Advanced Process Calculi

Lecture 4: higher-order psi-calculi

Copenhagen, August 2013

Joachim Parrow

Psi-calculi

T	(Data) Terms	M, N
A	Assertions	Ψ, Ψ'
C	Conditions	φ, φ'

Channel Object Pattern Test (aka guard)

$\overline{M}\, N.P$	Output
$\underline{M}(\lambda\widetilde{x})N.P$	Input
$\mathbf{case}\ \varphi_1 : P_1 \,[\!]\, \cdots \,[\!]\, \varphi_n : P_n$	Case
$(\nu a)P$	Restriction
$P \mid Q$	Parallel
$!P$	Replication
$(\!\vert \Psi \vert\!)$	Assertion

To mean tell Ψ

How to cook a Psi-calculus

1. Define names, data terms, assertions and conditions
   can be absolutely anything

How to cook a Psi-calculus

1. Define names, data terms, assertions and conditions
   can be absolutely anything

2. Define support and substitution
   must satisfy the axioms

How to cook a Psi-calculus

1. Define names, data terms, assertions and conditions
   can be absolutely anything

2. Define support and substitution
   must satisfy the axioms

3. Define the morphisms $\dot{\leftrightarrow}$, $\otimes$, $\mathbf{1}$, $\vdash$
   must satisfy the requisites

How to cook a Psi-calculus

1. Define names, data terms, assertions and conditions
   can be absolutely anything

2. Define support and substitution
   must satisfy the axioms

3. Define the morphisms $\dot{\leftrightarrow}$, $\otimes$, $\mathbf{1}$, $\vdash$
   must satisfy the requisites

Compositional semantics
Algebraic laws
Bisimulation theory

Machine certified

- **Can capture**
  - **Applied pi-calculus** (Abadi, Fournet 2001)
  - **Explicit fusion calculus** (Wischik, Gardner 2005)
  - **Concurrent constraint pi** (Buscemi, Montanari 2007)
  - **Polyadic synchronization** (Carbone, Maffeis 2003)
  - **Pattern matching** and **higher order values** (Various)
- **And moreover**
  - **Higher-order concurrent constraints**
  - **Algebraic operators on communication channels**

For every application there is a suitable psi-calculus?

Of course not

Current extensions

- **Higher-order** psi: Agents can be sent around as data objects.
- **Broadcast** psi: an output action can be received by many inputs
- **Sorted** psi: A sort system regulates what can be substituted, sent on channels etc
- **Priority** psi: actions carry priorties, lower are preempted by higher

Psi and sorts

Problem: Substitution must be total, and all terms can act as both subjects and objects

Psi and sorts

Problem: Substitution must be total, and all terms can act as both subjects and objects

Effect: over-expressiveness. It is difficult to restrict a calculus to avoid useless agents, aka **junk**

Psi and sorts

Example: polyadic pi. Objects of prefixes are name tuples, as in $a(x_1,\ldots,x_n).P$

Psi and sorts

Example: polyadic pi. Objects of prefixes are name tuples, as in $a(x_1,\ldots,x_n).P$

Corresponding psi-calculus: let data terms be name tuples. We then also get:

$$\overline{(x_1, x_n)}y \,.\, P$$

Tuples as channels

Psi and sorts

Example: polyadic pi. Objects of prefixes are name tuples, as in $a(x_1,\ldots,x_n).P$

Corresponding psi-calculus: let data terms be name tuples. We then also get:

$$\overline{(x_1, x_n)}y \,.\, P$$

Tuples as channels

$$(x, y, z)[y := (u, w)] \quad \stackrel{?}{=} \quad (x, (u, w), z)$$

Nested tuples (aka trees)

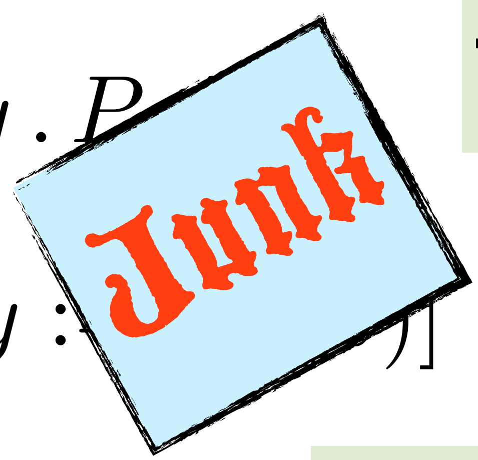
Psi and sorts

Example: polyadic pi. Objects of prefixes are name tuples, as in $a(x_1,\ldots,x_n).P$

Corresponding psi-calculus: let data terms be name tuples. We then also get:

$$\overline{(x_1, x_n)}y\,.\,P$$

Tuples as channels



$$(x, y, z)[y : \ldots)] \quad \stackrel{?}{=} \quad (x, (u, w), z)$$

Nested tuples (aka trees)

Dealing with junk

Allow it, using (ad hoc) invariants
to ensure it never arises

or

Disallow it, using a formal sort system

Sorts

Assume a set of sorts S

Names and data terms have unique sort

Sorts

Assume a set of sorts $\mathcal{S}$

Names and data terms have unique sort

$$
\begin{array}{llll}
\underline{\propto} & \subseteq & \mathcal{S} \times \mathcal{S} & \text{Can be used to receive} \\
\overline{\propto} & \subseteq & \mathcal{S} \times \mathcal{S} & \text{Can be used to send} \\
\prec & \subseteq & \mathcal{S} \times \mathcal{S} & \text{Can be substituted by}
\end{array}
$$

Sorts

Assume a set of sorts S

Names and data terms have unique sort

$$
\begin{array}{llll}
\underline{\propto} & \subseteq & \mathcal{S} \times \mathcal{S} & \text{Can be used to receive} \\
\overline{\propto} & \subseteq & \mathcal{S} \times \mathcal{S} & \text{Can be used to send} \\
\prec & \subseteq & \mathcal{S} \times \mathcal{S} & \text{Can be substituted by}
\end{array}
$$

Well-formedness criteria in writing agents, eg in

$\overline{M} N \,.\, P$ **requires** $\mathrm{sort}(M) \mathrel{\overline{\propto}} \mathrm{sort}(N)$

Sorts

Assume a set of sorts S

Names and data terms have unique sort

$$
\begin{array}{llll}
\underline{\propto} & \subseteq & \mathcal{S} \times \mathcal{S} & \text{Can be used to receive} \\
\overline{\propto} & \subseteq & \mathcal{S} \times \mathcal{S} & \text{Can be used to send} \\
\prec & \subseteq & \mathcal{S} \times \mathcal{S} & \text{Can be substituted by}
\end{array}
$$

Well-formedness criteria in writing agents, eg in

$\overline{M}N\,.\,P$ **requires** $\mathrm{sort}(M)\ \overline{\propto}\ \mathrm{sort}(N)$

Input rule: substitution conforms to $\prec$

Example: polyadic pi

$$\text{Names } \mathcal{N} = a, b, \ldots$$

$$\mathbf{T} = \mathcal{N} \uplus \mathcal{N}^*$$

$$\mathcal{S} = \{\mathbf{chan}, \mathbf{tup}\}$$

$$\textsc{sort}(a) = \mathbf{chan}$$

$$\textsc{sort}(\tilde{a}) = \mathbf{tup}$$

$$\mathbf{chan} \mathrel{\overline{\propto}} \mathbf{tup}$$

$$\mathbf{chan} \mathrel{\underline{\propto}} \mathbf{tup}$$

$$\mathbf{chan} \prec \mathbf{chan}$$

Higher-order psi

Higher-order

logic: quantify over predicates

functions: can have functions as parameters

process calculi: agents can be transmitted in communications

Higher-order pi

$\overline{a}P \,.\, Q$	Send the agent P along a and continue as Q
$a(X) \,.\, R$	Receive for the agent variable X along a and continue as R

New syntactic category!

Higher-order pi

$\overline{a}P \,.\, Q$ Send the agent P along a and continue as Q

$a(X) \,.\, R$ Receive for the agent variable X along a and continue as R

New syntactic category!

$$\overline{a}P \,.\, Q \mid a(X) \,.\, R \xrightarrow{\tau} Q \mid R[X := P]$$

Higher-order substitution!

Higher-order pi

$\overline{a}P\,.\,Q$	Send the agent P along a and continue as Q
$a(X)\,.\,R$	Receive for the agent variable X along a and continue as R

New syntactic category!

$$\overline{a}P\,.\,Q \mid a(X)\,.\,R \xrightarrow{\tau} Q \mid R[X := P]$$

Higher-order substitution!

Eg

$$\overline{a}P\,.\,Q \mid a(X)\,.\,(R' \mid X)$$

Variable used as agent

Higher-order pi

$\overline{a}P\,.\,Q$ Send the agent P along a and continue as Q

$a(X)\,.\,R$ Receive for the agent variable X along a and continue as R

New syntactic category!

$$\overline{a}P\,.\,Q \mid a(X)\,.\,R \xrightarrow{\tau} Q \mid R[X := P]$$

Higher-order substitution!

Eg

$$\overline{a}P\,.\,Q \mid a(X)\,.\,(R' \mid X) \xrightarrow{\tau} Q \mid R' \mid P$$

Variable used as agent

Higher-order psi already?

$\mathbf{T}$	(Data) Terms	M, N
$\mathbf{A}$	Assertions	Ψ, Ψ'
$\mathbf{C}$	Conditions	φ, φ'

Terms can be any nominal set

Agents constitute a nominal set!

Higher-order psi already?

$\mathbf{T}$	(Data) Terms	M, N
$\mathbf{A}$	Assertions	Ψ, Ψ'
$\mathbf{C}$	Conditions	φ, φ'

Terms can be any nominal set

Agents constitute a nominal set!

So we choose $\mathbf{T}$ = the set of agents!

Higher-order psi already?

$\mathbf{T}$	(Data) Terms	M, N
$\mathbf{A}$	Assertions	Ψ, Ψ'
$\mathbf{C}$	Conditions	φ, φ'

Terms can be any nominal set

Agents constitute a nominal set!

So we choose $\mathbf{T}$ = the set of agents!

$$\overline{M}\,P\,.\,Q \mid a(x)\,.\,R \xrightarrow{\tau} Q \mid R[x := P]$$

R receives the agent P, substituting x

$$\overline{M}P \,.\, Q \mid a(x) \,.\, R \xrightarrow{\tau} Q \mid R[x := P]$$

Problem: How can R get to 'execute' the newly received P?

Where can x occur in R?

$$\overline{M}P\,.\,Q \mid a(x)\,.\,R \xrightarrow{\tau} Q \mid R[x := P]$$

Problem: How can R get to 'execute' the newly received P?

Where can x occur in R?

$\overline{M}\ N.P$	Output
$\underline{M}(\lambda\widetilde{x})N.P$	Input
$\mathbf{case}\ \varphi_1 : P_1\ [\!]\ \cdots\ [\!]\ \varphi_n : P_n$	Case
$(\nu a)P$	Restriction
$P \mid Q$	Parallel
$!P$	Replication
$(\!\mid \Psi \mid\!)$	Assertion

$$\overline{M}P\,.\,Q \mid a(x)\,.\,R \xrightarrow{\tau} Q \mid R[x := P]$$

Problem: How can R get to 'execute' the newly received P?

Where can x occur in R?

$\overline{M}\ N.P$	Output
$\underline{M}(\lambda\widetilde{x})N.P$	Input
$\textbf{case}\ \varphi_1 : P_1 \,[]\, \cdots \,[]\, \varphi_n : P_n$	Case
$(\nu a)P$	Restriction
$P \mid Q$	Parallel
$!P$	Replication
$(\!\mid \Psi \mid\!)$	Assertion

x can only occur in data terms, assertions and conditions :(

The rub

The rub

- Psi **already** can accommodate agents as data values

The rub

- Psi **already** can accommodate agents as data values
- Psi **lacks** a notion of higher order variable that can stand for agents and be substituted.

The rub

- Psi **already** can accommodate agents as data values
- Psi **lacks** a notion of higher order variable that can stand for agents and be substituted.
- **Introducing** that is more complicated than you would think.

The rub

- Psi **already** can accommodate agents as data values
- Psi **lacks** a notion of higher order variable that can stand for agents and be substituted.
- **Introducing** that is more complicated than you would think.
- There is a way that is both **easier and more general**!

Clauses

A **clause** is of the form $\boxed{M \Leftarrow P}$

Means that the data term M can be used as a **handle** for the agent P

The handle can be **invoked** in the new agent form $\boxed{\textbf{run}\ M}$

Intuition

Assume a clause $M \Leftarrow P$

Sending P along a is then $\overline{a}M \,.\, Q$

Receiving a process along a is $a(x) \,.\, (R \mid \mathbf{run}\ x)$

Intuition

Assume a clause $M \Leftarrow P$

Sending P along a is then $\overline{a}M\,.\,Q$

Receiving a process along a is $a(x)\,.\,(R \mid \mathbf{run}\ x)$

$$\overline{a}M\,.\,Q \mid a(x)\,.\,(R \mid \mathbf{run}\ x) \xrightarrow{\tau} Q \mid R \mid \mathbf{run}\ M$$

Ho-pi vs HO-psi

pi

$$a(X)\,.\,(R \mid X)$$

New kind of variable
New kind of substitution

psi

$$a(x)\,.\,(R \mid \mathbf{run}\ x)$$

New syntactic construct
(Notion of clause)

Where do clauses live?

One possibility: introduce a new instance parameter as a set of clauses

Where do clauses live?

One possibility: introduce a new instance parameter as a set of clauses

Again, there is an easier and more general way!

Hint: transitions always depend on environmental assertions.

$$\Psi \rhd P \xrightarrow{\alpha} P'$$

Entailed by assertions

A clause can be **entailed** by assertions, as in

$$\Psi \vdash M \Leftarrow P$$

Formally, clauses can be a **subset of the conditions**

Clearer terminology: extend $\vdash$ to also relate assertions with conditions **and clauses**

Semantics of **run**

$$\frac{\Psi \vdash M \Leftarrow P \qquad \Psi \;\rhd\; P \;\xrightarrow{\alpha}\; P'}{\Psi \;\rhd\; \mathbf{run}\, M \;\xrightarrow{\alpha}\; P'}$$

Universal clauses

In some applications it might be sufficient with **universal** clauses, entailed by **all** assertions

Example: universal clauses can express recursive definitions!

$$\forall\Psi. \quad \Psi \vdash M \Leftarrow a(x) \,.\, \overline{b}x \,.\, \mathbf{run}\ M$$

cf pi-calculus

$$A \Leftarrow a(x) \,.\, \overline{b}x \,.\, A$$

Local clauses

Clauses are entailed by assertions

Handles may **contain names** and be **scoped**

$$z \in \mathrm{n}(M), \quad \Psi \vdash M \Leftarrow a(x) \,.\, \overline{b}x \,.\, \mathbf{run}\; M$$

$$P \mid (\nu z)((\!|\Psi|\!) \mid Q)$$

Here Q but not P can use $\mathbf{run}\; M$

Mobile clauses

$$z \in \mathrm{n}(M), \quad \Psi \vdash M \Leftarrow a(x) \,.\, \overline{b}x \,.\, \mathbf{run}\ M$$

$$P \mid (\nu z)((\!|\Psi|\!) \mid Q)$$

The ability to use $\mathbf{run}\ M$ can be transmitted by Q by sending z

Or by sending M itself

In both cases extruding z

Multiple clauses

Nothing prevents the **same** handle to occur in **many** clauses

$$M \Leftarrow P_1$$
$$M \Leftarrow P_2$$
$$\vdots$$

The rule for **run** M is applicable to all

Nondeterminism (can represent +)

Requirement on clauses

In any clause $M \Leftarrow P$

we require $\mathrm{n}(P) \subseteq \mathrm{n}(M)$

ie, the support of the handle contains at least the support of the agent it represents.

Motivation: otherwise scope extension fails

$$\Psi \vdash M \Leftarrow \overline{b}N \, . \, \mathbf{0}$$

Motivation: otherwise scope extension fails

$$\Psi \vdash M \Leftarrow \overline{b}N \,.\, \mathbf{0}$$

$b \# M$ violating $\mathrm{n}(\overline{b}N \,.\, 0) \subseteq \{b\} \subseteq \mathrm{n}(M)$

Motivation: otherwise scope extension fails

$$\Psi \vdash M \Leftarrow \overline{b}N\,.\,\mathbf{0}$$

$b \# M$ violating $\mathrm{n}(\overline{b}N\,.\,0) \subseteq \{b\} \subseteq \mathrm{n}(M)$

$b \# \mathbf{run}\ M$

Motivation: otherwise scope extension fails

$$\Psi \vdash M \Leftarrow \overline{b}N \, . \, \mathbf{0}$$

$b \# M$ violating $\mathrm{n}(\overline{b}N \, . \, 0) \subseteq \{b\} \subseteq \mathrm{n}(M)$

$$b \# \mathbf{run} \ M$$

$$(\nu b)\mathbf{run} \ M \sim \mathbf{run} \ M$$

Motivation: otherwise scope extension fails

$$\Psi \vdash M \Leftarrow \overline{b}N\,.\,\mathbf{0}$$

$b \# M$ violating $\mathrm{n}(\overline{b}N\,.\,0) \subseteq \{b\} \subseteq \mathrm{n}(M)$

$$b \# \mathbf{run}\ M$$

$$(\nu b)\mathbf{run}\ M \sim \mathbf{run}\ M$$

$$\Psi \rhd \mathbf{run}\ M \xrightarrow{\overline{b}N} \cdots$$

$\Psi \rhd (\nu b)\mathbf{run}\ M \cdots$ has no transition

Motivation: otherwise scope extension fails

$$\Psi \vdash M \Leftarrow \overline{b}N\,.\,\mathbf{0}$$

$b \# M$ violating $\mathrm{n}(\overline{b}N\,.\,0) \subseteq \{b\} \subseteq \mathrm{n}(M)$

$$b \# \mathbf{run}\ M$$

$$(\nu b)\mathbf{run}\ M \sim \mathbf{run}\ M$$

$$\Psi \rhd \mathbf{run}\ M \xrightarrow{\overline{b}N}$$

Contradiction

$\Psi \rhd (\nu b)\mathbf{run}\ M \cdots$ has no transition

Example: stacks

Let assertions be sets of parametrised clauses

$$M(\lambda\tilde{x})N \Leftarrow P$$

$$M(\lambda\tilde{x})N \Leftarrow P \in \Psi \quad \Longrightarrow \quad \Psi \vdash M\langle N[\tilde{x} := \tilde{L}]\rangle \Leftarrow P[\tilde{x} := \tilde{L}]$$

$$\begin{array}{l} \textsc{Stack}(\lambda x)x \Leftarrow \underline{Push}(\lambda y)y\,.\,\mathbf{run}\ \textsc{Stack}\langle \mathrm{cons}(y,x)\rangle \\ \textsc{Stack}(\lambda x,y)\mathrm{cons}(x,y) \Leftarrow \overline{Pop}\ x\,.\,\mathbf{run}\ \textsc{Stack}\langle y\rangle \end{array}$$

$$\textsc{Stack}(\lambda x)x \Leftarrow \underline{Push}(\lambda y)y\,.\,\mathbf{run}\ \textsc{Stack}\langle \mathrm{cons}(y,x)\rangle$$

$$\textsc{Stack}(\lambda x,y)\mathrm{cons}(x,y) \Leftarrow \overline{Pop}\ x\,.\,\mathbf{run}\ \textsc{Stack}\langle y\rangle$$

$$\textsc{Stack}(\lambda x)x \Leftarrow \underline{Push}(\lambda y)y\,.\,\mathbf{run}\ \textsc{Stack}\langle \mathrm{cons}(y,x)\rangle$$
$$\textsc{Stack}(\lambda x,y)\mathrm{cons}(x,y) \Leftarrow \overline{Pop}\ x\,.\,\mathbf{run}\ \textsc{Stack}\langle y\rangle$$

$$\Psi \vdash \textsc{Stack}\langle \mathrm{nil}\rangle \Leftarrow \underline{Push}(\lambda y)y\,.\,\mathbf{run}\ \textsc{Stack}\langle \mathrm{cons}(y,\mathrm{nil})\rangle$$

$$\textsc{Stack}(\lambda x)x \Leftarrow \underline{\mathit{Push}}(\lambda y)y \,.\, \mathbf{run}\ \textsc{Stack}\langle \mathrm{cons}(y,x)\rangle$$
$$\textsc{Stack}(\lambda x,y)\mathrm{cons}(x,y) \Leftarrow \overline{\mathit{Pop}}\ x \,.\, \mathbf{run}\ \textsc{Stack}\langle y\rangle$$

$$\Psi \vdash \textsc{Stack}\langle \mathrm{nil}\rangle \Leftarrow \underline{\mathit{Push}}(\lambda y)y \,.\, \mathbf{run}\ \textsc{Stack}\langle \mathrm{cons}(y,\mathrm{nil})\rangle$$

$$\Psi \ \rhd\ \mathbf{run}\ \textsc{Stack}\langle \mathrm{nil}\rangle \ \xrightarrow{\underline{\mathit{Push}}\, M}\ \mathbf{run}\ \textsc{Stack}\langle \mathrm{cons}(M,\mathrm{nil})\rangle$$

$$\textsc{Stack}(\lambda x)x \Leftarrow \underline{\mathit{Push}}(\lambda y)y\,.\,\mathbf{run}\ \textsc{Stack}\langle \mathrm{cons}(y,x)\rangle$$
$$\textsc{Stack}(\lambda x,y)\mathrm{cons}(x,y) \Leftarrow \overline{\mathit{Pop}}\ x\,.\,\mathbf{run}\ \textsc{Stack}\langle y\rangle$$

$$\Psi \vdash \textsc{Stack}\langle \mathrm{nil}\rangle \Leftarrow \underline{\mathit{Push}}(\lambda y)y\,.\,\mathbf{run}\ \textsc{Stack}\langle \mathrm{cons}(y,\mathrm{nil})\rangle$$

$$\Psi \ \rhd\ \mathbf{run}\ \textsc{Stack}\langle \mathrm{nil}\rangle \xrightarrow{\underline{\mathit{Push}}\,M} \mathbf{run}\ \textsc{Stack}\langle \mathrm{cons}(M,\mathrm{nil})\rangle$$

$$\Psi \ \rhd\ \mathbf{run}\ \textsc{Stack}\langle \mathrm{cons}(M,\mathrm{nil})\rangle \xrightarrow{\underline{\mathit{Push}}\,M'} \mathbf{run}\ \textsc{Stack}\langle \mathrm{cons}(M',\mathrm{cons}(M,\mathrm{nil}))\rangle$$

$$\textsc{Stack}(\lambda x)x \Leftarrow \underline{Push}(\lambda y)y\,.\,\mathbf{run}\ \textsc{Stack}\langle\mathrm{cons}(y,x)\rangle$$
$$\textsc{Stack}(\lambda x,y)\mathrm{cons}(x,y) \Leftarrow \overline{Pop}\ x\,.\,\mathbf{run}\ \textsc{Stack}\langle y\rangle$$

$$\Psi \vdash \textsc{Stack}\langle\mathrm{nil}\rangle \Leftarrow \underline{Push}(\lambda y)y\,.\,\mathbf{run}\ \textsc{Stack}\langle\mathrm{cons}(y,\mathrm{nil})\rangle$$

$$\Psi\ \triangleright\ \mathbf{run}\ \textsc{Stack}\langle\mathrm{nil}\rangle \xrightarrow{\underline{Push}\,M} \mathbf{run}\ \textsc{Stack}\langle\mathrm{cons}(M,\mathrm{nil})\rangle$$

$$\Psi\ \triangleright\ \mathbf{run}\ \textsc{Stack}\langle\mathrm{cons}(M,\mathrm{nil})\rangle \xrightarrow{\underline{Push}\,M'} \mathbf{run}\ \textsc{Stack}\langle\mathrm{cons}(M',\mathrm{cons}(M,\mathrm{nil}))\rangle$$

$$\Psi\ \triangleright\ \mathbf{run}\ \textsc{Stack}\langle\mathrm{cons}(M,\mathrm{nil})\rangle \xrightarrow{\overline{Pop}\ M} \mathbf{run}\ \textsc{Stack}\langle\mathrm{nil}\rangle$$

$$\textsc{Stack}(\lambda x)x \Leftarrow \underline{Push}(\lambda y)y\,.\,\mathbf{run}\ \textsc{Stack}\langle \mathrm{cons}(y,x)\rangle$$
$$\textsc{Stack}(\lambda x,y)\mathrm{cons}(x,y) \Leftarrow \overline{Pop}\ x\,.\,\mathbf{run}\ \textsc{Stack}\langle y\rangle$$

$$\textsc{Stack}(\lambda x)x \Leftarrow \underline{Push}(\lambda y)y\,.\,\mathbf{run}\ \textsc{Stack}\langle \mathrm{cons}(y,x)\rangle$$
$$\textsc{Stack}(\lambda x,y)\mathrm{cons}(x,y) \Leftarrow \overline{Pop}\ x\,.\,\mathbf{run}\ \textsc{Stack}\langle y\rangle$$

A stack factory

$$!\overline{a}\ \textsc{Stack}\,.\,\mathbf{0}$$

$$\textsc{Stack}(\lambda x)x \Leftarrow \underline{Push}(\lambda y)y\,.\,\mathbf{run}\ \textsc{Stack}\langle \mathrm{cons}(y,x)\rangle$$
$$\textsc{Stack}(\lambda x,y)\mathrm{cons}(x,y) \Leftarrow \overline{Pop}\ x\,.\,\mathbf{run}\ \textsc{Stack}\langle y\rangle$$

A stack factory

$$!\overline{a}\ \textsc{Stack}\,.\,\mathbf{0}$$

But this uses the **same** push and pop channels for **all** stacks

$$\textsc{Stack}(\lambda x)x \Leftarrow \underline{Push}(\lambda y)y\,.\,\mathbf{run}\;\textsc{Stack}\langle \mathrm{cons}(y,x)\rangle$$
$$\textsc{Stack}(\lambda x,y)\mathrm{cons}(x,y) \Leftarrow \overline{Pop}\;x\,.\,\mathbf{run}\;\textsc{Stack}\langle y\rangle$$

A stack factory

$$!\overline{a}\;\textsc{Stack}\,.\,\mathbf{0}$$

But this uses the **same** push and pop channels for **all** stacks

Alternative:

$$\textsc{Stack}(\lambda i,o,x)i,o,x \Leftarrow \underline{i}(\lambda y)y\,.\,\mathbf{run}\;\textsc{Stack}\langle i,o,\mathrm{cons}(y,x)\rangle$$
$$\textsc{Stack}(\lambda i,o,x,y)i,o,\mathrm{cons}(x,y) \Leftarrow \overline{o}\;x\,.\,\mathbf{run}\;\textsc{Stack}\langle i,o,y\rangle$$

$$\textsc{StackStart} \Leftarrow c(Push, Pop)\,.\,\mathbf{run}\;\textsc{Stack}\langle (Push, Pop, \mathrm{nil})\rangle$$

Canonical HO-calculi

The stack example can be generalised considerably

Thm *(paraphrased, see paper for details)* Any ordinary psi-calculus of nontrivial expressiveness can be systematically raised to a higher-order calculus by letting assertions be sets of parametrised clauses.

Representing replication

In HO-pi we can **encode replication**.
Can we do that in HO-psi?

Yes - at least in enough expressive psi-calculi

Representing replication

In HO-pi we can **encode replication**.
Can we do that in HO-psi?

Yes - at least in enough expressive psi-calculi

$\Psi^{M \Leftarrow P}$ is a **characteristic assertion** for M and P if

1. $\Psi \vdash M \Leftarrow Q$ implies $\mathrm{n}(M) \subseteq \mathrm{n}(\Psi)$
2. $\Psi \otimes \Psi^{M \Leftarrow P} \vdash \xi \quad \text{iff} \quad (\xi = M \Leftarrow P \;\vee\; \Psi \vdash \xi)$
3. $\mathrm{n}(\Psi^{M \Leftarrow P}) = \mathrm{n}(M)$

1. $\Psi \vdash M \Leftarrow Q$ implies $\mathrm{n}(M) \subseteq \mathrm{n}(\Psi)$

2. $\Psi \otimes \Psi^{M \Leftarrow P} \vdash \xi \quad \text{iff} \quad (\xi = M \Leftarrow P \quad \vee \quad \Psi \vdash \xi)$

3. $\mathrm{n}(\Psi^{M \Leftarrow P}) = \mathrm{n}(M)$

1. $\Psi \vdash M \Leftarrow Q$ implies $\mathrm{n}(M) \subseteq \mathrm{n}(\Psi)$

2. $\boxed{\Psi \otimes \Psi^{M \Leftarrow P} \vdash \xi \quad \text{iff} \quad (\xi = M \Leftarrow P \quad \vee \quad \Psi \vdash \xi)}$

3. $\mathrm{n}(\Psi^{M \Leftarrow P}) = \mathrm{n}(M)$

This means that the only effect of the characteristic assertion is to entail the clause $M \Leftarrow P$

1. $\Psi \vdash M \Leftarrow Q$ implies $\mathrm{n}(M) \subseteq \mathrm{n}(\Psi)$

2. $\Psi \otimes \Psi^{M \Leftarrow P} \vdash \xi \quad \text{iff} \quad (\xi = M \Leftarrow P \quad \vee \quad \Psi \vdash \xi)$

3. $\mathrm{n}(\Psi^{M \Leftarrow P}) = \mathrm{n}(M)$

This means that the only effect of the characteristic assertion is to entail the clause $M \Leftarrow P$

Thm characteristic assertions always exist in canonical higher-order calculi

Representing replication

Let a be fresh and $a \in \mathrm{n}(M)$

Let $\Psi^{M \Leftarrow P \mid \mathbf{run}\ M}$ be characteristic for M and $P \mid \mathbf{run}\ M$

Then $\boxed{!P \stackrel{\bullet}{\sim} (\nu a)(\mathbf{run}\ M \mid (\!|\Psi^{M \Leftarrow P \mid \mathbf{run}\ M}|\!))}$

Representing replication

Let a be fresh and $a \in \mathrm{n}(M)$
Let $\Psi^{M \Leftarrow P \,|\, \mathbf{run}\ M}$ be characteristic for M and $P \mid \mathbf{run}\ M$

Then $\boxed{!P \dot{\sim} (\nu a)(\mathbf{run}\ M \mid (\!|\Psi^{M \Leftarrow P \,|\, \mathbf{run}\ M}|\!))}$

Idea:

$$\mathbf{run}\ M \mid (\!|\Psi^{M \Leftarrow P \,|\, \mathbf{run}\ M}|\!)$$

$$(P \mid \mathbf{run}\ M) \mid (\!|\Psi^{M \Leftarrow P \,|\, \mathbf{run}\ M}|\!)$$

$$(P \mid (P \mid \mathbf{run}\ M)) \mid (\!|\Psi^{M \Leftarrow P \,|\, \mathbf{run}\ M}|\!)$$

$$\vdots$$

By the semantic rules these all have the **same** transitions!

Representing replication

Let a be fresh and $a \in \mathrm{n}(M)$
Let $\Psi^{M \Leftarrow P \mid \mathbf{run}\ M}$ be characteristic for M and $P \mid \mathbf{run}\ M$

Then $\boxed{!P \stackrel{\centerdot}{\sim} (\nu a)(\mathbf{run}\ M \mid (\!| \Psi^{M \Leftarrow P \mid \mathbf{run}\ M} |\!))}$

Idea:

$$\mathbf{run}\ M \mid (\!| \Psi^{M \Leftarrow P \mid \mathbf{run}\ M} |\!)$$
$$(P \mid \mathbf{run}\ M) \mid (\!| \Psi^{M \Leftarrow P \mid \mathbf{run}\ M} |\!)$$
$$(P \mid (P \mid \mathbf{run}\ M)) \mid (\!| \Psi^{M \Leftarrow P \mid \mathbf{run}\ M} |\!)$$
$$\vdots$$

By the semantic rules these all have the **same** transitions!

Why the (νa)?

Representing replication

Let a be fresh and $a \in \mathrm{n}(M)$
Let $\Psi^{M \Leftarrow P \,|\, \mathbf{run}\ M}$ be characteristic for M and $P \mid \mathbf{run}\ M$

Then $\boxed{!P \dot{\sim} (\nu a)(\mathbf{run}\ M \mid (\!|\Psi^{M \Leftarrow P \,|\, \mathbf{run}\ M}|\!))}$

Idea:

$$\mathbf{run}\ M \mid (\!|\Psi^{M \Leftarrow P \,|\, \mathbf{run}\ M}|\!)$$
$$(P \mid \mathbf{run}\ M) \mid (\!|\Psi^{M \Leftarrow P \,|\, \mathbf{run}\ M}|\!)$$
$$(P \mid (P \mid \mathbf{run}\ M)) \mid (\!|\Psi^{M \Leftarrow P \,|\, \mathbf{run}\ M}|\!)$$
$$\vdots$$

By the semantic rules these all have the **same** transitions!

Otherwise an environment might bestow **additional** clauses with M

Representing replication

Let a be fresh and $a \in \mathrm{n}(M)$

Let $\Psi^{M \Leftarrow P \mid \mathbf{run}\ M}$ be characteristic for M and $P \mid \mathbf{run}\ M$

Then $\boxed{!P \dot{\sim} (\nu a)(\mathbf{run}\ M \mid (\!|\Psi^{M \Leftarrow P \mid \mathbf{run}\ M}|\!))}$

Anyway, what is this in HO-Psi?

$$\mathbf{run}\ M \mid (\!|\Psi^{M \Leftarrow P \mid \mathbf{run}\ M}|\!)$$

$$(P \mid \mathbf{run}\ M) \mid (\!|\Psi^{M \Leftarrow P \mid \mathbf{run}\ M}|\!)$$

$$(P \mid (P \mid \mathbf{run}\ M)) \mid (\!|\Psi^{M \Leftarrow P \mid \mathbf{run}\ M}|\!)$$

$$\vdots$$

By the semantic rules these all have the **same** transitions!

Otherwise an environment might bestow **additional** clauses with M

Bisimulation

Formally, the **only** new aspect of higher-order psi is the inclusion of the **run** construct with accompanying rule!

No new syntactic categories, substitution etc

Just one more case when doing induction proofs

So perhaps we can just re-use the old definition!

R is a *bisimulation* if $R(\Psi, P, Q)$ implies

1. $R(\Psi, Q, P)$
2. $\forall \alpha . \mathrm{bn}(\alpha) \# Q, \Psi.$
   $\Psi \rhd P \xrightarrow{\alpha} P'$ implies $\Psi \rhd Q \xrightarrow{\alpha} Q'$ and $R(\Psi, P', Q')$
3. $\Psi \otimes \mathcal{F}(P) \simeq \Psi \otimes \mathcal{F}(Q)$
4. $\forall \Psi'.\ R(\Psi \otimes \Psi', P, Q)$

So perhaps we can just re-use the old definition!

R is a *bisimulation* if $R(\Psi, P, Q)$ implies

1. $R(\Psi, Q, P)$
2. $\forall\alpha.\mathrm{bn}(\alpha)\#Q, \Psi.$
   $\Psi \rhd P \xrightarrow{\alpha} P'$ implies $\Psi \rhd Q \xrightarrow{\alpha} Q'$ and $R(\Psi, P', Q')$
3. $\Psi \otimes \mathcal{F}(P) \simeq \Psi \otimes \mathcal{F}(Q)$
4. $\forall\Psi'.\ R(\Psi{\otimes}\Psi', P, Q)$

Thm: all laws and congruence properties that used to hold still hold!

Isabelle proof in approx one day!

So we are done?

So we are done?

Not quite.

So we are done?

Not quite.

In normal HO-calculi we would expect, as part of compositionality, that

$$P \stackrel{\bullet}{\sim} Q \quad \Rightarrow \quad \overline{a}P\,.\,R \stackrel{\bullet}{\sim} \overline{a}Q\,.\,R$$

So we are done?

Not quite.

In normal HO-calculi we would expect, as part of compositionality, that

$$P \mathrel{\dot\sim} Q \quad \Rightarrow \quad \overline{a}P\,.\,R \mathrel{\dot\sim} \overline{a}Q\,.\,R$$

In HO-psi the counterpart could be

$$P \mathrel{\dot\sim} Q \quad \Rightarrow \quad \overline{a}M\,.\,R \mid \Psi^{M \Leftarrow P} \;\mathrel{\dot\sim}\; \overline{a}M\,.\,R \mid \Psi^{M \Leftarrow Q}$$

You believe this?

$$P \stackrel{\bullet}{\sim} Q \quad \Rightarrow \quad \overline{a}M\,.\,R \mid \Psi^{M \Leftarrow P} \stackrel{\bullet}{\sim} \overline{a}M\,.\,R \mid \Psi^{M \Leftarrow Q}$$

R is a *bisimulation* if $R(\Psi, P, Q)$ implies

1. $R(\Psi, Q, P)$
2. $\forall \alpha.\mathrm{bn}(\alpha) \# Q, \Psi.$
   $\Psi \triangleright P \xrightarrow{\alpha} P'$ implies $\Psi \triangleright Q \xrightarrow{\alpha} Q'$ and $R(\Psi, P', Q')$
3. $\Psi \otimes \mathcal{F}(P) \simeq \Psi \otimes \mathcal{F}(Q)$
4. $\forall \Psi'.\ R(\Psi \otimes \Psi', P, Q)$

$$P \stackrel{\bullet}{\sim} Q \quad \Rightarrow \quad \overline{a}M\,.\,R \mid \Psi^{M \Leftarrow P} \;\stackrel{\bullet}{\sim}\; \overline{a}M\,.\,R \mid \Psi^{M \Leftarrow Q}$$

$\mathcal{R}$ is a *bisimulation* if $\mathcal{R}(\Psi, P, Q)$ implies

1. $\mathcal{R}(\Psi, Q, P)$
2. $\forall \alpha.\mathrm{bn}(\alpha) \# Q, \Psi.$
   $\Psi \rhd P \xrightarrow{\alpha} P'$ implies $\Psi \rhd Q \xrightarrow{\alpha} Q'$ and $\mathcal{R}(\Psi, P', Q')$
3. $\Psi \otimes \mathcal{F}(P) \simeq \Psi \otimes \mathcal{F}(Q)$
4. $\forall \Psi'.\ \mathcal{R}(\Psi \otimes \Psi', P, Q)$

$$\overline{a}M\,.\,R \mid \Psi^{M \Leftarrow P} \;\stackrel{\bullet}{\not\sim}\; \overline{a}M\,.\,R \mid \Psi^{M \Leftarrow Q}$$

since the frames are different

The culprit

$$\Psi \otimes \mathcal{F}(P) \simeq \Psi \otimes \mathcal{F}(Q)$$

$$\Psi \otimes \mathcal{F}(P) \vdash \varphi \quad \text{implies} \quad \Psi \otimes \mathcal{F}(Q) \vdash \varphi$$

The culprit

$$\Psi \otimes \mathcal{F}(P) \simeq \Psi \otimes \mathcal{F}(Q)$$

or in other words

$$\Psi \otimes \mathcal{F}(P) \vdash \varphi \quad \text{implies} \quad \Psi \otimes \mathcal{F}(Q) \vdash \varphi$$

The culprit

$$\Psi \otimes \mathcal{F}(P) \simeq \Psi \otimes \mathcal{F}(Q)$$

or in other words

$$\Psi \otimes \mathcal{F}(P) \vdash \varphi \quad \text{implies} \quad \Psi \otimes \mathcal{F}(Q) \vdash \varphi$$

Relax this condition so that for clauses it suffices with bisimilar ones!

$$\text{(a)} \;\; \forall \varphi \in \mathbf{C}. \quad \Psi \otimes \mathcal{F}(P) \vdash \varphi \;\Rightarrow\; \Psi \otimes \mathcal{F}(Q) \vdash \varphi$$

$$\text{(b)} \;\; \boxed{\begin{array}{l} \forall (M \Leftarrow P') \in \mathbf{Cl}. \quad \Psi \otimes \mathcal{F}(P) \vdash M \Leftarrow P' \;\Rightarrow \\ \quad \exists Q'.\, \Psi \otimes \mathcal{F}(Q) \vdash M \Leftarrow Q' \wedge (\mathbf{1}, P', Q') \in \mathcal{R} \end{array}}$$

A *strong HO-bisimulation* $\mathcal{R}$ is a ternary relation between assertions and pairs of agents such that $(\Psi, P, Q) \in \mathcal{R}$ implies all of

1. Static equivalence:

   (a) $\forall \varphi \in \mathbf{C}. \quad \Psi \otimes \mathcal{F}(P) \vdash \varphi \ \Rightarrow \ \Psi \otimes \mathcal{F}(Q) \vdash \varphi$

   (b) $\forall (M \Leftarrow P') \in \mathbf{Cl}. \quad \Psi \otimes \mathcal{F}(P) \vdash M \Leftarrow P' \ \Rightarrow$
   $\exists Q'.\ \Psi \otimes \mathcal{F}(Q) \vdash M \Leftarrow Q' \wedge (\mathbf{1}, P', Q') \in \mathcal{R}$

   The only new thing

2. Symmetry: $(\Psi, Q, P) \in \mathcal{R}$

3. Extension of arbitrary assertion: $\forall \Psi'.\ (\Psi \otimes \Psi', P, Q) \in \mathcal{R}$

4. Simulation: for all α, P' such that $\mathrm{bn}(\alpha) \# \Psi, Q$ there exists a Q' such that

$$\text{if } \Psi \vartriangleright P \xrightarrow{\alpha} P' \text{ then } \Psi \vartriangleright Q \xrightarrow{\alpha} Q' \wedge (\Psi, P', Q') \in \mathcal{R}$$

We define $\Psi \vartriangleright P \stackrel{\bullet}{\sim}^{\text{HO}} Q$ to mean that there exists a strong HO-bisimulation $\mathcal{R}$ such that $\Psi \vartriangleright P \mathrel{\mathcal{R}} Q$, and write $P \stackrel{\bullet}{\sim}^{\text{HO}} Q$ for $\mathbf{1} \vartriangleright P \stackrel{\bullet}{\sim}^{\text{HO}} Q$.

Thm

$$P \stackrel{\bullet}{\sim}^{\mathrm{HO}} Q \quad \Rightarrow \quad \Psi^{M \Leftarrow P} \stackrel{\bullet}{\sim}^{\mathrm{HO}} \Psi^{M \Leftarrow Q}$$

Thm

$$P \overset{\bullet}{\sim}{}^{\mathrm{HO}}\, Q \quad \Rightarrow \quad \Psi^{M \Leftarrow P} \ \overset{\bullet}{\sim}{}^{\mathrm{HO}}\ \Psi^{M \Leftarrow Q}$$

Thm: all laws and congruence properties that used to hold still holds!

The proof took forever to complete (several months)

$$\text{(a)}\ \forall \varphi \in \mathbf{C}.\quad \Psi \otimes \mathcal{F}(P) \vdash \varphi \ \Rightarrow\ \Psi \otimes \mathcal{F}(Q) \vdash \varphi$$

$$\text{(b)}\ \forall (M \Leftarrow P') \in \mathbf{Cl}.\quad \Psi \otimes \mathcal{F}(P) \vdash M \Leftarrow P' \ \Rightarrow$$
$$\exists Q'.\ \Psi \otimes \mathcal{F}(Q) \vdash M \Leftarrow Q' \wedge \boxed{(\mathbf{1}, P', Q') \in \mathcal{R}}$$

Why not instead $\boxed{(\Psi, P', Q') \in \mathcal{R}}$

$$\text{(a)}\ \ \forall \varphi \in \mathbf{C}. \quad \Psi \otimes \mathcal{F}(P) \vdash \varphi \ \Rightarrow\ \Psi \otimes \mathcal{F}(Q) \vdash \varphi$$

$$\text{(b)}\ \ \forall (M \Leftarrow P') \in \mathbf{Cl}. \quad \Psi \otimes \mathcal{F}(P) \vdash M \Leftarrow P' \ \Rightarrow$$
$$\exists Q'.\ \Psi \otimes \mathcal{F}(Q) \vdash M \Leftarrow Q' \wedge \boxed{(\mathbf{1}, P', Q') \in \mathcal{R}}$$

Why not instead $\boxed{(\Psi, P', Q') \in \mathcal{R}}$

With this we fail to prove compositionality
(still unknown if it holds)

Conclusion

Psi-calculi is a family of process calculi

Accommodates a wide variety of data terms, functions and properties etc, based on nominal sets

Meta-theory proved once and for all in Isabelle

Outlook

- Extensions
- Combinations
- Applications
- Tool support

Thank you for your attention