Università degli Studi di Pisa
Corso di Laurea Magistrale in Informatica

Anno Accademico 2021-2022

Insegnamento di

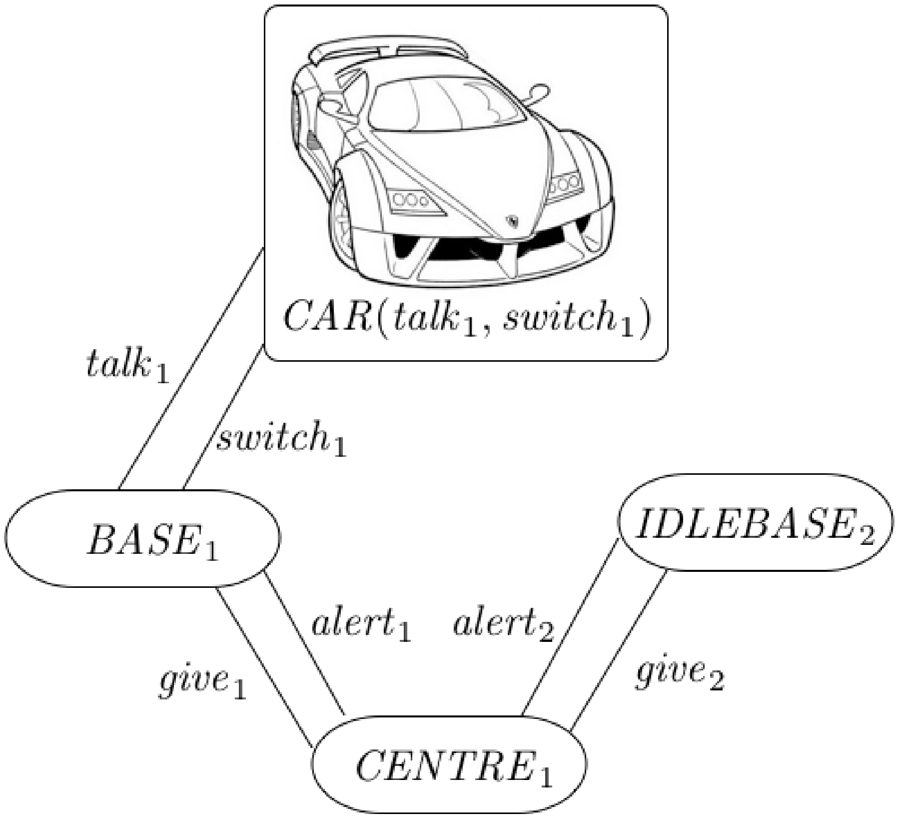
Foundation of Computing

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



$$CAR(talk, switch) \stackrel{\text{def}}{=} \overline{talk}.CAR(talk, switch) + switch(xt, xs).CAR(xt, xs).$$

$$BASE_i \stackrel{\text{def}}{=} talk_i.BASE_i + give_i(xt, xs).\overline{switch_i}(xt, xs).IDLEBASE_i$$
$$IDLEBASE_i \stackrel{\text{def}}{=} alert_i.BASE_i.$$

$$CENTRE_1 \stackrel{\text{def}}{=} \overline{give_1}\langle talk_2, switch_2\rangle.\overline{alert_2}.CENTRE_2$$
$$CENTRE_2 \stackrel{\text{def}}{=} \overline{give_2}\langle talk_1, switch_1\rangle.\overline{alert_1}.CENTRE_1.$$

$$SYSTEM \stackrel{\text{def}}{=} CAR(talk_1, switch_1) \mid BASE_1 \mid IDLEBASE_2 \mid CENTRE_1.$$

Operational semantics of the π-calculus

$$p ::= nil \;\big|\; \alpha p \;\big|\; [x=y]p \;\big|\; p+p \;\big|\; p|p \;\big|\; (y)p \;\big|\; !p$$

$$\alpha ::= \tau \;\big|\; x(y) \;\big|\; \bar{x}y$$

$$\tau p \xrightarrow{\tau} p \qquad x(y)p \xrightarrow{x(w)} p\{w/y\} \quad w \notin fn((y)p) \qquad \bar{x}y\, p \xrightarrow{\bar{x}y} p$$

$$\frac{p \xrightarrow{\alpha} p'}{[x=x]p \xrightarrow{\alpha} p'} \qquad \frac{p \xrightarrow{\alpha} p'}{p+q \xrightarrow{\alpha} p'} \qquad \frac{p \xrightarrow{\alpha} p'}{p|q \xrightarrow{\alpha} p'|q} \quad bn(\alpha) \cap fn(q) = \phi$$

+simm

$$\frac{p \xrightarrow{\bar{x}y} p' \quad q \xrightarrow{x(z)} q'}{p|q \xrightarrow{\tau} p'|q'\{y/z\}} \qquad \frac{p \xrightarrow{\alpha} p'}{(y)p \xrightarrow{\alpha} (y)p'} \quad y \notin n(\alpha)$$

$$\frac{p \xrightarrow{\bar{x}y} p' \quad y \neq x}{(y)p \xrightarrow{\bar{x}(w)} p'\{w/y\}} \qquad \frac{p \xrightarrow{\bar{x}(w)} p' \quad q \xrightarrow{x(w)} q'}{p|q \xrightarrow{\tau} (w)(p'|q')}$$

$$\frac{p \,|\, !p \xrightarrow{\alpha} p'}{!p \xrightarrow{\alpha} p'}$$

open   close

Example

$$\frac{\dfrac{\dfrac{\bar{x}y.p \xrightarrow{\bar{x}y} p}{(y)\bar{x}y.p \xrightarrow{\bar{x}(v)} p\{v/y\}} \quad v \notin fn((y)p)}{((y)\bar{x}y.p)|q \xrightarrow{\bar{x}(v)} p\{v/y\}|q} \quad v \notin fn(q) \qquad x(z).r \xrightarrow{x(v)} r\{v/z\} \quad v \notin fn((z)r)}{(((y)\bar{x}y.p)|q)|x(z).r \xrightarrow{\tau} (v)((p\{v/y\}|q)|r\{v/z\})}$$

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:

$$(((y)\bar{x}y.p)\ |\ q)\ |\ x(z).r$$

where p,q,r are π-calculus processes. The process $(y)\bar{x}\,y.p$ 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:

$$
\begin{array}{rl}
\multicolumn{2}{l}{(((y)\bar{x}y.p)\ |\ q)\ |\ x(z).r \xrightarrow{\alpha} s} \\
\nwarrow_{(\mathrm{CloseL}),\ \alpha=\tau,\ s=(w)(s_1\ |\ r_1)} & ((y)\bar{x}y.p)\ |\ q \xrightarrow{\bar{x}(w)} s_1, \quad x(z).r \xrightarrow{x(w)} r_1 \\
\nwarrow_{(\mathrm{ParL}),\ s_1=p_1\ |\ q,\ w\notin\mathrm{fn}(q)} & (y)\bar{x}y.p \xrightarrow{\bar{x}(w)} p_1 \quad x(z).r \xrightarrow{x(w)} r_1 \\
\nwarrow_{(\mathrm{Open}),\ p_1=p_2[^{w}/_{y}],\ w\notin\mathrm{fn}((y).p)} & \bar{x}y.p \xrightarrow{\bar{x}y} p_2, \quad x(z).r \xrightarrow{x(w)} r_1 \\
\overset{*}{\nwarrow}_{(\mathrm{Out})+(\mathrm{In}),\ r_1=r[^{w}/_{z}],\ p_2=p,\ w\notin\mathrm{fn}((z).r)} & \square
\end{array}
$$

so we have:

$$
\begin{aligned}
p_2 &= p \\
p_1 &= p_2[^{w}/_{y}] = p[^{w}/_{y}] \\
r_1 &= r[^{w}/_{z}] \\
s_1 &= p_1\ |\ q = p[^{w}/_{y}]\ |\ q \\
s &= (w)(s_1\ |\ r_1) = (w)(\ (p[^{w}/_{y}]\ |\ q)\ |\ (r[^{w}/_{z}])\ ) \\
\alpha &= \tau
\end{aligned}
$$

In conclusion:

$$(((y)\bar{x}y.p)\ |\ q)\ |\ x(z).r \xrightarrow{\tau} (w)(\ (p[^{w}/_{y}]\ |\ q)\ |\ (r[^{w}/_{z}])\ )$$

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

Bisimulation

Early bisimilar processes

Formally, a binary relation S on π-calculus agents is a *strong early ground bisimulation* if:

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

Late bisimilar processes

$$\begin{aligned} &\forall x,y,p'.\ \text{ if } p \xrightarrow{x(y)} p' \text{ with } y \notin \mathrm{fn}(q), \\ &\qquad\qquad \text{then } \exists q'.\ q \xrightarrow{x(y)} q' \text{ and } \forall w.\ p'[{}^{w}/_{y}]\,S\,q'[{}^{w}/_{y}] \end{aligned}$$

Processes which are **early** but not **late** bisimilar

$$p \stackrel{\text{def}}{=} x(y).\tau.\mathbf{nil} + x(y).\mathbf{nil} \qquad\qquad q \stackrel{\text{def}}{=} p \,+\, x(y).[y = z]\tau.\mathbf{nil}$$

hose transitions are (for any fresh name u):

$$\begin{array}{ll} p \xrightarrow{x(u)} \tau.\mathbf{nil} & \qquad q \xrightarrow{x(u)} \tau.\mathbf{nil} \\ p \xrightarrow{x(u)} \mathbf{nil} & \qquad q \xrightarrow{x(u)} \mathbf{nil} \\ & \qquad q \xrightarrow{x(u)} [u = z]\tau.\mathbf{nil} \end{array}$$

Example 13.6 (Ground bisimilarities are not congruences). Let us consider the following agents:

$$p \stackrel{\text{def}}{=} \bar{x}x.\,\mathbf{nil} \mid x'(y).\,\mathbf{nil} \qquad q \stackrel{\text{def}}{=} \bar{x}x.x'(y).\,\mathbf{nil} + x'(y).\bar{x}x.\,\mathbf{nil}$$

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:

$$C[p] = z(x').(\bar{x}x.\,\mathbf{nil} \mid x'(y).\,\mathbf{nil}) \qquad C[q] = z(x').(\bar{x}x.x'(y).\,\mathbf{nil} + x'(y).\bar{x}x.\,\mathbf{nil})$$

$C[p]$ 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

$$p' \stackrel{\text{def}}{=} \bar{x}x.\,\mathbf{nil} \mid x(y).\,\mathbf{nil} \qquad q' \stackrel{\text{def}}{=} \bar{x}x.x(y).\,\mathbf{nil} + x(y).\bar{x}x.\,\mathbf{nil}$$

Now p' can perform a τ-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:

$$p \simeq_E q \quad \Leftrightarrow \quad p\sigma \sim_E q\sigma \text{ for every substitution } \sigma$$

where a substitution σ 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

$$p \simeq_L q \quad \Leftrightarrow \quad p\sigma \sim_L q\sigma \text{ for every substitution } \sigma$$

Axioms for structural equivalence

$$
\begin{array}{rclcrclcrcl}
p + \mathbf{nil} & \equiv & p & \quad & p + q & \equiv & q + p & \quad & (p+q)+r & \equiv & p+(q+r) \\
p \mid \mathbf{nil} & \equiv & p & & p \mid q & \equiv & q \mid p & & (p \mid q) \mid r & \equiv & p \mid (q \mid r) \\
(x)\,\mathbf{nil} & \equiv & \mathbf{nil} & & (y)(x)p & \equiv & (x)(y)p & & (x)(p \mid q) & \equiv & p \mid (x)q \ \text{ if } \ x \notin \mathrm{fn}(p) \\
[x=y]\,\mathbf{nil} & \equiv & \mathbf{nil} & & [x=x]\,p & \equiv & p & & p \mid !p & \equiv & !p
\end{array}
$$

Categorical Semantics of π-Calculus

CCS : algebra + coalgebra $\Rightarrow$ bialgebra

π-calculus : names/name generation in addition

Some problems

$$\frac{p \xrightarrow{\alpha} p'}{p|q \xrightarrow{\alpha} p'|q} \quad bn(\alpha) \cap fn(q) = \phi \quad \text{is not definable}$$

$p \mathrel{S} q$ implies $p \xrightarrow{\alpha} p'$ with $bn(\alpha) \cap fn(q) = \phi$ ...

this does not correspond to a coalgebra

States with names

FM sets sets equipped with actions as permutations on natural numbers

Fraenkel-Mostowski

FM sets are algebras with finite-kernel permutations as operations

$Autf = \{\pi : \omega \to \omega\}$ $\quad \{x \in \omega \mid \pi(x) \neq x\}$ the kernel

monadic operations $\quad id(x) = x$

with axioms $\quad \pi_1(\pi_2(x)) = (\pi_1 \circ \pi_2)(x)$ $\quad \int \pi_\pi$

Permutation algebras $\quad \langle A, \{\pi_A : A \to A \mid \pi \in Autf\} \rangle$

$$orb_A(a) = \{\pi_A(a) \mid \pi \in Autf\}$$

Symmetry of an element of a Perm. Algebra

$$G_A(a) = \{\pi \in Aut f \mid \pi_A(a) = a\}$$

identity group of $X \subseteq \omega$

$$fix(X) = \{\pi \in Aut f \mid \pi|_X = id_X\}$$

X supports a iff $fix(X) \subseteq G_A(a)$

$supp_A(a)$ is the minimal such X

$\underline{Alg}^{\Pi}$ $\underline{FSAlg}^{\Pi}$ are categories of algebras

There is a theory morphism (injective!) $-^{+1} : \Gamma \to \Gamma$

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

right shift operator

$\delta : Alg^{\Pi} \to Alg^{\Pi}$ is the forgetful functor | name abstraction

associated to the right shift morphism

Syntax of λ calculus : De Bruijn indexes

$$\lambda x.\lambda y.(x\, y) \equiv \lambda.\lambda.(1\, 0)$$

$$L ::= \lambda x.L \mid LL \mid x$$

$$T(X) \to X$$

$$T(X) = \delta(X) + X \times X + \omega$$

Fixpoint $\bar{X} \cong T(\bar{X})$ is the initial algebra

Similarly for the π-calculus

$$TX = \underbrace{K}_{\text{inaction}} + \underbrace{w \times w \times X}_{\text{output}} + \underbrace{w \times \delta X}_{\text{input}} + \underbrace{X \times X}_{\text{parallel}} + \underbrace{\delta X}_{\text{restriction}} + \cdots$$

Initial algebra Z_π

Now, coalgebras! Where?

First option: coalgebras in $\underline{Alg}^{\pi}$!

$$G(X) = P_f\left(L \times X + L' \times \delta X\right)$$

transitions may allocate fresh names. LTS in Set can be lifted

Second option: coalgebras in Alg_T

Bisim is not a congruence $\Rightarrow$ no lifting possible

Fix #1: Eliminate prefix

$\bar{x}y.P + z(w).Q$ is a constant, with De Simone axioms combines well with recursion.

Fix #2: Move from Alg_π to Alg_{fusion}

Additional transitions $p \xrightarrow{[x/y]} p[x/y]$

or equivalent. $p[x/y] \xrightarrow{\mu} q \Rightarrow p \xrightarrow{[x/y],\,\mu} q$

Bialgebraic semantics with axioms

91

We stay in the category $Alg(\Sigma)$ and functor P_Δ defined by the SOS rules

$\underline{Alg(\Sigma)}$

$$\begin{tikzcd} A \arrow[r, "h"] \arrow[d, "f"'] & B \\ P_\Delta(A) & \end{tikzcd}$$

$A = T_\Sigma$
algebra B gives the states of the transition systems, i.e. $B = T_{\Sigma,E}$ with h surjective where E are the structural axioms

The actual transition system is just defined in $\underline{Set}$

$\underline{Set}$

$$\begin{tikzcd} {|B|} \arrow[d, "g"] \\ P_L(|B|) \end{tikzcd}$$

where g is defined by the De Simone rules plus rule STRUCT:

$$\frac{p \equiv_E p' \quad p' \overset{\alpha}{\Longrightarrow} q' \quad q' \equiv_E q}{p \overset{\alpha}{\Longrightarrow} q}$$

Assume that the diagram commutes in $\underline{Set}$

$\underline{Set}$

$$\begin{tikzcd} {|A|} \arrow[r, "h"] \arrow[d, "f"'] \arrow[dr, phantom, "="] & {|B|} \arrow[d, "g"] \\ P_L(|A|) \arrow[r, "P_L(h)"'] & P_L(|B|) \end{tikzcd}$$

then it can be lifted!

$\underline{Alg(\Sigma)}$

$$\begin{tikzcd} A \arrow[r, "h"] \arrow[d, "f"'] \arrow[dr, phantom, "="] & B \arrow[d, "g"] \\ P_\Delta(A) \arrow[r, "P_\Delta(h)"'] & P_\Delta(B) \end{tikzcd}$$

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

$g(\sigma_B(b)) = g(h(\sigma_A(a)) = (P_\Delta(h)(f(\sigma_A(a)) = \sigma_{P_\Delta(A)}(P_\Delta(h)(f(a))) = \sigma_{P_\Delta(A)}(g(h(a))) = \sigma_{P_\Delta(A)}g(b)$

$b = h(a)$

commuta il quadrato

To prove that the diagram commutes

$$
\begin{tikzcd}
{|A|} \arrow[r, "h"] \arrow[d, "f"'] \arrow[dr, phantom, "="] & {|B|} \arrow[d, "g"] \\
P_L(|A|) \arrow[r, "P_L(h)"'] & P_L(|B|)
\end{tikzcd}
$$

corresponds to show that the Kernel^{E} of h

(i.e. the equivalence $u = v \iff h(u) = h(v)$)

is a bisimulation.

To do this, it is enough to prove that E bisimilar ||

all the axioms in E bisimulate, namely

$L = R \quad \in E$

equational proof

$u = u_1 = u_2 = u_3 \cdots \quad u_n = v$

$\uparrow \quad \uparrow$ axiom

implies that for every De Simone proof

$\dfrac{prec}{L \to q_1}$ there must exist another D.S. proof $\dfrac{prec}{R \to q_2}$ with $q_1 \equiv_E q_2$

starting from the same set of premises "prec", and viceversa.

(i) the Kernel of h is a bisimulation: $p \equiv q \Rightarrow p \underline{\sim} q$ purely logical

(ii) the diagram commutes in set

(iii) the diagram can be lifted to Alg

70bis

Saturation via Right Kan Extension

Construction by Fiore et al.

Instead of Alg^{Π} we use Set^{I}: the presheaf where the exponent is I, the category of finite sets with injective mapping.

Set^{I} is equivalent to Alg^{Π}, since Π has only monadic operations.

However also Set^{I} has the same problem: no lifting to Alg_T. We could consider Set^{F} (all mappings), but in this case the LTS of π-calculus could not be lifted to a coalgebra in Set^{F} for the usual reason: bisimulation is not a congruence for fusions.

The forgetful functor $Set^{F} \xrightarrow{\lfloor - \rfloor} Set^{I}$ $\quad \lfloor A \rfloor = V;A$

has not only a left, but also a $\quad V: F \to I$

right adjoint (left Kan Extension) $\lceil \_ \rceil$

$$\begin{tikzcd} Set^{F} \arrow[r, bend left, "\lfloor - \rfloor"] \arrow[r, phantom, "\perp"] & Set^{I} \arrow[l, bend left, "\lceil - \rceil"] \end{tikzcd}$$

Now a behavioral functor B in Set^{I} can be lifted to $\hat{B}(P) = \lceil B(\lfloor P \rfloor) \rceil$ and a coalgebra $(\lfloor P \rfloor, \rho)$ for B can be lifted to a coalgebra $(P, \lceil \rho \rceil \circ \eta_P)$ for $\hat{B}$

$\lceil \rho \rceil : \lceil \lfloor P \rfloor \rceil \to \lceil B(\lfloor P \rfloor) \rceil = \hat{B}(P)$ $\quad \eta_P : P \to \lceil \lfloor P \rfloor \rceil$ the unity of the adjunction.

The lifted coalgebra has richer labels (σ, α) where $\sigma \in F[N_1 \to N_2]$ (possibly a fusion) and α is a label of B. We have

$$N_1 \vdash p \xrightarrow{(\sigma,\alpha)} N \vdash p' \iff N_2 \vdash p[\sigma] \xrightarrow{(id,\alpha)} N \vdash p'$$

where the second transition has a corresponding transition $N_2 \vdash p[\sigma] \xrightarrow{\alpha} N \vdash p'$ in $(\lfloor P \rfloor, P)$.

Thus lifting adds transitions (saturation) which correspond to observe the effects of all the arrows in the index category. Thus bisimilar states must behave correspondingly for all such substitutions.

History-Dependent Automata

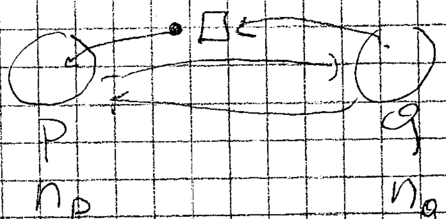
$$\frac{p \xrightarrow{\overline{x}y} q}{(y)\,p \xrightarrow{\overline{x}(z)} q[z/y]}$$

infinite branching
chose one for finite state.
different than all the others.

$p \xrightarrow{\overline{x}(y)}$
$q \xrightarrow{\overline{x}(y)}$ $\Rightarrow$ choose the same z!

OK on the fly, doesnot work for minimization.

HD automata   local names



injection $n_q \to n_p \cup \{*\}$
new name

Bisimulation: set of states with partial bijection
Group of symmetries: bisimilar to itself
with a p. bijection different from id.

Coalgebraic theory based on Named Sets

$$H = \langle Q_N, S_N \rangle$$

set $: Q_N \to \mathrm{Symset}$ ← set of groups

Named sets are compact representations of orbits.

Coalgebra category in named set $\cong$ in Alg^{π}
but smaller objects