch}) \triangleq \text{\texttt{talk}}.\text{CAR}(\text{talk}, \text{switch}) + \text{\texttt{switch}}(x, y).\text{CAR}(x, y).
\]

\[
\text{BASE}_i \triangleq \text{\texttt{talk}}_i.\text{BASE}_i + \text{\texttt{give}}_i(x, y).\text{\texttt{switch}}_i(x, y).\text{IDLEBASE}_i
\]

\[
\text{IDLEBASE}_i \triangleq \text{\texttt{alert}}_i.\text{BASE}_i.
\]

\[
\text{CENTRE}_1 \triangleq \text{\texttt{give}}_1(\text{\texttt{talk}}_2, \text{\texttt{switch}}_2).\text{\texttt{alert}}_2.\text{CENTRE}_2
\]

\[
\text{CENTRE}_2 \triangleq \text{\texttt{give}}_2(\text{\texttt{talk}}_1, \text{\texttt{switch}}_1).\text{\texttt{alert}}_1.\text{CENTRE}_1.
\]
Operational semantics of the TT-calculus

\[
p ::= \text{nil} \mid \Delta p \mid [x=y]p \mid p+p \mid plp \mid (\exists) p \mid !p
\]

\[
\chi ::= \tau \mid x(x) \mid \overline{xy}
\]

\[
\begin{align*}
\tau p & \xrightarrow{\tau} p \\
x(y)p & \xrightarrow{(w)} p[w/x] \quad \nu \neq \Pi ((x)p) \quad \overline{xy} p & \xrightarrow{\overline{xy}} p \\
p & \xrightarrow{q} p' \\
q & \xrightarrow{q} p' \\
plq & \xrightarrow{\Pi q} p'q' \\
plq & \xrightarrow{\nu q} p'q'
\end{align*}
\]

Example

\[
\begin{align*}
\overline{xy} p & \xrightarrow{\overline{xy}} p \\
(\exists)\overline{xy} p & \xrightarrow{(v)} plv/\exists x \\
(\exists)\overline{xy} p & \xrightarrow{(v)} plv/\exists x \\
(\exists)\overline{xy} p & \xrightarrow{(v)} plv/\exists x \\
(\exists)\overline{xy} p & \xrightarrow{(v)} plv/\exists x \\
((\exists)\overline{xy} p) & \xrightarrow{(v)} plv/\exists x \\
(\exists)\overline{xy} p & \xrightarrow{(v)} plv/\exists x \\
((\exists)\overline{xy} p) & \xrightarrow{(v)} plv/\exists x \\
(\exists)\overline{xy} p & \xrightarrow{(v)} plv/\exists x \\
((\exists)\overline{xy} p) & \xrightarrow{(v)} plv/\exists x \\
(\exists)\overline{xy} p & \xrightarrow{(v)} plv/\exists x \\
\end{align*}
\]
Example 13.3 (Scope extrusion). We conclude this section by showing an example of the use of the rule system. Let us consider the following system:

\[
(((y)\overline{x}.p) \mid q) \mid x(z).r
\]

where \( p, q, r \) are \( \pi \)-calculus processes. The process \((y)\overline{x}.p\) would like to set up a private channel with \(x(z).r\), which however should remain hidden to \(q\). By using the inference rules of the operational semantics we can proceed in a goal-oriented fashion to find a derivation for the corresponding transition:

\[
(((y)\overline{x}.p) \mid q) \mid x(z).r \xrightarrow{\alpha} s
\]

\[
\xrightarrow{(\text{CloseL}, \alpha=\tau, s=(w)(s_1 \mid r_1)} (((y)\overline{x}.p) \mid q \xrightarrow{\overline{x}(w)} s_1, x(z).r \xrightarrow{x(w)} r_1)
\]

\[
\xrightarrow{(\text{ParL}, s_1=p_1 \mid q, w\notin\text{fn}(q)} (y)\overline{x}.p \xrightarrow{\overline{x}(w)} p_1, x(z).r \xrightarrow{x(w)} r_1
\]

\[
\xrightarrow{(\text{Open}, p_1=p_2[w/y], w\notin\text{fn}((y).p)} \overline{x}.p \xrightarrow{\overline{x}y} p_2, x(z).r \xrightarrow{x(w)} r_1
\]

so we have:

\[
p_2 = p
\]

\[
p_1 = p_2[w/y] = p[w/y]
\]

\[
r_1 = r[w/z]
\]

\[
s_1 = p_1 \mid q = p[w/y] \mid q
\]

\[
s = (w)(s_1 \mid r_1) = (w)( (p[w/y] \mid q) \mid (r[w/z]) )
\]

\[
\alpha = \tau
\]

In conclusion:

\[
(((y)\overline{x}.p) \mid q) \mid x(z).r \xrightarrow{\tau} (w)( (p[w/y] \mid q) \mid (r[w/z]) )
\]

under the condition that \(w\) is fresh, i.e., that \(w \notin \text{fn}(q) \cup \text{fn}((y)p) \cup \text{fn}((z)r)\).
Bisimulation

Early bisimilar processes

Formally, a binary relation $S$ on $\pi$-calculus agents is a strong early ground bisimulation if:

\[
\forall p, q. \ p \ S \ q \Rightarrow \begin{cases} 
\forall \alpha, p'. \text{ if } p \xrightarrow{\alpha} p' \text{ with } \alpha \neq x(y) \land \text{bn}(\alpha) \cap \text{fn}(q) = \emptyset, \\
\text{then } \exists q'. q \xrightarrow{\alpha} q' \text{ and } p' \ S \ q' \\
\forall x, y, p'. \text{ if } p \xrightarrow{x(y)} p' \text{ with } y \notin \text{fn}(q), \\
\text{then } \forall w. \exists q'. q \xrightarrow{x(y)} q' \text{ and } p'[^w/y] \ S \ q'[^w/y] \\
\text{(and vice versa)}
\end{cases}
\]

Late bisimilar processes

\[
\forall x, y, p'. \text{ if } p \xrightarrow{x(y)} p' \text{ with } y \notin \text{fn}(q), \\
\text{then } \exists q'. q \xrightarrow{x(y)} q' \text{ and } \forall w. \ p'[^w/y] \ S \ q'[^w/y]
\]

Processes which are early but not late bisimilar

\[
p \overset{\text{def}}{=} x(y).\tau.\text{nil} + x(y).\text{nil} \quad q \overset{\text{def}}{=} p + x(y).[y = z]\tau.\text{nil}
\]

whose transitions are (for any fresh name $u$):

\[
\begin{align*}
& p \xrightarrow{x(u)} \tau.\text{nil} & q \xrightarrow{x(u)} \tau.\text{nil} \\
& p \xrightarrow{x(u)} \text{nil} & q \xrightarrow{x(u)} \text{nil} \\
& q \xrightarrow{x(u)} [u = z]\tau.\text{nil}
\end{align*}
\]
**Example 13.6 (Ground bisimilarities are not congruences).** Let us consider the following agents:

\[ p \overset{\text{def}}{=} \overline{x}.nil \mid x'(y).nil \quad q \overset{\text{def}}{=} \overline{x}.x'(y).nil + x'(y).\overline{x}.nil \]

We leave the reader to check that the agents \( p \) and \( q \) are bisimilar (according to both early and late bisimilarities). Now, in order to show that ground bisimulations are not congruences, we define the following context:

\[ C[\cdot] = z(x').[\cdot] \]

by plugging \( p \) and \( q \) inside the hole of \( C[\cdot] \) we get:

\[ C[p] = z(x').(\overline{x}.nil \mid x'(y).nil) \quad C[q] = z(x').(\overline{x}.x'(y).nil + x'(y).\overline{x}.nil) \]

\( C[p] \) and \( C[q] \) are not early bisimilar (and thus not late bisimilar). In fact, suppose the name \( x \) is received on \( z \): we need to compare the agents

\[ p' \overset{\text{def}}{=} \overline{x}.nil \mid x(y).nil \quad q' \overset{\text{def}}{=} \overline{x}.x(y).nil + x(y).\overline{x}.nil \]

Now \( p' \) can perform a \( \tau \)-transition, but \( q' \) cannot.

The problem illustrated by the previous example is due to aliasing, and it appears often in programming languages with both global variables and parameter passing to procedures. It can be solved by defining a finer relation between agents called strong early full bisimilarity and defined as follows:

\[ p \simeq_E q \quad \iff \quad p\sigma \simeq_E q\sigma \text{ for every substitution } \sigma \]

where a substitution \( \sigma \) is a function from names to names that is equal to the identity function almost everywhere (i.e., it differs from the identity function only on a finite number of elements of the domain).

Analogously, we can define strong late full bisimilarity \( \simeq_L \) by letting

\[ p \simeq_L q \quad \iff \quad p\sigma \simeq_L q\sigma \text{ for every substitution } \sigma \]
The operational semantics of CCS because it needs to handle name passing and scope extrusion. By exploiting a finite number of names.

Lemma 13.1 (Harmony Lemma).

Then, a reduction is either a silent action performed by some \( \tau \) in one step. The rules defining the relation as a binary relation on processes using the notation \( R = \{ (p, q) \mid p \equiv q \} \) are the following:

\[
\begin{align*}
\text{null} + \text{null} & \equiv \text{null} \\
\text{null} \, \| \, \text{null} & \equiv \text{null} \\
(x)\text{null} & \equiv \text{null} \\
[x = y] \text{null} & \equiv \text{null} \\
\text{null} + q & \equiv q + p \\
\text{null} \, \| \, q & \equiv q \, \| \, p \\
(p + q) + r & \equiv p + (q + r) \\
(p \, \| \, q) \, \| \, r & \equiv p \, \| \, (q \, \| \, r) \\
(x)(p \, \| \, q) & \equiv p \, \| \, (x)q \text{ if } x \notin \text{fn}(p) \\
\text{null} \, \| \, !p & \equiv !p
\end{align*}
\]
Categorical Semantics of $\Pi$-Calculus

$\Pi$-calculus: names/naive generation in addition

Some problems

$\frac{p \rightarrow p'}{p \parallel q \rightarrow p' \parallel q}$

So implies $p \rightarrow p'$ with $bn(d) \eta_{fn}(q) = \phi$...

This does not correspond to a coalgebra

Finitary sets equipped with actions as

$\mathbf{FinSet}$

$\mathbf{FinSet}$ are algebras with finite-Kuratowski partitions as operations

$\text{Aut}(f) = \{T : \omega \rightarrow \omega \mid \frac{1}{2} x \in \omega \mid p(x) \neq x \}$

The kernel

$\text{id}(x) = x$

$\Pi_1(T_2)(x) = (\Pi_1 \circ T_2)(x)$

Permutation algebra

$\langle A, \Sigma_{T} : A \rightarrow A \mid \Pi \in \text{Aut}(\Sigma) \rangle$

$\text{orb}_{A}(a) = \{ \Pi A(a) \mid \Pi \in \text{Aut}(\Sigma) \}$
Symmetry of an element of a Poincaré Algebra

\[ G_A(a) = \left\{ \pi \in \text{Aut} f \mid \pi_A(a) = a \right\} \]

identity group of \( X \subseteq \omega \)

\[ \text{fix} (X) = \left\{ \pi \in \text{Aut} f \mid \pi_x = \pi x \right\} \]

\( X \) supports \( a \) if \( \forall x \in X \text{fix} (x) \subseteq G_A(a) \)

\( \text{supp}_A(a) \) is the minimal such \( X \)

\[ \text{Alg} = \text{FSAAlg} \] are categories of algebras

There is a theory morphism (injective!) \( \cong^{+} : \text{Alg} \to \text{Alg}^{\text{right-shift}} \)

\[ \pi^{+1} (i) = \begin{cases} 0 & i = 0 \\ \pi (i) + 1 & \text{otherwise} \end{cases} \]

\( \textbf{S} : \text{Alg} \to \text{Alg}^{\text{right-shift}} \) is the forgetful functor on the right-shift morphism

syntax of \( \lambda \)-calculus : De Bruijn indices

\[ \lambda \equiv \lambda \times. L / \lambda \] \[ \lambda L. \lambda g \cdot (x g) \equiv \lambda L. (\lambda) \]

\( X \)

\[ T(x) = S(x) + x \times x + w \]

\( \pi(x) \) \[ \text{fixed point} \; X = T(x) \] is the initial algebra
Similarly for the $\mathcal{H}$-calculus:

$$T\xrightarrow{} X = K + \omega \times \omega \times X + \omega \times \delta X + X \times X + \omega \times \delta X + \cdots$$

must be output. Input, serial, restrictions.

Initial algebra $\mathcal{H}$.

Now, coalgebras! Where?

**First option:** coalgebra in $\mathcal{H}^*$. Consider

$$G(x) = \{ \kappa \in L \times X \cup L' \times \delta X \}$$

This may allocate fresh names. LTS must be ranked.

**Second option:** coalgebra in $\mathcal{H}^*$. Consider

Brain is not a sequence $\Rightarrow$ no lifting possible.

**Fix #1:** eliminate prefix

Say $P+\delta(x)$ is a constant, with DeSmith anxious.
continues well with recursion.

**Fix #2:** Move from $\mathcal{H}^*$ to $\mathcal{H}^*$

Additional necessities $\quad p \in \mathcal{X} \quad p \in \mathcal{X}$

or equivalent:  $\quad p \in \mathcal{X} \quad \to \quad p \in \mathcal{X} \to \mathcal{X}$
Bialgebraic semantics with axioms

We stay in the category \( \mathsf{Alg}(E) \) and functor \( P_\Delta \) defined by the SOS rules:

\[
\begin{align*}
A &\to B \\
\text{alg. } B &\text{ gives the states of the transition systems} \\
&\text{i.e. } B = T_{E,E} \text{ with } k \text{ surjective} \\
\text{where } E_n &\text{ are the structural axioms}
\end{align*}
\]

The actual transition system is just defined in \( \mathsf{Set} \):

\[
\begin{align*}
\mathsf{Set} &\mid B \mid \\
\downarrow &\downarrow \quad \text{where } g \text{ is defined by } A_k \\
\Rightarrow &\Rightarrow \\
P_E(B) &\text{ the signature rules plus role STRUCT:} \\
P_E(B) &\text{ by } p = p' \Rightarrow q' \Rightarrow q = q' \\
P &\Rightarrow q
\end{align*}
\]

Assume that the diagrams commute in \( \mathsf{Set} \):

\[
\begin{align*}
\mathsf{Set} &\mid A \mid \to \mid B \mid \\
\downarrow &\downarrow \\
P_A(A) &\to P_A(B) \\
P_E(A) &\Rightarrow P_E(B) \\
P &\Rightarrow q \\
\text{where } g &\text{ is the } \rightarrow \text{ arrow}
\end{align*}
\]

In fact, since \( h \downarrow \) and \( P_\Delta(h) \) are homomorphisms and \( h \) is surjective, then also \( g \) is a homomorphism:

\[
\begin{align*}
g(h(a)) &= g(h(a))' = (P_\Delta(h)(f(a)) = P_\Delta(h)(f(a)) = P_\Delta(h)(f(a)) = P_\Delta(h)(f(a)) = P_\Delta(h)(f(a)) = P_\Delta(h)(f(a))
\end{align*}
\]

\( b = h(a) \) commutes with \( g \).
To prove that the diagram commutes

\[ \begin{array}{ccc}
    A & \xrightarrow{h'} & B \\
    g & \downarrow & \mathcal{E} \\
    P \circ (A) & \xrightarrow{P \circ (h')} & P \circ (B)
\end{array} \]

corresponds it show that the kernel of \( h \)
(i.e., the equivalence \( u = v \iff h(u) = h(v) \))
is a bimimulation.

To do this, it is enough to prove that \( E \) is
all the axioms in \( E \) bimimulate exactly

\[ L = R \quad \forall E \]

implies that for every DeGroote proof

\[ \begin{array}{c}
    \text{prec} \\
    L \Rightarrow q_1 \\
    \text{another D.S. proof} \\
    R \Rightarrow q_2
\end{array} \]

starting from the weakest of premises "prec".

(i) The kernel of \( h \) is a bimimulation: \( p \Rightarrow q \Rightarrow p \Rightarrow q \) roughly dual
(ii) The diagram commutes in \( k \)
(iii) The diagram can be lifted to \( H \)
Saturated via Right Kan Extension

Construction by Fiore et al.

Instead of \( \mathcal{HP} \) use \( \text{Set}^I \) the presheaf where the exponent is \( I \), the category of finite sets with injective mappings.

\( \text{Set}^I \) is equivalent to \( \mathcal{HP} \) since it has only monadic operations.

However also \( \text{Set}^I \) has the same problems: no lifting to \( \mathcal{HP} \). We could consider \( \text{Set}^F \) (all mappings), but in this case the LTS of \( \mu \)-calculus could not be lifted to a coalgebra in \( \text{Set}^F \) for the usual reasons: bisimulation is not a congruence for lenses.

The forgetful functor \( \text{Set}^F \to \text{Set}^I \) \( L(A) = V; A \) has not only a left, but also a right adjoint \(( \text{Kan Extension}) \Gamma \text{-} I\)

Now a behavioral functor \( B \) in \( \text{Set}^I \) can be lifted to \( \hat{B}(P) = [B(LP); I] \) and a coalgebra \((LP, P)\) for \( B \) can be lifted to a coalgebra \((P, [P] \circ \eta_P)\) for \( \hat{B} \)

\( \Gamma_P : [LP; I] \to [B(LP); I] = \hat{B}(P) \quad \eta_P : P \to [P] \) is unity of the adjunction.
The lifted coalgebra has richer labels
\((\mathcal{C}, \lambda)\) where \(C \in F[N_1 \to N_2]\) (possibly a fusion)
and \(\lambda\) is a label of \(B\), we have

\[ N_1 \vdash p \xrightarrow{(\lambda, \alpha)} N_1 \vdash p' \iff N_2 \vdash p[\alpha] \xrightarrow{(\text{id}, \lambda)} N_1 \vdash p' \]

where the second transition has a corresponding transition \(N_2 \vdash p[\alpha] \xrightarrow{\text{id}} N_1 \vdash p'\) in \((\mathcal{LPI}, p)\).

Thus lifting adds transitions (saturation) which correspond to observe the effects of all the arrows in the index category. Thus similar states must behave corresponding ly for all such substitutions.
**History - Dependent Automata**

\[ P \xrightarrow{x} q \]

\[ \left( \exists P \xrightarrow{x} q \right) \xrightarrow{y} [P] \]

- Infinite branching

- Choose one for finite state, different from all the others.

\[ P \xrightarrow{x} q \]

\[ \rightarrow \]

- Choose the same \& 1

**OK, actually, doesn't work for minimization.**

**HDAutotama** local names

![Diagram](image)

- Injection: \( n_q \rightarrow n_{q, \#} \)

**Biimmune set:** set of states with partial bijection

- Group of symmetries: is similar to itself with a p. bijection different from \( id \)

**Coalgebraic theory based on Named Sets**

\[ N = \langle Q_N, S_N \rangle \]

\[ \sigma : Q_N \rightarrow \text{Symset} \]

- Named sets are compact representations of biinfinite categories

- Named sets are compact representations of biinfinite categories in named set with Alg but smaller objects