Higher-order encoding of the π-calculus in (Co)Inductive Type Theories

Furio Honsell, Marino Miculan, Ivan Scagnetto

Università di Udine

Introduction

This work is part of an ongoing research at the Computer Science Department of the University of Udine in the area of the Computer Aided Formal Reasoning about programs and concurrent systems.

Motivations: the application of formal systems to the analysis of programs and concurrent systems is difficult and error-prone, due to their complexity.

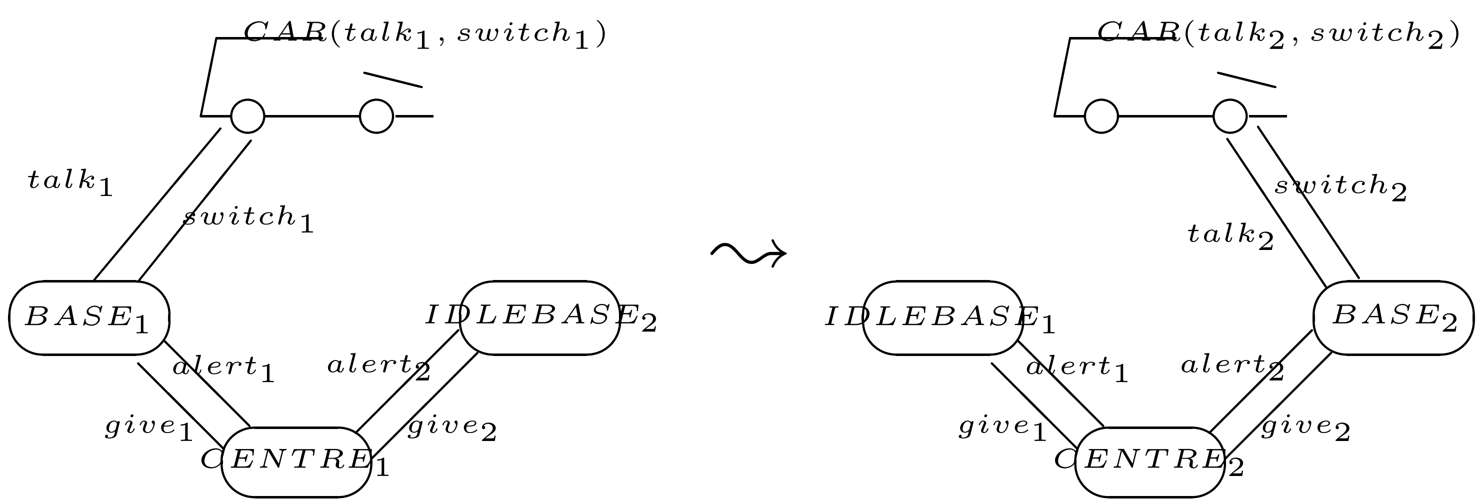
Aim: developing *proof editors*, aiding the user in the development of error-free formal proofs.

In this talk: we will report about our work in the encoding of a popular and widespread process algebra.

The π-calculus

[MPW92]: "…A calculus of communicating systems in which one can naturally express processes which have changing structure"

Mobility of channels: channels are denoted by *names*, which may be exchanged by processes along themselves.



The π-calculus

Three components:

syntax of *names* ($\mathcal{N}$), *actions* and *agents* (*processes*, $\mathcal{P}$);

operational semantics i.e., labelled transition relation:
$\xrightarrow{\alpha} \subseteq \mathcal{P} \times \mathcal{P}$;

equivalence relation between processes: $\dot{\sim} \subseteq \mathcal{P} \times \mathcal{P}$.

Syntax of the π-calculus

Processes

$$\begin{aligned} P ::= \quad & 0 \mid \bar{x}y.P \mid x(y).P \mid \tau.P \mid (\nu x)P \mid !P \\ & \mid P_1|P_2 \mid P_1 + P_2 \mid [x = y]P \mid [x \neq y]P \end{aligned}$$

Actions

α	Kind	$fn(\alpha)$	$bn(\alpha)$
τ	Free	$\emptyset$	$\emptyset$
$\overline{x}y$	Free	$\{x, y\}$	$\emptyset$
$x(y)$	Bound	$\{x\}$	$\{y\}$
$\overline{x}(y)$	Bound	$\{x\}$	$\{y\}$

Operational Semantics of the π-calculus

$$\text{OUT}\ \frac{-}{\overline{x}y.P \xrightarrow{\overline{x}y} P} \qquad \text{IN}\ \frac{-}{x(z).P \xrightarrow{x(w)} P\{w/z\}}\ w \notin fn((\nu z)P)$$

$$\text{SUM}_\text{l}\ \frac{P \xrightarrow{\alpha} P'}{P+Q \xrightarrow{\alpha} P'} \qquad \text{PAR}_\text{l}\ \frac{P \xrightarrow{\alpha} P'}{P|Q \xrightarrow{\alpha} P'|Q}\ bn(\alpha) \cap fn(Q) = \emptyset$$

$$\text{COM}_\text{l}\ \frac{P \xrightarrow{\overline{x}y} P' \quad Q \xrightarrow{x(z)} Q'}{P|Q \xrightarrow{\tau} P'|Q'\{y/z\}} \qquad \text{RES}\ \frac{P \xrightarrow{\alpha} P'}{(y)P \xrightarrow{\alpha} (y)P'}\ y \notin n(\alpha)$$

$$\text{MATCH}\ \frac{P \xrightarrow{\alpha} P'}{[x=x]P \xrightarrow{\alpha} P'} \qquad \text{MISMATCH}\ \frac{P \xrightarrow{\alpha} P'}{[x \neq y]P \xrightarrow{\alpha} P'}\ x \neq y$$

$$\text{OPEN}\ \frac{P \xrightarrow{\overline{x}y} P'}{(\nu y)P \xrightarrow{\overline{x}(w)} P'\{w/y\}}\ \begin{array}{l} y \neq x \\ w \notin fn((\nu y)P') \end{array} \qquad \text{TAU}\ \frac{-}{\tau.P \xrightarrow{\tau} P}$$

$$\text{CLOSE}_\text{l}\ \frac{P \xrightarrow{\overline{x}(w)} P' \quad Q \xrightarrow{x(w)} Q'}{P|Q \xrightarrow{\tau} (\nu w)(P'|Q')} \qquad \text{REPL}\ \frac{P \xrightarrow{\alpha} P'}{!P \xrightarrow{\alpha} P'|!P}$$

Strong (late) bisimilarity

Definition: A binary relation $\mathcal{S}$ on processes is a *strong simulation* iff, for all P, Q processes, if $P\ \mathcal{S}\ Q$ then

1. if $P \xrightarrow{\alpha} P'$ and α is a free action, then $\exists Q'.Q \xrightarrow{\alpha} Q'$ and $P'\ \mathcal{S}\ Q'$;
2. if $P \xrightarrow{x(y)} P'$ and $y \notin n(P,Q)$, then $\exists Q'.Q \xrightarrow{x(y)} Q'$ and for all $w \in \mathcal{X}$: $P'\{w/y\}\ \mathcal{S}\ Q'\{w/y\}$;
3. if $P \xrightarrow{\overline{x}(y)} P'$ and $y \notin n(P,Q)$, then $\exists Q'.Q \xrightarrow{\overline{x}(y)} Q'$ and $P'\ \mathcal{S}\ Q'$.

$\mathcal{S}$ is a *strong bisimulation* if both $\mathcal{S}$ and $\mathcal{S}^{-1}$ are strong simulations.

The *strong bisimilarity* is the binary relation $\dot{\sim}$ defined as

$$P \dot{\sim} Q \iff \exists \mathcal{S}.(P\ \mathcal{S}\ Q)$$

Encoding the π-calculus

In formalizing the π-calculus within some Logical Framework, we face three levels of encoding:

1. the theory of the π-calculus: sintax, operational semantics;
2. the theory of strong (late) bisimilarity ($\dot\sim$);
3. the metatheory: properties about transitions, $\dot\sim$ as an equivalence, algebraic laws for $\dot\sim$,...

Aim: encoding faithfully these components in $\mathrm{CC}^{(Co)Ind}$ by taking full advantage of the HOAS approach.

Encoding the Theory of the π-calculus

Syntax

Names are represented by variables of type `name`,
processes are terms of type `proc`

$\mathcal{N}, \mathcal{P}, \mathcal{L}$	$\leadsto$	`name,proc,label:Set`
0	$\leadsto$	`0 : proc`
$+, \mid$	$\leadsto$	`sum,par :  proc -> proc -> proc`
ν	$\leadsto$	`nu :  (name -> proc) -> proc`
$\_(\_)$	$\leadsto$	`in_pref :  name -> (name -> proc) -> proc`

$x(y).P \leadsto$ `(in_pref x [y:name]`$\hat{P}$`) :  proc`

Note: `proc, label` are inductive, `name` is not.

Encoding the Theory of the π-calculus
Operational semantics

Two mutually defined inductive predicates:

```
Inductive trans : proc -> label -> proc -> Prop := ...
with    btrans : proc -> (name -> label)
                      -> (name -> proc) -> Prop := ...
```

and two auxilary predicates implementing *freshness*:

```
Inductive     notin [x:name] : proc  -> Prop := ...
Inductive lab_notin [x:name] : label -> Prop := ...
```

Encoding the Theory of the π-calculus

$$\text{IN}\ \frac{-}{x(z).P \xrightarrow{x(w)} P\{w/z\}}\ w \notin fn((\nu z)P)$$

```
IN : (p:name->proc)(x:name)
     (btrans (in_pref x p) [w:name](In x w) p)
```

$$\text{COM}_\text{l}\ \frac{P \xrightarrow{\overline{x}y} P' \quad Q \xrightarrow{x(z)} Q'}{P|Q \xrightarrow{\tau} P'|Q'\{y/z\}}$$

```
COM1: (p1,p2,q2:proc)(q1:name -> proc)(x,y:name)
       (btrans p1 [z:name](In x z) q1)
             -> (trans p2 (Out x y) q2)
      -> (trans (par p1 p2) tau (par (q1 y) q2))
```

The side condition are automatically dealt with by the HOAS.

Adequacy of the encoding

Proposition 1 *There is a compositional bijection $\varepsilon^{\mathcal{P}}_{\mathcal{X}'}$ between the processes P with $fn(P) \subseteq \mathcal{X}'$ and the canonical forms* $\mathtt{t}$ *such that* $\Gamma_{\mathcal{X}'} \vdash_{\Sigma} \mathtt{t} : \mathtt{proc}$.

Proposition 2 *There is a compositional bijection $\varepsilon^{LTS}_{\mathcal{X}'}$ between the proof trees $\Pi : P_1 \xrightarrow{\alpha} P_2$ with $fn(\Pi) \subseteq \mathcal{X}'$ and the canonical forms* $\mathtt{t}$ *such that the following holds:*

1. $P_1 \xrightarrow{\tau} P_2$ *iff* $\Gamma_{\mathcal{X}'} \vdash_{\Sigma} \mathtt{t} : (\mathtt{trans}\ \varepsilon^{\mathcal{P}}_{\mathcal{X}'}(P_1)\ \mathtt{tau}\ \varepsilon^{\mathcal{P}}_{\mathcal{X}'}(P_2))$;
2. $P_1 \xrightarrow{\bar{x}y} P_2$ *iff* $\Gamma_{\mathcal{X}'} \vdash_{\Sigma} \mathtt{t} : (\mathtt{trans}\ \varepsilon^{\mathcal{P}}_{\mathcal{X}'}(P_1)\ (\mathtt{Out\ x\ y})\ \varepsilon^{\mathcal{P}}_{\mathcal{X}'}(P_2))$;
3. $P_1 \xrightarrow{x(y)} P_2$ *iff* $\Gamma_{\mathcal{X}'} \vdash_{\Sigma} \mathtt{t} : (\mathtt{btrans}\ \varepsilon^{\mathcal{P}}_{\mathcal{X}'}(P_1)[\mathtt{y} : \mathtt{name}](\mathtt{In\ x\ y})$
$$[\mathtt{y} : \mathtt{name}]\varepsilon^{\mathcal{P}}_{\mathcal{X}' \cup \{y\}}(P_2))$$
4. $P_1 \xrightarrow{\bar{x}(y)} P_2$ *iff* $\Gamma_{\mathcal{X}'} \vdash_{\Sigma} \mathtt{t} : (\mathtt{btrans}\ \varepsilon^{\mathcal{P}}_{\mathcal{X}'}(P_1)[\mathtt{y} : \mathtt{name}](\mathtt{Out\ x\ y})$
$$[\mathtt{y} : \mathtt{name}]\varepsilon^{\mathcal{P}}_{\mathcal{X}' \cup \{y\}}(P_2))$$

Encoding the Theory of $\dot{\sim}$
Inductive Approach

The straightforward approach is to directly represent the theory underlying the definition of $\dot{\sim}$, i.e., part of the theory of greatest fixpoint operators:

```
Inductive StBisim' : proc -> proc -> Prop =
      Co_Ind : (R:proc->proc->Prop)
       (Inclus R (Op_StBisim R)) ->
       (p1,p2:proc)(R p1 p2) -> (StBisim' p1 p2).
```

But there is a better solution...

Encoding the Theory of $\dot\sim$

Coinductive Approach

In $CC^{(Co)Ind}$ we can take full advantage of `CoInductive` types: $\dot\sim$ can be defined as a single "circular" constructor-guarded predicate.

$\dot\sim \;\rightsquigarrow$
```
CoInductive StBisim :  proc -> proc -> Prop = ...
    sb : (p,q:proc)(...)/\(...)/\(...) -> (StBisim p q).
```

Advantage: proofs of $\dot\sim$ are greatly simplified, due to the support offered by `Coq` to "circular" proofs.

Internal adequacy: the two approaches are provably equal in `Coq`

```
Lemma Adequacy : (p1,p2:proc)
                 (StBisim p1 p2) <-> (StBisim' p1 p2).
```

Hence, we can switch between the two approaches whenever it is needed.

Encoding the MetaTheory of the π-calculus

We aim to build a *workbench* providing a set of *tools* for reasoning about processes of π-calculus.

There are many useful facts which can be proved once and for all the processes: *metatheoretic* properties. E.g.: algebraic laws.

```
Variables p,q:proc.
Lemma SYM   : (StBisim p q) -> (StBisim q p).
Lemma TRANS : (StBisim p q)->(StBisim q r)->(StBisim p r).
Variables p',q':name->proc.
Lemma NU_S  : ((z:name)
                 (notin z (nu p'))->(notin z (nu q'))->
                 (StBisim (p' z) (q' z)))
            -> (StBisim (nu p') (nu q')).
```

This is a work in progress.

Encoding the MetaTheory of the π-calculus

In proving these general properties, we may need to deal with syntactic features which are hidden by HOAS: substitution, α-conversion, freshness. . .

Example of "difficult" properties, needed for proving `TRANS`:

Lemma 1 If $P \xrightarrow{\alpha} P'$ then $fn(\alpha) \subseteq fn(P)$ and $fn(P') \subseteq fn(P) \cup bn(\alpha)$.

Lemma 3 If $P \xrightarrow{\alpha} P'$, $bn(\alpha) \cap fn(P'\{x/y\}) = \emptyset$ and $y \notin bn(\alpha)$, then $P\{x/y\} \xrightarrow{\alpha\{x/y\}} P'\{x/y\}$.

Lemma 4 If $P\{x/y\} \xrightarrow{\alpha} P'$, $x \notin fn(P)$, $bn(\alpha) \cap fn(P', x) = \emptyset$, then there exist Q, β such that $Q\{x/y\} = P'$ and $\beta\{x/y\} = \alpha$ and $P \xrightarrow{\beta} Q$.

Lemma 6 If $P \dot{\sim} Q$ and $w \notin fn(P, Q)$, then $P\{w/x\} \dot{\sim} Q\{w/x\}$.

Proof of Lemma 6

It is proved by defining a bisimulation and by internal adequacy:

```
Inductive BL6 : proc->proc->Prop :=
  bl6: (p,q:proc)(n:nat)((Bfun n) p q)->(BL6 p q).
Lemma BisimL6: (Inclus BL6 (Op_StBisim BL6)).
Lemma Lemma6': (p,q:name->proc)(z:name)
         (notin z (nu p)) -> (notin z (nu q)) ->
         (StBisim' (p z) (q z)) ->
 (w:name)~(w=z)->(notin w (nu p)) -> (notin w (nu q)) ->
         (StBisim' (p w) (q w)).
Lemma Lemma6: (p,q:name->proc)(z:name)
         (notin z (nu p)) -> (notin z (nu q)) ->
         (StBisim (p z) (q z)) ->
 (w:name)~(w=z)->(notin w (nu p)) -> (notin w (nu q)) ->
         (StBisim (p w) (q w)).
```

Axiomatizing HOAS internals

We need to explicate some syntactic properties about process contexts, by adding some *language-independent* postulates about HOAS behaviour.

$$
\begin{array}{rcl}
\textsc{Unsaturation} & : & \forall P \exists x. x \notin fn(P) \\
\textsc{Expansion} & : & \forall P, x \exists Q(\cdot). P = Q(x) \wedge x \notin Q(\cdot) \\
\textsc{ExpansionHO} & : & \forall P(\cdot), x \exists Q(\cdot,\cdot). P(\cdot) = Q(x,\cdot) \wedge x \notin Q(\cdot,\cdot) \\
\textsc{EqCongr} & : & \dfrac{P(x) = Q(x)}{P(y) = Q(y)} x, y \notin P(\cdot), Q(\cdot)
\end{array}
$$

Encoding the MetaTheory of the π-calculus

We have formally proved some of Milner's results about $\dot{\sim}$:

- symmetry, reflexivity and transitivity of $\dot{\sim}$;
- the algebraic laws of *summation*;
- the algebraic laws regarding *match* and *mismatch* operators.

To do next:

- other algebraic laws, congruence properties, ...
- Milner's Lemmata 1,3,4 [MPW92] (Actually, these properties have been postulated)

Conclusions I: the good news...

We are investigating HOAS-based encodings of the π-calculus in type-theory based Logical Framework (namely, $\mathrm{CC}^{(Co)Ind}$).

- ♡ the theory of the π-calculus (sintax, operational semantics) is successfully encoded by means of HOAS.
- ♡ the theory of strong (late) bisimilarity is easily encoded by taking advantage of `CoInductive` types;
- ♡ most algebraic laws (metaproperties) are easily proved, thansk to the HOAS encoding.

Conclusions II: . . . and the bad ones

♠ extra axioms may be needed in order to prove metaproperties regarding syntactic features, such as substitutions.

♠ In proving properties, these situations may easily arise:

– coinductive calls guarded by axioms (e.g., TRANS)

– nested application of coinductive hypothesis (e.g., Lemma 6)

but they are rejected by Coq guardedness checking, leading to innatural arguments:

Open Problems and Future work

- What is the *rationale* of the added axioms about HOAS?
  Is there a general underlying theory?
- Can the axioms be eliminated if we introduce a suitable (higher-order) induction principle over process contexts?
- Investigating the applicability of this approach to the *polyadic* π-calculus (already in progress)

Related work

Other relevant works about the encoding of π-calculus in LFs:

- Melham [Mel94]: a first order encoding with explicit substitutions. No use of HOAS.
- Hirschkoff [Hir96]: based on de Bruijn indexes (no variables at all), and Sangiorgi's theory of progressions.

References

[Coq96] *The Coq Proof Assistant Reference Manual - Version 6.1.* INRIA, Rocquencourt, Dec. 96.

[CH88] Thierry Coquand and Gérard Huet. The calculus of constructions. *Information and Control*, 76:95–120, 1988.

[Chu40] Alonzo Church. A formulation of the simple theory of types. *Journal of Symbolic Logic*, 5:56–68, 1940.

[Gim95] Eduardo Giménez. Codifying guarded recursion definitions with recursive schemes. In *Proc. of TYPES'94*, LNCS 996, June 1995. Springer-Verlag.

[HHP93] Robert Harper, Furio Honsell, and Gordon Plotkin. A framework for defining logics. *J.ACM*, 40(1):143–184, Jan 1993.

[Hir96] Daniel Hirschkoff. Bisimulation proofs for the π-calculus in the Calculus of Constructions. Technical Report 96-62, CERMICS, April 1996. http://cermics.enpc.fr.

[Mel94] Thomas F. Melham. A mechanized theory of the π-calculus in HOL. *Nordic J. Comput.*, 1(1):50–76, 1994.

[MPW92] Robin Milner, Joachim Parrow, and David Walker. A calculus of mobile processes. *Inform. and Comput.*, 100(1):1–77, 1992.

[NPS92] Bengt Nordström, Kent Petersson, and Jan M. Smith. Martin-Löf's type theory. In *Handbook of Logic in Computer Science*. OUP, 1992.

[Pau93] Christine Paulin-Mohring. Inductive definitions in the system Coq; rules and properties. In *Proc. of TLCA 93*, LNCS 664, pages 328–345. Springer-Verlag, 1993.