

Formalisation en Coq du calcul de processus HOcore

Simon Boulier¹ Alan Schmitt²

JFLA 2012

-
1. ENS Cachan - Antenne de Bretagne
 2. INRIA

Nous montrons que \mathcal{R} est une bisimulation tardive d'ordre supérieure. La relation est symétrique, il suffit donc de montrer qu'il s'agit d'une simulation. Nous procédons par analyse de cas sur les transitions de $P\{R/X\}$.

Si la transition est engendrée uniquement par P , nous avons $P\{R/X\} \xrightarrow{\alpha} A\{R/X\}$ avec $P \xrightarrow{\alpha} A$. Nous avons donc $P\{\bar{m}.\bar{n}.\mathbf{0}/X\} \xrightarrow{\alpha} A\{\bar{m}.\bar{n}.\mathbf{0}/X\}$. La transition $\xrightarrow{\alpha}$ provient de P , qui ne contient pas d'occurrences de m, n , donc m, n n'apparaissent pas dans α . Nous distinguons les trois cas possibles pour A .

- Supposons que A est un processus P' . Comme $P\{\bar{m}.\bar{n}.\mathbf{0}/X\} \sim_l Q\{\bar{m}.\bar{n}.\mathbf{0}/X\}$, il existe Q' tel que $Q\{\bar{m}.\bar{n}.\mathbf{0}/X\} \xrightarrow{\alpha} Q'$ et $P'\{\bar{m}.\bar{n}.\mathbf{0}/X\} \sim_l Q'$. Comme $m, n \notin \alpha$, la transition $Q\{\bar{m}.\bar{n}.\mathbf{0}/X\} \xrightarrow{\alpha} Q'$ provient de Q : il existe Q'' tel que $Q \xrightarrow{\alpha} Q''$ et $Q' = Q''\{\bar{m}.\bar{n}.\mathbf{0}/X\}$. Nous avons donc $Q\{R/X\} \xrightarrow{\alpha} Q''\{R/X\}$ avec $P'\{R/X\} \mathcal{R} Q''\{R/X\}$, comme voulu.
- Supposons que A est une fonction F . Il existe F' telle que $Q\{\bar{m}.\bar{n}.\mathbf{0}/X\} \xrightarrow{\alpha} F'$ et $(F\{\bar{m}.\bar{n}.\mathbf{0}/X\}) \circ T \sim_l F' \circ T$ pour tout processus T . Comme $m, n \notin \alpha$, la transition $Q\{\bar{m}.\bar{n}.\mathbf{0}/X\} \xrightarrow{\alpha} F'$ provient de Q : il existe F'' tel que $Q \xrightarrow{\alpha} F''$ et $F' = F''\{\bar{m}.\bar{n}.\mathbf{0}/X\}$. Nous avons donc $Q\{R/X\} \xrightarrow{\alpha} F''\{R/X\}$, et pour tout T clos, nous avons $(F\{R/X\}) \circ T = (F \circ T)\{R/X\}$ et $(F''\{R/X\}) \circ T = (F'' \circ T)\{R/X\}$. Nous avons également $(F \circ T)\{\bar{m}.\bar{n}.\mathbf{0}/X\} = (F\{\bar{m}.\bar{n}.\mathbf{0}/X\}) \circ T \sim_l (F''\{\bar{m}.\bar{n}.\mathbf{0}/X\}) \circ T = (F'' \circ T)\{\bar{m}.\bar{n}.\mathbf{0}/X\}$. Finalement nous avons donc $(F \circ T)\{R/X\} \mathcal{R} (F'' \circ T)\{R/X\}$ comme voulu.
- Supposons que $A = \langle S \rangle T$. Il existe S' et T' tels que $Q\{\bar{m}.\bar{n}.\mathbf{0}/X\} \xrightarrow{\alpha} \langle S' \rangle T'$, $S\{\bar{m}.\bar{n}.\mathbf{0}/X\} \sim_l S'$ et $T\{\bar{m}.\bar{n}.\mathbf{0}/X\} \sim_l T'$. Comme nous avons $m, n \notin \alpha$, la transition $Q\{\bar{m}.\bar{n}.\mathbf{0}/X\} \xrightarrow{\alpha} \langle S' \rangle T'$ provient de Q : il existe S'' , T'' tels que $Q \xrightarrow{\alpha} \langle S'' \rangle T''$, $S' = S''\{\bar{m}.\bar{n}.\mathbf{0}/X\}$ et $T' = T''\{\bar{m}.\bar{n}.\mathbf{0}/X\}$. Par conséquent, nous avons $Q\{R/X\} \xrightarrow{\alpha} \langle S'' \rangle T''\{R/X\}$ avec $S\{R/X\} \mathcal{R} S''\{R/X\}$ et $T\{R/X\} \mathcal{R} T''\{R/X\}$, comme voulu.

On the Expressiveness and Decidability of Higher-Order Process Calculi*[†]

Ivan Lanese

Focus Team, Università di Bologna/INRIA, Italy

`lanese@cs.unibo.it`

Jorge A. Pérez

CITI - Dept. of Computer Science, New University of Lisbon, Portugal

`jorge.perez@di.fct.unl.pt`

Davide Sangiorgi

Focus Team, Università di Bologna/INRIA, Italy

`davide.sangiorgi@cs.unibo.it`

Alan Schmitt

INRIA, France

`alan.schmitt@inria.fr`

June 1, 2010

Abstract

In *higher-order process calculi*, the values exchanged in communications may contain processes. A core calculus of higher-order concurrency is studied; it has

Syntaxe d'un processus

HOcore :

$P, P' ::=$

| $\bar{a}\langle P \rangle$

| $a(x).P$

| $P \parallel P'$

| x

| **0.**

Syntaxe d'un processus

HOcore :

$P, P' ::=$

| $\bar{a}\langle P \rangle$

| $a(x).P$

| $P \parallel P'$

| x

| $0.$

Definition chan := nat.

Definition lvar := nat.

Inductive process : Set :=

| Send : chan → process → process

| Receive : chan → lvar → process
→ process

| Par : process → process →
process

| Lvar : lvar → process

| Nil : process.

Communication

Communication :

$$\bar{a}\langle Q \rangle \parallel a(x).P \longrightarrow [Q / x]P$$

Réduction :

$$\dots \parallel \bar{a}\langle Q \rangle \parallel \dots \parallel a(x).P \parallel \dots \longrightarrow \dots \parallel [Q / x]P \parallel \dots$$

Congruence Structurale

Congruence Structurale :

- ▶ $P \parallel (Q \parallel R) \equiv (P \parallel Q) \parallel R$
- ▶ $P \parallel Q \equiv Q \parallel P$
- ▶ $P \parallel \mathbf{0} \equiv P$

Exemple :

$$\bar{a}\langle x \parallel y \rangle \equiv \bar{a}\langle y \parallel x \rangle$$

Réduction :

$$\frac{P \equiv P' \quad P' \longrightarrow Q' \quad Q \equiv Q'}{P \longrightarrow Q}$$

$$\frac{P \longrightarrow P'}{(P \parallel Q) \longrightarrow (P' \parallel Q)}$$

Résumé

 $P, P' ::=$ $| \bar{a}\langle P \rangle$ $| a(x).P$ $| P \parallel P'$ $| x$ $| \mathbf{0}.$ $\bar{a}\langle Q \rangle \parallel a(x).P \longrightarrow [Q / x]P$
$$\frac{P \longrightarrow P'}{(P \parallel Q) \longrightarrow (P' \parallel Q)}$$
$$\frac{P \equiv P' \quad P' \longrightarrow Q' \quad Q \equiv Q'}{P \longrightarrow Q}$$

- ▶ $P \parallel (Q \parallel R) \equiv (P \parallel Q) \parallel R$
- ▶ $P \parallel Q \equiv Q \parallel P$
- ▶ $P \parallel \mathbf{0} \equiv P$

HOcore est Turing complet !

Une nouvelle sémantique

Les règles du LTS :

Une nouvelle sémantique

Les règles du LTS :

$$\text{OUT } \bar{a}\langle P \rangle \xrightarrow{\bar{a}\langle P \rangle} \mathbf{0}$$

Une nouvelle sémantique

Les règles du LTS :

OUT $\bar{a}\langle P \rangle \xrightarrow{\bar{a}\langle P \rangle} \mathbf{0}$

INP $a(x).P \xrightarrow{a} (x).P$

Fonction :

$F ::=$

| $(x).P$

Une nouvelle sémantique

Les règles du LTS :

OUT $\bar{a}\langle P \rangle \xrightarrow{\bar{a}\langle P \rangle} \mathbf{0}$

INP $a(x).P \xrightarrow{a} (x).P$

ACT si $P \xrightarrow{\alpha} A$ alors $P \parallel Q \xrightarrow{\alpha} A \parallel Q$

Fonction :

$F ::=$

| $(x).P$

| $F \parallel P$

| $P \parallel F.$

$A ::= P \mid F.$

Une nouvelle sémantique

Les règles du LTS :

OUT $\bar{a}\langle P \rangle \xrightarrow{\bar{a}\langle P \rangle} \mathbf{0}$

INP $a(x).P \xrightarrow{a} (x).P$

ACT si $P \xrightarrow{\alpha} A$ alors $P \parallel Q \xrightarrow{\alpha} A \parallel Q$

TAU si $P \xrightarrow{\bar{a}\langle P'' \rangle} P'$ et $Q \xrightarrow{a} F$ alors $P \parallel Q \xrightarrow{\tau} P' \parallel F \bullet P''$

Fonction :

$F ::=$

| $(x).P$

| $F \parallel P$

| $P \parallel F.$

Instanciation :

$(x).P \bullet P'' = [P'' / x]P$

$(F \parallel P) \bullet P'' = (F \bullet P'') \parallel P$

$(P \parallel F) \bullet P'' = P \parallel (F \bullet P'')$

$A ::= P \mid F.$

Le problème de la capture

$$a(y).x \sim_{\alpha} a(z).x$$

$$[y / x](a(y).x) = a(y).y \not\sim_{\alpha} [y / x](a(z).x) = a(z).y$$

Une Solution : la Représentation Canonique des Lieurs

- ▶ pour cela, on distingue variables libres et liées

libres : X, Y, Z (gvar : globales)

liées : x, y, z (lvar : locales)

$P, P' ::=$

| $\bar{a}\langle P \rangle$

| $a(x).P$

| $P \parallel P'$

| x

| X

| $\mathbf{0}$.

Une Solution : la Représentation Canonique des Lieurs

- ▶ pour cela, on distingue variables libres et liées

libres : X, Y, Z (gvar : globales)

liées : x, y, z (lvar : locales)

$P, P' ::=$

| $\bar{a}\langle P \rangle$

| $a(x).P$

| $P \parallel P'$

| x

| X

| $\mathbf{0}$.

- ▶ on veut choisir un représentant

Exemple

$a(x).(x \parallel x)$ ou $a(y).(y \parallel y)$?

Exemple

$$a(x).(x \parallel x) \text{ ou } a(y).(y \parallel y) ?$$

1. on considère $X \parallel X$
2. on calcule $f_x(X \parallel X) \rightarrow x_0$
3. le représentant canonique est :
 $a(x_0).([x_0 / X](X \parallel X)) = a(x_0).(x_0 \parallel x_0)$

On note $\mathbf{abs}_f a X P = a(x_0).([x_0 / X]P)$ où $x_0 = f_x(P)$.

Conclusion : $a(x).(x \parallel x) \rightsquigarrow \mathbf{abs}_f a X (X \parallel X)$.

On note $\mathbf{abs}_f a X P = a(x_0).([x_0 / X]P)$ où $x_0 = f_x(P)$.

$f : \text{gvar} \rightarrow \text{process} \rightarrow \text{lvar}$ est une fonction de hauteur, elle vérifie :

Équivariance $f_X(P) = f_{\pi(X)}(\pi(P))$ pour toute permutation π
 $\Rightarrow \mathbf{abs}_f a X (X \parallel X) = \mathbf{abs}_f a Y (Y \parallel Y)$

Fraîcheur ex : $\mathbf{abs}_f b X (a(x).X) \begin{cases} \neq b(x).(a(x).x) \\ = b(y).(a(x).y) \end{cases}$

Substitution si $X \# Q, Q'$ alors $f_X(\dots Q \dots) = f_X(\dots Q' \dots)$

Congruence si $P \equiv Q$ alors $f_X(P) = f_X(Q)$

```

Fixpoint GoodF X P :=
  match P with
  | Gvar Y => if beq_nat X Y then 1 else 0
  | Lvar _ => 0
  | Nil => 0
  | Par P1 P2 => max (GoodF X P1) (GoodF X P2)
  | Send a P => GoodF X P
  | Receive a x P => let m' := GoodF X P in
    if beq_nat m' 0 then 0
    else if lt_dec x m' then m'
    else S x
  end.

```

Theorem GoodF_is_good : good GoodF.

- ▶ $\forall X, \text{wf}_f(X)$
- ▶ si $\text{wf}_f(P)$ alors $\text{wf}_f(\bar{a}\langle P \rangle)$
- ▶ si $\text{wf}_f(P)$ et $\text{wf}_f(Q)$ alors $\text{wf}_f(P \parallel Q)$
- ▶ $\text{wf}_f(\mathbf{0})$
- ▶ si $\text{wf}_f(P)$, alors $\text{wf}_f(\mathbf{abs}_f a X P)$

Inductive wf (f:height_fun) : process \rightarrow Prop :=
| WfGvar : $\forall X, \text{wf } f \text{ (Gvar } X)$
| WfSend : $\forall a \text{ } p, (\text{wf } f \text{ } p) \rightarrow \text{wf } f \text{ (Send } a \text{ } p)$
| WfPar : $\forall p \text{ } q, (\text{wf } f \text{ } p) \rightarrow (\text{wf } f \text{ } q) \rightarrow \text{wf } f \text{ (Par } p \text{ } q)$
| WfNil : wf f Nil
| WfReceive : $\forall a \text{ } X \text{ } p, (\text{wf } f \text{ } p) \rightarrow \text{wf } f \text{ (abs } f \text{ } a \text{ } X \text{ } p)$.

Lemma `wf_ind` : $\forall (P : \text{process} \rightarrow \text{Prop}),$
 $(\forall a p, \text{wf } f \text{ p} \rightarrow P \text{ p} \rightarrow P (\text{Send } a \text{ p})) \rightarrow$
 $(\forall a X p, \text{wf } f \text{ p} \rightarrow$
 $\quad P \text{ p} \rightarrow$
 $\quad\quad P (\text{abs } f \text{ a } X \text{ p})) \rightarrow$
 $(\forall X, P (\text{Gvar } X)) \rightarrow$
 $(\forall p q, \text{wf } f \text{ p} \rightarrow P \text{ p} \rightarrow \text{wf } f \text{ q} \rightarrow P \text{ q} \rightarrow P (\text{Par } p \text{ q})) \rightarrow$
 $(P \text{ Nil}) \rightarrow$
 $\quad \forall p, \text{wf } f \text{ p} \rightarrow P \text{ p}.$

Lemma `wf_ind_gen` : $\forall (P : \text{process} \rightarrow \text{Prop}),$

$(\forall a p, \text{wf } f \text{ } p \rightarrow P \text{ } p \rightarrow P (\text{Send } a \text{ } p)) \rightarrow$

$(\forall a X p, \text{wf } f \text{ } p \rightarrow$

$(\forall Y, Y \# p \rightarrow P ([Y/X]p)) \rightarrow$

$P (\text{abs } f \text{ } a \text{ } X \text{ } p)) \rightarrow$

$(\forall X, P (\text{Gvar } X)) \rightarrow$

$(\forall p q, \text{wf } f \text{ } p \rightarrow P \text{ } p \rightarrow \text{wf } f \text{ } q \rightarrow P \text{ } q \rightarrow P (\text{Par } p \text{ } q)) \rightarrow$

$(P \text{ Nil}) \rightarrow$

$\forall p, \text{wf } f \text{ } p \rightarrow P \text{ } p.$

Équivalence de processus

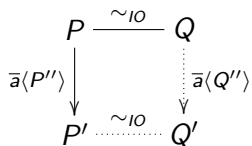
La Congruence Barbu

$$\begin{array}{ccc} P & \simeq & Q \\ \bar{a} \downarrow & & \downarrow \bar{a} \\ P' & & Q' \end{array}$$

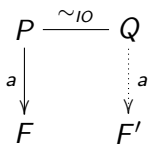
$$\begin{array}{ccc} P & \simeq & Q \\ \tau \downarrow & & \downarrow \tau \\ P' & \dots \simeq \dots & Q' \end{array}$$

$P \simeq Q \Rightarrow C[P] \simeq C[Q]$
pour tout contexte C

L'IO-bissimulation

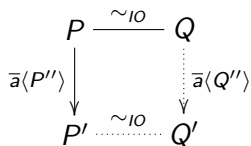


$$P'' \sim_{IO} Q''$$

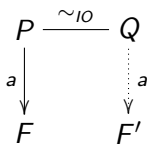


$$F \bullet M \sim_{IO} F' \bullet M \quad \forall M$$

L'IO-bissimulation



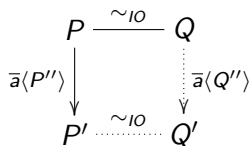
$$P'' \sim_{IO} Q''$$



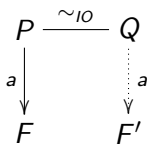
$$F \bullet M \sim_{IO} F' \bullet M \quad \forall M$$

$$F \bullet Y \sim_{IO} F' \bullet Y \quad \text{avec } Y \# P, Q$$

L'IO-bissimulation

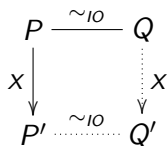


$$P'' \sim_{IO} Q''$$



$$F \bullet M \sim_{IO} F' \bullet M \quad \forall M$$

$$F \bullet Y \sim_{IO} F' \bullet Y \quad \text{avec } Y \# P, Q$$



REM $X \xrightarrow{X} \mathbf{0}$

Lemma bio_ok : $\forall p q,$
wf f p \rightarrow wf f q \rightarrow (p \sim q \leftrightarrow bio p q = true).

Théorème : $P \sim_{10} Q \iff P \simeq Q$

Théorème : $P \sim_{IO} Q \iff P \simeq Q$

Donc \simeq n'est pas si compliquée !

- ▶ décidable
- ▶ axiomatisable :

$$\prod_{k+1} a(x).P \simeq a(x).(P \parallel \prod_k a(x).P)$$

Theorem IObis_correct : $\forall p q,$
wf f p \rightarrow wf f q \rightarrow IObis p q \rightarrow equiv p q.

Conclusion