A π-calculus with preorders

Daniel Hirschkoff, **Jean-Marie Madiot**, Davide Sangiorgi

École Normale Supérieure de Lyon
Università di Bologna

Chocola, Lyon, 2013-05-16

Summary

π-calculus

The π-calculus is a process calculus which is name-passing:

$$P ::= 0 \mid P|Q \mid !P \mid \overline{a}\langle b\rangle.P \mid a(x).P \mid (\nu a)P$$

$$\overline{a}\langle b\rangle.P \mid a(x).Q \quad \rightarrow \quad P \mid Q[b/x]$$

Examples:

- $link = !a(x).\bar{b}\langle x\rangle$
- $spy = !a(x).(\overline{a}\langle x\rangle \mid \overline{\mathsf{third}}\langle x\rangle)$

Explicit fusions [Wischik, Gardner, 00]

Non-binding input, construct "$=$" to equate names:

$$P ::= 0 \mid P|Q \mid !P \mid \overline{a}\langle b\rangle.P \mid a\langle c\rangle.P \mid (\nu a)P \mid b = c$$

$$\overline{a}\langle b\rangle.P \mid a\langle c\rangle.Q \quad \rightarrow \quad P \mid Q \mid b = c$$

- $(P \mid b = c) \;\equiv\; (P[c/b] \mid b = c)$,
- only one binder,
- simpler theory than π (only one bisimulation),
- outputs $\overline{a}\langle b\rangle$ and inputs $a\langle b\rangle$ are of the same kind.

Fusion-like calculi [Parrow, Victor, Fu, Wischik, Gardner, . . .]

- Explicit fusions:

$$\overline{a}\langle b\rangle.P \mid a\langle c\rangle.Q \;\rightarrow\; P \mid Q \mid b = c$$

(and customized $\equiv$)

- Fusion calculus and χ-calculus contain these rules:

$$(\boldsymbol{\nu} c)(\overline{a}\langle b\rangle.P \mid a\langle c\rangle.Q \mid R) \;\rightarrow\; (P \mid Q \mid R)[b/c]$$
$$(\boldsymbol{\nu} b)(\overline{a}\langle b\rangle.P \mid a\langle c\rangle.Q \mid R) \;\rightarrow\; (P \mid Q \mid R)[c/b]$$

- Update calculus and asymmetric χ-calculus contain this rule:

$$(\boldsymbol{\nu} c)(\overline{a}\langle b\rangle.P \mid a\langle c\rangle.Q \mid R) \;\rightarrow\; (P \mid Q \mid R)[b/c]$$

Summary

Types in π : i/o-types [Pierce Sangiorgi, 93]

Types for names:

$$
\begin{array}{rcll}
T & ::= & \mathbf{1} & \text{no capability} \\
& | & iT & \text{receive capability of } T\text{-values} \\
& | & oT & \text{send capability of } T\text{-values} \\
& | & \sharp T & \text{both capabilities } (iT \text{ and } oT)
\end{array}
$$

$$
\frac{\Gamma \vdash a : iT \quad \Gamma, x : T \vdash P}{\Gamma \vdash a(x).P} \qquad \frac{\Gamma \vdash a : oT \quad \Gamma \vdash b : T \quad \Gamma \vdash P}{\Gamma \vdash \overline{a}\langle b \rangle.P}
$$

$$
\frac{}{\Gamma, a : \sharp T \vdash a : oT} \qquad \frac{}{\Gamma, a : \sharp T \vdash a : iT}
$$

(shallow subtyping)

Subtyping in i/o-types

$T_1 \leq T_2$: a T_1-name is also a T_2-name.
(easier to use, harder to provide)

$$\frac{}{\sharp T \leq iT} \qquad \frac{}{\sharp T \leq oT}$$

$$\frac{T_1 \leq T_2}{iT_1 \leq iT_2} \qquad \frac{T_1 \leq T_2}{oT_{2} \leq oT_{1}}$$

Say $T_1 \leq T_2$.

- $(x : T_1)$ is easier to use in P than $(x : T_2)$;
  using $(a : iT_1)$ in $a(x).P$ is easier than using $(a : iT_2)$.
  (Guarantee from outside)
- providing T_1-names is harder,
  so using $a : oT_1$ is harder since in $\overline{a}b.P$ you have to provide b.
  (Guarantee from you)

Typing issues in fusions

Fusion calculi break (naive) i/o-types:

$$\left.\begin{array}{c}\Gamma \vdash P \\ P \to P'\end{array}\right\} \quad \not\Rightarrow \quad \Gamma \vdash P' \ .$$

It seems reasonable to have:

$$\begin{array}{rcl} a : \sharp i,\ b : \sharp & \vdash & \overline{b}\langle\rangle \mid \overline{a}\langle b\rangle \\ a : \sharp i,\ c : i & \vdash & a\langle c\rangle \end{array}$$

but:

$$\begin{array}{rcl} a : \sharp i,\ b : \sharp,\ c : i & \vdash & \overline{b}\langle\rangle \mid \overline{a}\langle b\rangle \mid a\langle c\rangle \\ & & \downarrow \\ a : \sharp i,\ b : \sharp,\ c : i & \not\vdash & \overline{b}\langle\rangle \mid b = c \end{array}$$

Origin of the problem

In plain fusions, free inputs can "send" (their object replaces outputs'):

$$u\langle b\rangle \mid (\boldsymbol{\nu} a)(\overline{u}\langle a\rangle \mid P) \quad \rightarrow \quad P[b/a]$$

The substitution $P[b/a]$ suggests $T_b \leq T_a$.

The i/o-types suggest $T_a \leq T_b$.

Naive patch

Bad for i/o-types: output objects can be replaced (fusions):

$$u\langle b\rangle \mid (\boldsymbol{\nu} a)(\overline{u}\langle a\rangle \mid P) \xrightarrow[\text{replacing the emitted name}]{} P[b/a]$$

What about only input objects being replaced (oriented fusions/update)?

$$\overline{u}\langle b\rangle \mid (\boldsymbol{\nu} a)(u\langle a\rangle \mid P) \xrightarrow[\text{replacing the received name}]{} P[b/a]$$

this does not work: outputs will be replaced, too:

$$\begin{aligned} & \overline{u}\langle b\rangle \mid (\boldsymbol{\nu} ac)(u\langle c\rangle \mid P \mid \overline{v}\langle a\rangle \mid v\langle c\rangle) \\ \xrightarrow[\text{replacing the received name}]{} & \overline{u}\langle b\rangle \mid (\boldsymbol{\nu} a)(u\langle a\rangle \mid P) \\ \xrightarrow[\text{replacing the received name}]{} & P[b/a] \end{aligned}$$

Sum up

Theorem

In a fusion-calculus where the following reduction is admissible

$$(\boldsymbol{\nu} c)(\overline{a}\langle b\rangle.P \mid a\langle c\rangle.Q) \;\Rightarrow\; (P \mid Q)[b/c] \;,$$

a regular type system makes subtyping **essentially trivial**, *i.e. in a well-typed process* $\Gamma \vdash P$, *if* $\Gamma(a) \leq \Gamma(b)$ *we can exchange* a *and* b *anywhere in a process and it remains well-typed: e.g.* $\Gamma \vdash P[a/b]$ *and* $\Gamma \vdash P[b/a]$.
(regular: basic, standard assumptions about the type system.)

Fusions "$a = b$" induce an equivalence relation.
Clash between:

- the symmetry of this equivalence relation, and
- the asymmetry of subtyping.

Summary

πP : π with preorders

Idea: removing symmetry from fusions' equivalence relation

$$P ::= \overline{a}\langle b\rangle.P \mid \boldsymbol{a\langle b\rangle}.P \mid \boldsymbol{a/b} \mid (\nu a)P \mid 0 \mid P|P \mid !P$$

- an *arc* $\boldsymbol{a/b}$ makes a an alias of b:

  "a can be used in place of b"

  $$b \leq a$$

- $\boldsymbol{a\langle b\rangle}$ is a free input prefix, symmetric of $\overline{a}\langle b\rangle$

  $$\overline{c}\langle a\rangle.P \mid c\langle b\rangle.Q \;\rightarrow\; P \mid a/b \mid Q$$

Details:

- arcs define the preorder,
- $\equiv$ is not changed (same as in π),
- ν is the only binder.

Semantics

$$\frac{C \rhd a \curlyvee b}{C[\overline{a}\langle c\rangle.P \mid b\langle d\rangle.Q] \to C[P \mid c/d \mid Q]}$$

$C \rhd a \curlyvee b$: some name u can impersonate a and b.

$$C \rhd a \curlyvee b \quad \text{iff} \quad \exists u \quad a \leq u \text{ and } b \leq u$$

$$\text{e.g. } C = (\boldsymbol{\nu} uv)(u/a \mid u/v \mid v/b \mid -)$$

$$\begin{array}{ll} & (\boldsymbol{\nu} uv)(u/a \mid u/v \mid v/b \mid \overline{a}c \mid bd) \\ \to & (\boldsymbol{\nu} uv)(u/a \mid u/v \mid v/b \mid c/d) \end{array}$$

πP examples

$$a/c \mid \overline{a}\langle x\rangle \mid c\langle y\rangle.P \to a/c \mid P \mid x/y$$

$$c/a \mid \overline{a}\langle x\rangle \mid c\langle y\rangle.P \to c/a \mid P \mid x/y$$

$a \curlyvee c$ $$u/a \mid u/c \mid \overline{a}\langle x\rangle \mid c\langle y\rangle.P \to u/a \mid u/c \mid P \mid x/y$$

$a \not\curlyvee c$ $$a/u \mid c/u \mid \overline{a}\langle x\rangle \mid c\langle y\rangle.P \not\to$$

$$(\boldsymbol{\nu} u)(a/u \mid b/u \mid \overline{u}\langle x\rangle) \mid a\langle y\rangle.P_1 \mid b\langle z\rangle.P_2 \begin{array}{l} \nearrow P_1 \mid x/y \mid \dots \\ \searrow P_2 \mid x/z \mid \dots \end{array}$$

πP examples

$$a/c \mid \overline{a}\langle x\rangle \mid c\langle y\rangle.P \to a/c \mid P \mid x/y$$

$$c/a \mid \overline{a}\langle x\rangle \mid c\langle y\rangle.P \to c/a \mid P \mid x/y$$

$a \curlyvee c$
$$u/a \mid u/c \mid \overline{a}\langle x\rangle \mid c\langle y\rangle.P \to u/a \mid u/c \mid P \mid x/y$$

$a \not\curlyvee c$
$$a/u \mid c/u \mid \overline{a}\langle x\rangle \mid c\langle y\rangle.P \not\to$$

$$(\boldsymbol{\nu} u)(a/u \mid b/u \mid \overline{u}\langle x\rangle) \mid a\langle y\rangle.P_1 \mid b\langle z\rangle.P_2 \nearrow P_1 \mid x/y \mid \dots$$

$$(\boldsymbol{\nu} u)(a/u \mid b/u \mid \overline{u}\langle x\rangle) \mid a\langle y\rangle.P_1 \mid b\langle z\rangle.P_2 \searrow P_2 \mid x/z \mid \dots$$

$$(\boldsymbol{\nu} u)(a/u \mid b/u \mid \overline{u}x) \sim \overline{a}\langle x\rangle + \overline{b}\langle x\rangle$$

πP : polarized fusions

$$a \curlyvee b \quad \Rightarrow \quad \overline{a}t.P \mid bu.Q \to P \mid Q \mid t/\!\!u$$

Positive and negative occurrences of names:

$$P ::= a/\!\!b \mid a\langle b\rangle.P \mid \overline{a}\langle b\rangle.P \mid P|P \mid !P \mid 0 \mid (\nu a)P$$

- Negative occurrences of b may affect other occurrences of b;
- polarity does not change along reduction.

The π-calculus is positive

$$P ::= a/\!\!\!b \mid a\langle b\rangle.P \mid \overline{a}\langle b\rangle.P \mid P|P \mid !P \mid 0 \mid (\nu a)P$$

- The usual π-calculus is positive: all negative positions are immediately bound:

$$a(x).P = (\boldsymbol{\nu} x)(a\langle x\rangle.P) \ ,$$
$$P[b/x] = (\boldsymbol{\nu} x)(P \mid b/\!\!\!x) \ .$$

- Extrapolating: in if a appears in P:
  - in 0 negative occurrence: a is a **channel** in $(\boldsymbol{\nu} a)P$,
  - in 1 negative occurrence: a is a **variable** in $(\boldsymbol{\nu} a)P$,
  - in 2+ negative occurrences: a is "concurrently bound" in $(\boldsymbol{\nu} a)P$.

Behavioural equivalence

Definition (Barbed congruence)

Biggest symmetric relation $\simeq$ such that $P \simeq Q$ implies:

- $C[P] \simeq C[Q]$,
- $P{\downarrow_b}$ iff $Q{\downarrow_b}$,
- if $P \to P'$ then $Q \to Q'$ for some Q' such that $P' \simeq Q'$.

Arcs can be used to operate substitutions:

$$\begin{aligned}(\nu x)(a/x \mid P) &\simeq P[a/x] \quad \text{if } x > 0\\ (\nu x)(x/a \mid P) &\simeq P[a/x] \quad \text{if } x < 0\\ (\nu x)(x/a \mid a/x \mid P) &\simeq P[a/x]\end{aligned}$$

Context-free characterization: in the strong case we have an LTS with simple labels $(\tau, ab, \overline{a}b)$:

$$P \sim Q \quad \text{iff} \quad P \simeq Q$$

Context free characterization of $\simeq$

Bisimulation:

$$\begin{array}{ccc} P & \overset{\mathcal{R}}{\text{———}} & Q \\ \big\downarrow \alpha & & \big\downarrow \alpha \\ P' & \overset{\mathcal{R}}{\text{———}} & Q' \end{array}$$

$$\frac{P \;\mathcal{R}\; Q}{P|a/b \;\mathcal{R}\; Q|a/b}$$

$$\frac{P \;\mathcal{R}\; Q}{P \rhd a \curlyvee b \;\Leftrightarrow\; Q \rhd a \curlyvee b}$$

LTS:

$$\frac{P \longrightarrow P'}{P \xrightarrow{\tau} P'}$$

$$\frac{E \rhd a \curlyvee b \qquad b, d \text{ not bound in } E}{E[a\langle c\rangle.P] \xrightarrow{b\langle d\rangle} E[d/c|P]}$$

$$\frac{E \rhd a \curlyvee b \qquad b, d \text{ not bound in } E}{E[\overline{a}\langle c\rangle.P] \xrightarrow{\overline{b}\langle d\rangle} E[c/d|P]}$$

Summary

Types for πP

$$T ::= iT \mid oT \mid \sharp T \mid \mathbf{1}$$

$$\frac{}{\sharp T \leq iT} \qquad \frac{}{\sharp T \leq oT} \qquad \frac{T_1 \leq T_2}{iT_1 \leq iT_2} \qquad \frac{T_1 \leq T_2}{oT_2 \leq oT_1}$$

$$\frac{\Gamma \vdash P \quad \Gamma(a) \leq oT \quad \Gamma(b) \leq T}{\Gamma \vdash \overline{a}\langle b\rangle.P} \qquad \frac{\Gamma \vdash P \quad \Gamma(a) \leq iT \quad \Gamma(b) \geq T}{\Gamma \vdash a\langle b\rangle.P} \qquad \frac{\Gamma(a) \leq \Gamma(b)}{\Gamma \vdash a/b}$$

$\uparrow$ like in the π-calculus ($\Gamma(a) \leq .. \wedge \Gamma(b) \leq ..$)

$\uparrow$ $\uparrow$ backwards constraint on b: $\Gamma(b) \geq T$

subtyping works backwards on negative occurrences

The problem of fusions disappears

Counterexample in fusions:

$$\begin{array}{lcl} a : \sharp i,\ b : \sharp & \vdash & \bar{b}\langle\rangle \mid \overline{a}\langle b\rangle \\ a : \sharp i,\ c : i & \vdash & a\langle c\rangle \end{array}$$

not a problem in πP:

$$\begin{array}{lcl} a : \sharp i,\ b : \sharp,\ c : i & \vdash & \bar{b}\langle\rangle \mid \overline{a}\langle b\rangle \mid a\langle c\rangle \\ & & \quad\downarrow \\ a : \sharp i,\ b : \sharp,\ c : i & \vdash & \bar{b}\langle\rangle \mid b/c \quad \text{(no output on } c\text{)} \end{array}$$

$$(\Gamma(b) = \sharp \leq i = \Gamma(c))$$

Narrowing and polarities

To prove soundness we usually rely on narrowing:

$$\left.\begin{array}{c} T_1 \leq T_2 \\ \Gamma, a : T_2 \vdash P \end{array}\right\} \Rightarrow \Gamma, a : T_1 \vdash P$$

Narrowing and polarities

To prove soundness we usually rely on narrowing:

$$\left.\begin{array}{c}\Gamma(a) \leq \Gamma(b) \\ \Gamma \vdash P\end{array}\right\} \Rightarrow \Gamma \vdash P[a/b]$$

Narrowing and polarities

To prove soundness we usually rely on narrowing:

$$\left.\begin{array}{c}\Gamma(a) \leq \Gamma(b) \\ \Gamma \vdash P\end{array}\right\} \Rightarrow \Gamma \vdash P[a/b]$$

... but it does not hold in πP:

$$a{:}i,\ b{:}i,\ c{:}\sharp \ \vdash\ \overline{c} \mid (\nu x)(x\langle a\rangle \mid \overline{x}\langle b\rangle)$$

$$a{:}i,\ b{:}i,\ c{:}\sharp \ \nvdash\ \overline{c} \mid (\nu x)(x\langle c\rangle \mid \overline{x}\langle b\rangle) \rightarrow (\overline{c} \mid b/c) \qquad \text{(can do } \overline{b}\text{)}$$

Properties of the type system

We can make narrowing polarized: if $\Gamma(a) \leq \Gamma(b)$ then

$$\Gamma \vdash P[b/x] \;\Rightarrow\; \Gamma \vdash P[a/x] \qquad \text{if } x > 0 \tag{1}$$

$$\Gamma \vdash P[a/x] \;\Rightarrow\; \Gamma \vdash P[b/x] \qquad \text{if } x < 0 \tag{2}$$

This implies soundness: if $P \to P'$ and $\Gamma \vdash P$ then $\Gamma \vdash P'$.

In π binders hide all negative occurrences hence narrowing is only positive; thus we only need (1) in the π-calculus.

Summary

Private names

In the π-calculus (expansion law):

$$(\nu ab)\overline{u}\langle a,b\rangle.(\overline{a} \mid b) \sim_\pi (\nu ab)\overline{u}\langle a,b\rangle.(\overline{a}.b + b.\overline{a})$$

Which does not hold in the case of fusions:

$$\begin{array}{ccc} u\langle d,d\rangle \mid (\nu ab)\overline{u}\langle a,b\rangle.(\overline{a} \mid b) & \not\sim_f & u\langle d,d\rangle \mid (\nu ab)\overline{u}\langle a,b\rangle.(\overline{a}.b + b.\overline{a}) \\ \downarrow & & \downarrow \\ (\overline{d} \mid d) & \not\sim_f & (\overline{d}.d + d.\overline{d}) \end{array}$$

Modelisation in fusions

$$(\nu ab)\overline{u}\langle a,b\rangle.(\overline{a} \mid b) \not\sim_f (\nu ab)\overline{u}\langle a,b\rangle.(\overline{a}.b + b.\overline{a})$$

A bad client fusing names:

$$c\langle d, d\rangle \quad \mid \quad (\boldsymbol{\nu} ab)(\mathsf{int\_server}_a \mid \mathsf{list\_server}_b \mid \overline{c}\langle a,b\rangle.P \mid \ldots)$$

Modelisation is more difficult using fusions than π:

- we lack private names,
- we lack types (beyond simple types).

Private names in πP

Better control on names:

- if a is not negative in P then it is private in $(\nu a)P$;
- if a and b are not negative in P then a and b in $(\nu ab)P$ will never be equated by the context.

In fusions:

$$\begin{aligned} & (\nu ab)(P \mid \overline{u}\langle a,b\rangle) \quad \mid \; u\langle d,d\rangle \\ \to\; & (\nu ab)(P \mid a = d \mid b = d) \ . \end{aligned}$$

In πP a and b are not compromised:

$$\begin{aligned} & (\nu ab)(P \mid \overline{u}\langle a,b\rangle) \quad \mid \; u\langle d,d\rangle \\ \to\; & (\nu ab)(P \mid a/d \mid b/d) \ . \end{aligned}$$

In πP the law holds:

$$(\nu ab)\overline{u}\langle a,b\rangle.(\overline{a} \mid b) \sim_{\pi\mathtt{P}} (\nu ab)\overline{u}\langle a,b\rangle.(\overline{a}.b + b.\overline{a}) \ .$$

Summary

Encoding π

πP is close to π (more than fusions):

$$a(x).P \leftrightarrow (\nu x)(a\langle x\rangle.P)$$

And since $a(x).P \mid \overline{a}\langle b\rangle \rightarrow (\nu x)(P \mid b/x)$ we would like :

$$(\nu x)(P \mid b/x) = P[b/x]$$

Encoding π

πP is close to π (more than fusions):

$$a(x).P \leftrightarrow (\nu x)(a\langle x\rangle.P)$$

And since $a(x).P \mid \overline{a}\langle b\rangle \to (\nu x)(P \mid b/x)$ we would like :

$$(\nu x)(P \mid b/x) = P[b/x]$$

Which holds if x is positive in P (True since P is a π-process).

Remarks :

- full abstraction for Aπ (maybe for π as well),
- full abstraction for a subcalculus of πP.

Encoding explicit fusions

$$\llbracket a = b \rrbracket = a/b \mid b/a$$
$$\llbracket \overline{a}\langle b\rangle.P \rrbracket = (\boldsymbol{\nu} u)\overline{a}\langle b, u\rangle.u\langle b\rangle.\llbracket P \rrbracket$$
$$\llbracket a\langle c\rangle.Q \rrbracket = (\boldsymbol{\nu} v)a\langle c, v\rangle.\overline{v}\langle c\rangle.\llbracket Q \rrbracket$$

$$\overline{a}\langle b\rangle.P \mid a\langle c\rangle.Q \to b = c \mid P \mid Q$$

$$\begin{aligned}\llbracket \overline{a}\langle b\rangle.P \mid a\langle c\rangle.Q \rrbracket &\to (\boldsymbol{\nu} uv)(b/c \mid u/v \mid u\langle b\rangle.\llbracket P \rrbracket \mid \overline{v}\langle c\rangle.\llbracket Q \rrbracket) \\ &\approx (\boldsymbol{\nu} uv)(u/v \mid b/c \mid c/b \mid \llbracket P \rrbracket \mid \llbracket Q \rrbracket) \\ &\sim \llbracket b = c \mid P \mid Q \rrbracket\end{aligned}$$

Operational correspondence:

$$\begin{aligned} P \to P' \quad &\Rightarrow \quad \llbracket P \rrbracket \to\approx \llbracket P' \rrbracket \\ \llbracket P \rrbracket \to Q \quad &\Rightarrow \quad P \to P' \ \wedge \ Q \approx \llbracket P' \rrbracket \end{aligned}$$

Encoding implicit fusions

$$\llbracket \overline{a}\langle b\rangle.P \rrbracket = (\boldsymbol{\nu} u)\overline{a}\langle b,u\rangle.u\langle b\rangle.\llbracket P\rrbracket$$
$$\llbracket a\langle c\rangle.Q \rrbracket = (\boldsymbol{\nu} v)a\langle c,v\rangle.\overline{v}\langle c\rangle.\llbracket Q\rrbracket$$

$$\llbracket \overline{a}\langle b\rangle.P \mid a\langle c\rangle.Q \rrbracket \to\approx\ (b/c \mid c/b) \mid \llbracket P \mid Q \rrbracket$$
$$\overline{a}\langle b\rangle.P \mid a\langle c\rangle.Q \xrightarrow{\{b=c\}} P \mid Q$$

Operational correspondence:

$$P \to P' \quad \Rightarrow \quad \llbracket P\rrbracket \to\approx \llbracket P'\rrbracket$$
$$\llbracket P\rrbracket \to Q \quad \Rightarrow \quad \begin{cases} \quad P \xrightarrow{\tau} P' & \wedge\ Q \approx \llbracket P'\rrbracket \\ \vee\ \ P \xrightarrow{\{a=b\}} P' & \wedge\ Q \approx \llbracket P'\rrbracket \mid a/b \mid b/a \end{cases}$$

Conclusion

Fusions:

- behavioural theory scales from π (and gets simplified),
- types do not!

A refinement of fusions, $\pi\mathrm{P}$:

- has a preorder instead of an equivalence,
  (names get sorted, not equated)
- inherits π's types,
- also, has a real notion of restriction.

Now:

- weak case of the theory,
- extensions (e.g. what matching would be),
- abstract machine ($\sim$ fusion machine).

Thank you for your time.

The χ-calculus

Symmetric χ-calculus (3,4) $\simeq$ fusion calculus
Asymmetric χ-calculus (3) $\simeq$ update calculus

$$(\boldsymbol{\nu} x)(a[x].P \mid \overline{a}[y].Q \mid R) \rightarrow_\chi P[y/x] \mid Q[y/x] \mid R[y/x] \quad (3)$$
$$(\boldsymbol{\nu} x)(\overline{a}[x].P \mid a[y].Q \mid R) \rightarrow_\chi P[y/x] \mid Q[y/x] \mid R[y/x] \quad (4)$$

Even in (3) alone input objects are being rewritten, e.g.:

$$\begin{aligned} & (\boldsymbol{\nu} xzb)(\bar{b}[x] \mid b[z].a[z] \mid \overline{a}[y] \mid R) \\ \rightarrow_\chi\ & (\boldsymbol{\nu} x)(a[x] \mid \overline{a}[y] \mid R) \\ \rightarrow_\chi\ & R[y/x] \end{aligned}$$

The χ-calculus

Rules of (symmetric) χ:

$$(\boldsymbol{\nu} x)(a[x].P \mid \overline{a}[y].Q \mid R) \to_\chi P[y/x] \mid Q[y/x] \mid R[y/x]$$
$$(\boldsymbol{\nu} x)(\overline{a}[x].P \mid a[y].Q \mid R) \to_\chi P[y/x] \mid Q[y/x] \mid R[y/x]$$

Encoding into πP:

$$[\![\overline{a}\langle b\rangle.P]\!] = (\boldsymbol{\nu} u)\overline{a}\langle b,u\rangle.u\langle b\rangle.[\![P]\!]$$
$$[\![a\langle c\rangle.Q]\!] = (\boldsymbol{\nu} v)a\langle c,v\rangle.\overline{v}\langle c\rangle.[\![Q]\!]$$

$$P \to_\chi P' \quad \Rightarrow \quad [\![P]\!] \to\cong [\![P']\!]$$
$$[\![P]\!] \to P_1 \quad \Rightarrow \quad \begin{cases} \quad P \xrightarrow{[y/x]}_\chi P' \ \wedge \ (\boldsymbol{\nu} x)P_1 \cong [\![P']\!] \\ \vee \quad P \xrightarrow{\tau}_\chi P' \ \wedge \ P_1 \cong [\![P']\!] \end{cases} \qquad \frac{P \xrightarrow{[y/x]} P'}{(\boldsymbol{\nu} x)P \xrightarrow{\tau} P'}$$

Asymmetric χ: not sure how.

Fusions: an equivalence relation on names

	condition		effect
π	$\overline{a}\langle c\rangle.P \mid b(x).Q$ syntactic equality $a = b$	$\rightarrow$	$P \mid Q[c/x]$ syntactic replacement $x \mapsto c$
fusions	$\overline{a}\langle c\rangle.P \mid b\langle d\rangle.Q$ equivalence relation $a \sim b$	$\rightarrow$	$P \mid Q \mid c = d$ equivalence modification $c \sim d$

Fusions: an equivalence relation on names

	condition		effect
π	$\overline{a}\langle c\rangle.P \mid b(x).Q$ syntactic equality $a = b$	$\rightarrow$	$P \mid Q[c/x]$ syntactic replacement $x \mapsto c$
fusions	$\overline{a}\langle c\rangle.P \mid b\langle d\rangle.Q$ equivalence relation $a \sim b$	$\rightarrow$	$P \mid Q \mid c = d$ equivalence modification $c \sim d$
πP	$\overline{a}\langle c\rangle.P \mid b\langle d\rangle.Q$ preorder $a \curlyvee b$	$\rightarrow$	$P \mid Q \mid c/d$ preorder modification $c \succcurlyeq d$

Two semantics

Eager semantics

$$\overline{a}\langle b\rangle.P \mid a\langle c\rangle.Q \;\to\; P \mid b/c \mid Q$$
$$\overline{a}\langle c\rangle.P \mid b/a \;\to\; \overline{b}\langle c\rangle.P \mid b/a$$
$$a\langle c\rangle.P \mid b/a \;\to\; b\langle c\rangle.P \mid b/a$$

Some remarks:

- a/b can act at any time,
- more τ-reductions ($\approx$),
- easier to implement,
- $a.a \not\approx a \mid a$.

By-need semantics

$$\frac{C \vdash u/a \text{ and } C \vdash u/b}{C[\overline{a}\langle c\rangle.P \mid b\langle d\rangle.Q] \to C[P \mid c/d \mid Q]}$$

Some remarks:

- a/b acts as late as possible,
- more τ-sensible ($\sim$, $\approx$),
- more expressive,
- $a.a \sim a \mid a$.

Typing πI: π with internal mobility [Sangiorgi, 96]

Sending only fresh names. (Subcalculus of π)

$$((\nu x)\overline{a}x.P) \mid a(x).Q \quad \to \quad (\nu x)(P \mid Q)$$

- simpler theory, expressiveness (π, λ);
- duality at the operational level:

$$\overline{a(x).\overline{b}(y).P} = \overline{a}(x).b(y).\overline{P} \ ,$$
$$P \to P' \quad \Leftrightarrow \quad \overline{P} \to \overline{P'} \ .$$

- But subtyping does not improve expressiveness.

Typing πI: π with internal mobility [Sangiorgi, 96]

Sending only fresh names. (Subcalculus of π)

$$\overline{a}(x).P \mid a(x).Q \quad \rightarrow \quad (\nu x)(P \mid Q)$$

- simpler theory, expressiveness (π, λ);
- duality at the operational level:

$$\overline{a(x).\overline{b}(y).P} = \overline{a}(x).b(y).\overline{P} \ ,$$

$$P \rightarrow P' \quad \Leftrightarrow \quad \overline{P} \rightarrow \overline{P'} \ .$$

- But subtyping does not improve expressiveness.

A "maximal" solution: $\overline{\pi}$ [Concur'12]

The calculus $\overline{\pi}$:

- has duality,
- symmetry, free input and output prefixes,
- has types,
- is a conservative extension of π.

With the following drawbacks:

- not minimal (gotten by saturation of π),
- verbose,
- it is necessarily typed,
- no fusion constructor.

Everything but the first item: remaining of this talk.

Preservation of scope

$$\ldots \mid \overbrace{b\langle e\rangle \mid \ldots \mid \underbrace{a\langle b\rangle.P \mid \ldots \mid c/b}_{\text{binders of } b} \mid \ldots \mid \overline{c}\langle b\rangle}^{\text{scope of the binders of } b} \mid \ldots$$

With usual substitution:

$$R \mid (\boldsymbol{\nu} b)\overbrace{(P \mid \ldots \mid v\langle b\rangle \mid u\langle b\rangle)}^{\text{scope of } b} \mid \overbrace{\overline{u}\langle c\rangle \mid \overline{c}\langle\rangle}^{\text{scope of } c}$$

$$\downarrow$$

$$R \mid \underbrace{(P[c/b] \mid \ldots \mid v\langle c\rangle) \mid \overline{c}\langle\rangle}_{\text{new scope of } c}$$

- This breaks subtyping,
- controling the binders helps controling the scope,
- πP preserves locality of binders.

πI : π with internal mobility [Sangiorgi, 96]

Sending only fresh names (subcalculus of π):

$$((\nu x)\overline{a}\langle x\rangle.P) \mid a(x).Q \;\rightarrow\; (\nu x)(P \mid Q)$$

- expressive system (λ, π),
- simpler theory,
- the natures of $\overline{a}(x)$ and $a(x)$ are similar.

πI : π with internal mobility [Sangiorgi, 96]

Sending only fresh names (subcalculus of π):

$$\overline{a}(x).P \mid a(x).Q \quad \to \quad (\nu x)(P \mid Q)$$

- expressive system (λ, π),
- simpler theory,
- the natures of $\overline{a}(x)$ and $a(x)$ are similar.

Trying to invent typing rules

Fusions are two-way substitutions:

$$\begin{aligned} P \mid b = c &\equiv P[c/b] \mid b = c \\ &\equiv P[b/c] \mid b = c \end{aligned}$$

which trivializes subtyping:

- $\Gamma \vdash b = c$ implies $(\Gamma \vdash P \Rightarrow \Gamma \vdash P[c/b])$,
- which suggests $\Gamma(c) \leq \Gamma(b)$ (and thus $\Gamma(b) = \Gamma(c)$),
- hence $(\Gamma \vdash \overline{a}\langle b \rangle \mid a\langle c \rangle) \Rightarrow \Gamma(b) = \Gamma(c)$,
- so in $\dfrac{\Gamma(a) \stackrel{?}{=} oT_1 \qquad \Gamma(b) \stackrel{?}{=} T_2}{\Gamma \vdash \overline{a}\langle b \rangle}$, $\Gamma(a)$ should "know" T_2.

Intuition: there is no subtyping as we know it.

Formally: **impossibility result**.

Impossibility result

If we have

1. free inputs and outputs $a\langle b\rangle$ and $\overline{a}\langle b\rangle$,
2. $(\boldsymbol{\nu} c)(\overline{a}\langle b\rangle.P \mid a\langle c\rangle.Q) \to^* (P \mid Q)[b/c]$,
3. judgments like $\Gamma \vdash P$,
4. compositionality ($\Gamma \vdash P$; $\Gamma \vdash Q \Rightarrow \Gamma \vdash P \mid Q$ et $\Gamma \backslash a \vdash (\nu a)P$),
5. weakening, strengthening,
6. stability by injective name substitution,
7. narrowing : $\Gamma, a : T \vdash P$ and $U \leq T$ implies $\Gamma, a : U \vdash P$

then there are some Γ, P, P' such that :

$$\Gamma \vdash P \wedge P \to^* P' \wedge \Gamma \nvdash P'$$