HOπ in Coq

Guillaume Ambal, Sergueï Lenglet and Alan Schmitt

Higher-Order π-calculus

- Model of concurrent and communicating systems
  - First-order: inert data (channel names, ...)
  - Higher-order: executable processes
- Behavioral equivalence proofs (bisimulation): complex, prone to error
- Very few formalization of higher-order process calculi
- Difficulty: binders

Higher-Order π-calculus

Communication channel names $a, b, c, \ldots$
Process variables $X, Y, Z, \ldots$

$$P, Q ::=$$

Higher-Order π-calculus

Communication channel names $a, b, c, \ldots$
Process variables $X, Y, Z, \ldots$

$$P, Q ::= \varnothing \qquad \text{nil process}$$

Higher-Order π-calculus

Communication channel names $a, b, c, \ldots$
Process variables $X, Y, Z, \ldots$

$$
\begin{aligned}
P, Q ::= & \ \varnothing & \text{nil process} \\
| & \ P \parallel Q & \text{parallel composition}
\end{aligned}
$$

Higher-Order π-calculus

Communication channel names $a, b, c, \ldots$
Process variables $X, Y, Z, \ldots$

$$
\begin{aligned}
P, Q ::= & \ \emptyset && \text{nil process} \\
& \mid P \parallel Q && \text{parallel composition} \\
& \mid X && \text{variable} \\
& \mid a(X).P && \text{process input}
\end{aligned}
$$

$$a(X).X$$

Higher-Order π-calculus

Communication channel names $a, b, c, \ldots$
Process variables $X, Y, Z, \ldots$

$$
\begin{aligned}
P, Q ::= {} & \emptyset && \text{nil process} \\
\mid {} & P \parallel Q && \text{parallel composition} \\
\mid {} & X && \text{variable} \\
\mid {} & a(X).P && \text{process input}
\end{aligned}
$$

$$a(X).\big(X \parallel b(Y).Y\big)$$

Higher-Order π-calculus

Communication channel names $a, b, c, \ldots$
Process variables $X, Y, Z, \ldots$

$$
\begin{aligned}
P, Q ::= & \oslash && \text{nil process} \\
& \mid P \parallel Q && \text{parallel composition} \\
& \mid X && \text{variable} \\
& \mid a(X).P && \text{process input} \\
& \mid \overline{a}\langle P \rangle.Q && \text{process output}
\end{aligned}
$$

$$a(X).\big(X \parallel b(Y).Y \parallel \overline{b}\langle\oslash\rangle.\oslash\big)$$

Higher-Order π-calculus

Communication channel names $a, b, c, \ldots$
Process variables $X, Y, Z, \ldots$

$$
\begin{aligned}
P, Q ::= {} & \oslash && \text{nil process} \\
\mid {} & P \parallel Q && \text{parallel composition} \\
\mid {} & X && \text{variable} \\
\mid {} & a(X).P && \text{process input} \\
\mid {} & \overline{a}\langle P \rangle.Q && \text{process output}
\end{aligned}
$$

$$a(X).\big(X \parallel b(Y).Y \parallel \overline{b}\langle\oslash\rangle.\oslash\big) \parallel \overline{a}\big\langle\overline{b}\langle c(Z).Z\rangle.\oslash\big\rangle\oslash$$

Higher-Order π-calculus

Communication channel names $a, b, c, \ldots$
Process variables $X, Y, Z, \ldots$

$$
\begin{aligned}
P, Q ::= {} & \oslash && \text{nil process} \\
\mid {} & P \parallel Q && \text{parallel composition} \\
\mid {} & X && \text{variable} \\
\mid {} & a(X).P && \text{process input} \\
\mid {} & \overline{a}\langle P \rangle.Q && \text{process output}
\end{aligned}
$$

$$
a(X).\big(X \parallel b(Y).Y \parallel \overline{b}\langle \oslash \rangle.\oslash\big) \parallel \overline{a}\big\langle \overline{b}\langle c(Z).Z \rangle.\oslash \big\rangle \oslash
$$

Higher-Order π-calculus

Communication channel names $a, b, c, \ldots$
Process variables $X, Y, Z, \ldots$

$$
\begin{aligned}
P, Q ::= {} & \oslash && \text{nil process} \\
\mid {} & P \parallel Q && \text{parallel composition} \\
\mid {} & X && \text{variable} \\
\mid {} & a(X).P && \text{process input} \\
\mid {} & \overline{a}\langle P \rangle.Q && \text{process output}
\end{aligned}
$$

Communication: $a(X).P \parallel \overline{a}\langle R \rangle.Q \to P\{R/X\} \parallel Q$

$$a(X).\big(X \parallel b(Y).Y \parallel \overline{b}\langle \oslash \rangle.\oslash\big) \parallel \overline{a}\big\langle \overline{b}\langle c(Z).Z \rangle.\oslash \big\rangle \oslash$$

Higher-Order π-calculus

Communication channel names $a, b, c, \ldots$
Process variables $X, Y, Z, \ldots$

$$
\begin{aligned}
P, Q ::= &\ \oslash && \text{nil process} \\
\mid &\ P \parallel Q && \text{parallel composition} \\
\mid &\ X && \text{variable} \\
\mid &\ a(X).P && \text{process input} \\
\mid &\ \overline{a}\langle P \rangle.Q && \text{process output}
\end{aligned}
$$

Communication: $a(X).P \parallel \overline{a}\langle R \rangle.Q \to P\{R/X\} \parallel Q$

$$
\begin{aligned}
&a(X).\big(X \parallel b(Y).Y \parallel \overline{b}\langle \oslash \rangle.\oslash\big) \parallel \overline{a}\big\langle \overline{b}\langle c(Z).Z \rangle.\oslash \big\rangle \oslash \\
\to\ &\overline{b}\langle c(Z).Z \rangle.\oslash \parallel b(Y).Y \parallel \overline{b}\langle \oslash \rangle.\oslash \parallel \oslash
\end{aligned}
$$

Higher-Order π-calculus

Communication channel names $a, b, c, \ldots$
Process variables $X, Y, Z, \ldots$

$P, Q ::= \oslash$		nil process
	$\mid P \parallel Q$	parallel composition
	$\mid X$	variable
	$\mid a(X).P$	process input
	$\mid \overline{a}\langle P \rangle.Q$	process output

Communication: $a(X).P \parallel \overline{a}\langle R \rangle.Q \to P\{R/X\} \parallel Q$

$$a(X).\big(X \parallel b(Y).Y \parallel \overline{b}\langle\oslash\rangle.\oslash\big) \parallel \overline{a}\big\langle \overline{b}\langle c(Z).Z\rangle.\oslash \big\rangle \oslash$$
$$\to \overline{b}\langle c(Z).Z\rangle.\oslash \parallel b(Y).Y \parallel \overline{b}\langle\oslash\rangle.\oslash \parallel \oslash$$

Higher-Order π-calculus

Communication channel names $a, b, c, \ldots$
Process variables $X, Y, Z, \ldots$

$$
\begin{aligned}
P, Q ::= {} & \oslash && \text{nil process} \\
\mid {} & P \parallel Q && \text{parallel composition} \\
\mid {} & X && \text{variable} \\
\mid {} & a(X).P && \text{process input} \\
\mid {} & \overline{a}\langle P\rangle.Q && \text{process output}
\end{aligned}
$$

Communication: $a(X).P \parallel \overline{a}\langle R\rangle.Q \rightarrow P\{R/X\} \parallel Q$

$$
\begin{aligned}
& a(X).\big(X \parallel b(Y).Y \parallel \overline{b}\langle\oslash\rangle.\oslash\big) \parallel \overline{a}\big\langle\overline{b}\langle c(Z).Z\rangle.\oslash\big\rangle\oslash \\
\rightarrow {} & \overline{b}\langle c(Z).Z\rangle.\oslash \parallel b(Y).Y \parallel \overline{b}\langle\oslash\rangle.\oslash \parallel \oslash \\
\rightarrow {} & \oslash \parallel c(Z).Z \parallel \overline{b}\langle\oslash\rangle.\oslash \parallel \oslash
\end{aligned}
$$

Higher-Order π-calculus

Communication channel names $a, b, c, \ldots$
Process variables $X, Y, Z, \ldots$

$P, Q ::= \oslash$		nil process
	$\mid P \parallel Q$	parallel composition
	$\mid X$	variable
	$\mid a(X).P$	process input
	$\mid \overline{a}\langle P\rangle.Q$	process output
	$\mid \nu a.P$	name restriction

Communication: $a(X).P \parallel \overline{a}\langle R\rangle.Q \rightarrow P\{R/X\} \parallel Q$

Name restriction

Syntax: $P, Q ::= \oslash \mid P \parallel Q \mid X \mid a(X).P \mid \overline{a}\langle P \rangle.Q \mid \nu a.P$

$$\nu ab.\Big(\overline{a}\langle \overline{b}\langle \oslash \rangle.\oslash \rangle.P \parallel a(X).\overline{d}\langle X \rangle.Q\Big)$$

Name restriction

Syntax: $P, Q ::= \oslash \mid P \parallel Q \mid X \mid a(X).P \mid \overline{a}\langle P\rangle.Q \mid \nu a.P$

$$\nu ab.\Big(\overline{a}\langle\overline{b}\langle\oslash\rangle.\oslash\rangle.P \parallel a(X).\overline{d}\langle X\rangle.Q\Big) \parallel a(X).X$$

Name restriction

Syntax: $P, Q ::= \oslash \mid P \parallel Q \mid X \mid a(X).P \mid \overline{a}\langle P\rangle.Q \mid \nu a.P$

$$\nu ab.\Big(\overline{a}\langle\overline{b}\langle\oslash\rangle.\oslash\rangle.P \parallel a(X).\overline{d}\langle X\rangle.Q\Big)$$

Name restriction

Syntax: $P, Q ::= \oslash \mid P \parallel Q \mid X \mid a(X).P \mid \overline{a}\langle P\rangle.Q \mid \nu a.P$

$$\begin{aligned} & \nu ab.\Big(\overline{a}\langle\overline{b}\langle\oslash\rangle.\oslash\rangle.P \parallel a(X).\overline{d}\langle X\rangle.Q\Big) \\ \to\ & \nu ab.\Big(P \parallel \overline{d}\langle\overline{b}\langle\oslash\rangle.\oslash\rangle.Q\Big) \end{aligned}$$

Name restriction

Syntax: $P, Q ::= \varnothing \mid P \parallel Q \mid X \mid a(X).P \mid \overline{a}\langle P\rangle.Q \mid \nu a.P$

$$\nu ab.\Big(\overline{a}\langle\overline{b}\langle\varnothing\rangle.\varnothing\rangle.P \parallel a(X).\overline{d}\langle X\rangle.Q\Big)$$
$$\rightarrow \nu ab.\Big(P \parallel \overline{d}\langle\overline{b}\langle\varnothing\rangle.\varnothing\rangle.Q\Big) \parallel d(Y).(Y \parallel R)$$

Name restriction

Syntax: $P, Q ::= \varnothing \mid P \parallel Q \mid X \mid a(X).P \mid \overline{a}\langle P\rangle.Q \mid \nu a.P$

$$\nu ab.\Big(\overline{a}\langle\overline{b}\langle\varnothing\rangle.\varnothing\rangle.P \parallel a(X).\overline{d}\langle X\rangle.Q\Big)$$
$$\to \nu ab.\Big(P \parallel \overline{d}\langle\overline{b}\langle\varnothing\rangle.\varnothing\rangle.Q\Big) \parallel d(Y).(Y \parallel R)$$
$$\to \nu ab.\Big(P \parallel Q\Big) \parallel \overline{b}\langle\varnothing\rangle.\varnothing \parallel R$$

Name restriction

Syntax: $P, Q ::= \oslash \mid P \parallel Q \mid X \mid a(X).P \mid \overline{a}\langle P\rangle.Q \mid \nu a.P$

$$\begin{aligned}
&\nu ab.\Big(\overline{a}\langle\overline{b}\langle\oslash\rangle.\oslash\rangle.P \parallel a(X).\overline{d}\langle X\rangle.Q\Big)\\
\to\ &\nu ab.\Big(P \parallel \overline{d}\langle\overline{b}\langle\oslash\rangle.\oslash\rangle.Q\Big) \parallel d(Y).(Y \parallel R)\\
\simeq\ &\nu ba.\Big(P \parallel \overline{d}\langle\overline{b}\langle\oslash\rangle.\oslash\rangle.Q\Big) \parallel d(Y).(Y \parallel R)
\end{aligned}$$

Name restriction

Syntax: $P, Q ::= \oslash \mid P \parallel Q \mid X \mid a(X).P \mid \overline{a}\langle P\rangle.Q \mid \nu a.P$

$$
\begin{aligned}
& \nu ab.\Big(\overline{a}\langle\overline{b}\langle\oslash\rangle.\oslash\rangle.P \parallel a(X).\overline{d}\langle X\rangle.Q\Big) \\
\to\ & \nu ab.\Big(P \parallel \overline{d}\langle\overline{b}\langle\oslash\rangle.\oslash\rangle.Q\Big) \parallel d(Y).(Y \parallel R) \\
\simeq\ & \nu ba.\Big(P \parallel \overline{d}\langle\overline{b}\langle\oslash\rangle.\oslash\rangle.Q\Big) \parallel d(Y).(Y \parallel R) \\
\simeq\ & \nu b.\Big(\nu a.\big(P \parallel \overline{d}\langle\overline{b}\langle\oslash\rangle.\oslash\rangle.Q\big) \parallel d(Y).(Y \parallel R)\Big)
\end{aligned}
$$

Name restriction

Syntax: $P, Q ::= \oslash \mid P \parallel Q \mid X \mid a(X).P \mid \overline{a}\langle P \rangle.Q \mid \nu a.P$

$$
\begin{aligned}
& \nu ab.\Big(\overline{a}\langle \overline{b}\langle \oslash \rangle.\oslash \rangle.P \parallel a(X).\overline{d}\langle X \rangle.Q\Big) \\
\rightarrow\ & \nu ab.\Big(P \parallel \overline{d}\langle \overline{b}\langle \oslash \rangle.\oslash \rangle.Q\Big) \parallel d(Y).(Y \parallel R) \\
\simeq\ & \nu ba.\Big(P \parallel \overline{d}\langle \overline{b}\langle \oslash \rangle.\oslash \rangle.Q\Big) \parallel d(Y).(Y \parallel R) \\
\simeq\ & \nu b.\Big(\nu a.\Big(P \parallel \overline{d}\langle \overline{b}\langle \oslash \rangle.\oslash \rangle.Q\Big) \parallel d(Y).(Y \parallel R)\Big) \\
\rightarrow\ & \nu b.\Big(\nu a.(P \parallel Q) \parallel \overline{b}\langle \oslash \rangle.\oslash \parallel R\Big)
\end{aligned}
$$

Name restriction

Syntax: $P, Q ::= \oslash \mid P \parallel Q \mid X \mid a(X).P \mid \overline{a}\langle P\rangle.Q \mid \nu a.P$

$$
\begin{aligned}
&\nu ab.\Big(\overline{a}\langle\overline{b}\langle\oslash\rangle.\oslash\rangle.P \parallel a(X).\overline{d}\langle X\rangle.Q\Big)\\
\to\ &\nu ab.\Big(P \parallel \overline{d}\langle\overline{b}\langle\oslash\rangle.\oslash\rangle.Q\Big) \parallel d(Y).(Y \parallel R)\\
\simeq\ &\nu ba.\Big(P \parallel \overline{d}\langle\overline{b}\langle\oslash\rangle.\oslash\rangle.Q\Big) \parallel d(Y).(Y \parallel R)\\
\simeq\ &\nu b.\Big(\nu a.\Big(P \parallel \overline{d}\langle\overline{b}\langle\oslash\rangle.\oslash\rangle.Q\Big) \parallel d(Y).(Y \parallel R)\Big)\\
\to\ &\nu b.\Big(\nu a.(P \parallel Q) \parallel \overline{b}\langle\oslash\rangle.\oslash \parallel R\Big)
\end{aligned}
$$

Input: $P \xrightarrow{a} (X)R$ Output: $Q \xrightarrow{\overline{a}} \nu\widetilde{b}.\langle S\rangle T$

Name restriction

Syntax: $P, Q ::= \oslash \mid P \parallel Q \mid X \mid a(X).P \mid \overline{a}\langle P\rangle.Q \mid \nu a.P$

$$
\begin{aligned}
& \nu ab.\Big(\overline{a}\langle \overline{b}\langle\oslash\rangle.\oslash\rangle.P \parallel a(X).\overline{d}\langle X\rangle.Q\Big) \\
\to\ & \nu ab.\Big(P \parallel \overline{d}\langle \overline{b}\langle\oslash\rangle.\oslash\rangle.Q\Big) \parallel d(Y).(Y \parallel R) \\
\simeq\ & \nu ba.\Big(P \parallel \overline{d}\langle \overline{b}\langle\oslash\rangle.\oslash\rangle.Q\Big) \parallel d(Y).(Y \parallel R) \\
\simeq\ & \nu b.\Big(\nu a.\Big(P \parallel \overline{d}\langle \overline{b}\langle\oslash\rangle.\oslash\rangle.Q\Big) \parallel d(Y).(Y \parallel R)\Big) \\
\to\ & \nu b.\Big(\nu a.(P \parallel Q) \parallel \overline{b}\langle\oslash\rangle.\oslash \parallel R\Big)
\end{aligned}
$$

Input: $P \xrightarrow{a} (X)R$ Output: $Q \xrightarrow{\overline{a}} \nu\widetilde{b}.\langle S\rangle T$

$$
\frac{P \xrightarrow{a} (X)R \qquad Q \xrightarrow{\overline{a}} \nu\widetilde{b}.\langle S\rangle T}{P \parallel Q \to \nu\widetilde{b}.(R\{S/X\} \parallel T)} \quad \widetilde{b} \cap \mathrm{fn}(R) = \emptyset
$$

What we formalize

- Bisimilarity: if $\begin{array}{ccc} P & \sim & Q \\ \downarrow{\scriptstyle\alpha} & & \\ P' & & \end{array}$ then $\begin{array}{ccc} P & \sim & Q \\ \downarrow{\scriptstyle\alpha} & & \downarrow{\scriptstyle\alpha} \\ P' & \sim & Q' \end{array}$
- Congruence: if $P \sim Q$ then $P \parallel R \sim Q \parallel R$, $\nu a.P \sim \nu a.Q$, ...
- Howe's method [CONCUR 15]


Binder
It's a Match!
λx
x

Binders

Process input $a(X).P$: binds process variables X

- Static scope
- Process variables are substituted (by processes)
- Forbids computation

Name restriction $\nu a.P$, $\nu \widetilde{a}.\langle P \rangle Q$: binds names a

- Dynamic scope
- No substitution
- Allows computation

Binders

Process input $a(X).P$: binds process variables X

- Static scope
- Process variables are substituted (by processes)
- Forbids computation

Similar to λ-abstraction: any representation

Name restriction $\nu a.P$, $\nu\widetilde{a}.\langle P\rangle Q$: binds names a

- Dynamic scope
- No substitution
- Allows computation

Binders

Process input $a(X).P$: binds process variables X

- Static scope
- Process variables are substituted (by processes)
- Forbids computation

Similar to λ-abstraction: any representation

Name restriction $\nu a.P$, $\nu\widetilde{a}.\langle P\rangle Q$: binds names a

- Dynamic scope
- No substitution
- Allows computation

Locally nameless (CPP 18) and Nominal

Locally Nameless

Locally nameless

Bound names are de Bruijn indices

$$\nu ba.(\overline{a}\langle \overline{b}\langle\oslash\rangle.\oslash\rangle.\oslash \parallel a(X).\overline{d}\langle X\rangle.\oslash) \parallel d(Y).Y$$
$$\rightsquigarrow \nu.\nu.(\overline{\mathbb{0}}\langle\overline{\mathbb{1}}\langle\oslash\rangle.\oslash\rangle.\oslash \parallel \mathbb{0}(X).\overline{d}\langle X\rangle.\oslash) \parallel d(Y).Y$$

😐 Invalid terms $\nu.\overline{\mathbb{1}}\langle\oslash\rangle.\oslash \Rightarrow$ well-formedness predicate

Locally nameless

Bound names are de Bruijn indices

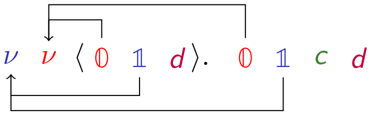
$$\nu ba.(\overline{a}\langle\overline{b}\langle\oslash\rangle.\oslash\rangle.\oslash \parallel a(X).\overline{d}\langle X\rangle.\oslash) \parallel d(Y).Y$$
$$\rightsquigarrow \nu.\nu.(\overline{\mathbb{0}}\langle\overline{\mathbb{1}}\langle\oslash\rangle.\oslash\rangle.\oslash \parallel \mathbb{0}(X).\overline{d}\langle X\rangle.\oslash) \parallel d(Y).Y$$

- 😐 Invalid terms $\nu.\overline{\mathbb{1}}\langle\oslash\rangle.\oslash \Rightarrow$ well-formedness predicate
- 😊 Message output $R \xrightarrow{\overline{a}} \nu\widetilde{b}.\langle P\rangle Q \rightsquigarrow \nu^n\langle P\rangle Q$
- 😐 Scope extrusion

Scope extrusion in locally nameless

Bind c then d in

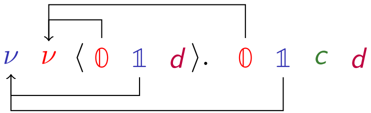
$$\nu b a. \langle P_{abd} \rangle Q_{abcd}$$



Scope extrusion in locally nameless

Bind c then d in

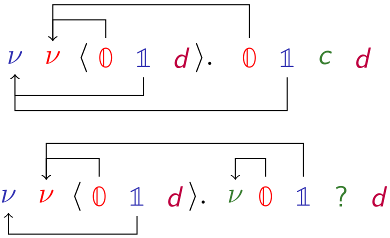
$$\nu b a.\langle P_{abd}\rangle \mathcal{V} c . Q_{abcd}$$



Scope extrusion in locally nameless

Bind c then d in

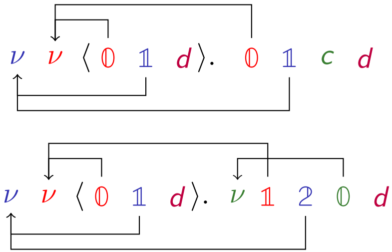
$$\nu b a.\langle P_{abd}\rangle \mathcal{V} c.\, Q_{abcd}$$



Scope extrusion in locally nameless

Bind c then d in

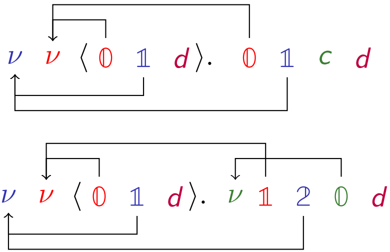
$$\nu ba.\langle P_{abd}\rangle \mathcal{V} c.Q_{abcd}$$



Scope extrusion in locally nameless

Bind c then d in

$$\nu d\, ba.\langle P_{abd}\rangle \mathcal{V} c.Q_{abcd}$$



$$\nu\ \nu\ \nu\ \langle\, \mathbb{0}\ \ \mathbb{1}\ \ \mathbb{2}\,\rangle.\ \nu\ \mathbb{1}\ \ \mathbb{2}\ \ \mathbb{0}\ \ \mathbb{3}$$

Computing under binders

$$\frac{P \to P'}{\nu a.P \to \nu a.P'}$$

$\{\mathbb{K} \to a\}P$ replaces $\mathbb{K}$ with a in P

$$\frac{\forall a \notin \mathrm{fn}(P) \cup \mathrm{fn}(P') \qquad \{\mathbb{0} \to a\}P \to \{\mathbb{0} \to a\}P'}{\nu.P \to \nu.P'}$$

Lemma (Renaming)

☹ *If $\mathscr{P}$ holds for $\{\mathbb{K} \to a\}P$, it holds for $\{\mathbb{K} \to b\}P$ if ...*

Nominal

Nominal

As on paper: names and α-equivalence

$$\nu a.P =_\alpha \nu b.(P\{b/a\}) \text{ if } b \notin \mathsf{fn}(P)$$

Swapping instead of renaming

$$[a \leftrightarrow b](\nu c.Q) \triangleq \nu([a \leftrightarrow b]c).[a \leftrightarrow b]Q$$

Nominal

As on paper: names and α-equivalence

$$\nu a.P =_\alpha \nu b.(P\{b/a\}) \text{ if } b \notin \mathsf{fn}(P)$$

Swapping instead of renaming

$$[a \leftrightarrow b](\nu c.Q) \triangleq \nu([a \leftrightarrow b]c).[a \leftrightarrow b]Q$$

Lemma

- $[b \leftrightarrow c](P\{Q/X\}) =_\alpha ([b \leftrightarrow c]P)\{([b \leftrightarrow c]Q)/X\}$;
- *if* $P =_\alpha P'$ *and* $Q =_\alpha Q'$ *then* $P\{Q/X\} =_\alpha P'\{Q'/X\}$

Nominal

As on paper: names and α-equivalence

$$\nu a.P =_\alpha \nu b.(P\{b/a\}) \text{ if } b \notin \mathsf{fn}(P)$$

Swapping instead of renaming

$$[a \leftrightarrow b](\nu c.Q) \triangleq \nu([a \leftrightarrow b]c).[a \leftrightarrow b]Q$$

Lemma

- *$[b \leftrightarrow c](P\{Q/X\}) =_\alpha ([b \leftrightarrow c]P)\{([b \leftrightarrow c]Q)/X\}$;*
- *if $P =_\alpha P'$ and $Q =_\alpha Q'$ then $P\{Q/X\} =_\alpha P'\{Q'/X\}$*

😐 Working modulo α-equivalence

☺ Swapping lemmas: much simpler than renaming lemmas

Representing outputs

$R \xrightarrow{\overline{a}} \nu\widetilde{b}.\langle P\rangle Q$: list $b_1, \ldots, b_n$, P, and Q

- ☹ New binding structure
- ☹ Redo what we did for processes
- ☹ Manipulation of lists

Evaluation

Nominal

Locally nameless

Evaluation

	Nominal		Locally nameless
intrinsic	α-equivalence	$<$	wf predicate
	name	$>$	de Bruijn indices

Evaluation

	Nominal		Locally nameless
intrinsic	α-equivalence	$<$	wf predicate
	name	$>$	de Bruijn indices
outputs	list of names	$<$	1 number
	specific α-equivalence	$\ll$	$\emptyset$

Evaluation

	Nominal		Locally nameless
intrinsic	α-equivalence	$<$	wf predicate
	name	$>$	de Bruijn indices
outputs	list of names	$<$	1 number
	specific α-equivalence	$\ll$	$\emptyset$
renaming	swapping	$\ggg$	renaming

Evaluation

	Nominal		Locally nameless
intrinsic	α-equivalence	$<$	wf predicate
	name	$>$	de Bruijn indices
outputs	list of names	$<$	1 number
	specific α-equivalence	$\ll$	$\emptyset$
renaming	swapping	$\ggg$	renaming
total	4k lines	$\gg$	5k lines

New challenger incoming

pure deBruijn indices

Evaluation (bis)


Nominal


de Bruijn


Locally nameless

Evaluation (bis)

	Nominal	de Bruijn	Locally nameless
intrinsic	α-equivalence	$\emptyset$	wf predicate
	name	dB indices	dB indices

Evaluation (bis)

	Nominal	de Bruijn	Locally nameless
intrinsic	α-equivalence	$\emptyset$	wf predicate
	name	dB indices	dB indices
outputs	list of names	1 number	1 number
	specific α-equivalence	$\emptyset$	$\emptyset$

Evaluation (bis)

			
	Nominal	de Bruijn	Locally nameless
intrinsic	α-equivalence	$\emptyset$	wf predicate
	name	dB indices	dB indices
outputs	list of names	1 number	1 number
	specific α-equivalence	$\emptyset$	$\emptyset$
renaming	$[a \leftrightarrow b]P$	$map\ f\ \ P$	$\{\mathbb{0} \to a\}P$
		$f : \mathbb{N} \to \mathbb{N}$	

Evaluation (bis)

	Nominal	de Bruijn	Locally nameless
intrinsic	α-equivalence	$\emptyset$	wf predicate
	name	dB indices	dB indices
outputs	list of names	1 number	1 number
	specific α-equivalence	$\emptyset$	$\emptyset$
renaming	$[a \leftrightarrow b]P$	$map\ f\ \ P$	$\{\mathbb{0} \to a\}P$
		$f : \mathbb{N} \to \mathbb{N}$	
total	4k lines	3k lines	5k lines

Conclusion and Future Work

- de Bruijn wins! (in a cripples fight)
- More automation, tactics
- Add support for name restriction to existing libraries (Autosubst?)
- More expressive calculi
- Tools for bisimulation (Howe's method)