Binary Session Types

Hernán Melgratti

ICC University of Buenos Aires-Conicet

Syntax of Types

Session Types

$S, T ::=$		$\mathtt{end}$	terminated session
	$\mid$	$\mathtt{?}t\mathtt{.}S$	receive (input)
	$\mid$	$\mathtt{!}t\mathtt{.}S$	send (output)
	$\mid$	$\&[\mathtt{l}_i : T_i]_{i \in I}$	branch
	$\mid$	$\oplus[\mathtt{l}_i : T_i]_{i \in I}$	select
	$\mid$	$\mu X\mathtt{.}S$	recursive session type
	$\mid$	X	session type variable
$s, t ::=$		S	A session type
	$\mid$	$\mathtt{int}$, $\mathtt{bool}$	basic types
	$\mid$	[T]	shared channels
	$\mid$	...	other types

Duality (naive, classical definition)

$\overline{S}$ is the dual of S

$$
\begin{aligned}
\overline{\mathsf{end}} &= \mathsf{end} \\
\overline{?t.S} &= !t.\overline{S} \\
\overline{!t.S} &= ?t.\overline{S} \\
\overline{\&[\mathtt{l}_i : T_i]_{i\in I}} &= \oplus[\mathtt{l}_i : \overline{T_i}]_{i\in I} \\
\overline{\oplus[\mathtt{l}_i : T_i]_{i\in I}} &= \&[\mathtt{l}_i : \overline{T_i}]_{i\in I} \\
\overline{\mu X.T} &= \mu X.\overline{T} \\
\overline{X} &= X
\end{aligned}
$$

Important

Contractive types

Recursive types are required to be *contractive*, i.e., containing no subexpressions of the form $\mu X \,.\, \mu X_1 \,.\, \ldots \mu X_n \,.\, X$

Problem with the naive definition[1,2]

Communication of a recursive type

$S = \mu X . !X . X$
$\overline{S} = \mu X . ?X . X$

Mismatch: S sends S, but $\overline{S}$ receives $\overline{S}$

Quick fix

- Recursion variables only occur in tail position in a session type
- Results in several papers do not hold for recursive types occurring in non-tail position.
- Alternatively, infinite terms for recursive types (coinductive definition)

[1] Giovanni Bernardi and Matthew Hennessy: Using Higher-Order Contracts to Model Session Types (Extended Abstract). CONCUR 2014

[2] Simon J. Gay, Peter Thiemann and Vasco T. Vasconcelos. Duality of Session Types: The Final Cut. Places 2020.

Syntax of Processes

Polarities

$p ::= + \mid - \mid \epsilon$ Optional polarities

Values (more in general expressions)

$$
\begin{array}{lll}
v, w ::= & x^p, y^q, \ldots & \text{(polarised) variables } \mathcal{X} = \{x, y, \ldots\} \\
\mid & () & \text{unit value} \\
\mid & \mathtt{true}, \mathtt{false} & \text{boolean values} \\
\mid & \ldots & \text{expressions}
\end{array}
$$

Processes

$$
\begin{array}{lll}
P, Q ::= & 0 & \text{terminated process} \\
\mid & x^p ?(y{:}\mathtt{t}).P & \text{input} \\
\mid & x^p !v.P & \text{output} \\
\mid & x^p \triangleright [\mathtt{l}_i : P_i]_{i \in I} & \text{branch} \\
\mid & x^p \triangleleft \mathtt{l}.P & \text{select} \\
\mid & P|Q & \text{parallel composition} \\
\mid & (\nu x{:}S)P & \text{channel creation} \\
\mid & !P & \text{replication}
\end{array}
$$

Notation

- for a polarity p, we write $\overline{p}$ for the complementary endpoint

$$\overline{+} = - \qquad\qquad \overline{-} = + \qquad\qquad \overline{\epsilon} = \epsilon$$

- we identify x^{ϵ} with x

Free names

fn

$$
\begin{aligned}
\mathrm{fn}(\mathtt{true}) = \mathrm{fn}(\mathtt{false}) \;=\; \mathrm{fn}(()) \;&=\; \emptyset \\
\mathrm{fn}(x^p) &= \{x^p\} \\
\\
\mathrm{fn}(0) &= \emptyset \\
\mathrm{fn}(P|Q) &= \mathrm{fn}(Q) \cup \mathrm{fn}(P) \\
\mathrm{fn}(x^p\,\mathtt{?}(y{:}\mathtt{t})\,\mathtt{.}\,P) &= \{x^p\} \cup (\mathrm{fn}(P) \setminus \{y\}) \\
\mathrm{fn}(x^p\,\mathtt{!}\,v\,\mathtt{.}\,P) &= \{x^p\} \cup \mathrm{fn}(v) \cup \mathrm{fn}(P) \\
\mathrm{fn}(x^p \triangleright [\mathtt{l}_i : P_i]_{i\in I}) &= \{x^p\} \cup (\bigcup_i \mathrm{fn}(P_i)) \\
\mathrm{fn}(x^p \triangleleft \mathtt{l}\,\mathtt{.}\,P) &= \{x^p\} \cup \mathrm{fn}(P) \\
\mathrm{fn}((\nu x{:}S)P) &= \mathrm{fn}(P) \setminus \{x, x^+, x^-\} \\
\mathrm{fn}(\,\mathtt{!}\,P) &= \mathrm{fn}(P)
\end{aligned}
$$

Operational semantics

Given in terms of a *Labelled Transition System* (LTS) $(P, \longrightarrow)$ where

- $\longrightarrow \subseteq P \times (\mathcal{X} \cup \{\tau\}) \times (\mathcal{L} \cup \{-\}) \times P$

- $(P, \alpha, \mathtt{l}, Q) \in \longrightarrow$
  - means P evolves to Q after communicating the choice $\mathtt{l}$ on the session α
  - is abbreviated as $P \xrightarrow{\alpha, \mathtt{l}} Q$
- τ stands for a hidden session
- $-$ for no choice

Operational semantics

$$x^p\,!\,v\,.\,P \mid x^{\overline{p}}\,?(y{:}\mathtt{t})\,.\,Q \xrightarrow{x,-} P \mid Q\{v/y\} \quad \text{[R-Comm]}$$

Substitution

$$
\begin{array}{rcll}
x\{v/x\} &=& v & \\
x^p\{v/y\} &=& x^p & \text{if } x \neq y \\
\\
0\{v/y\} &=& 0 & \\
(P|Q)\{v/y\} &=& P\{v/y\}|Q\{v/y\} & \\
(x^p\,?(z{:}\mathtt{t})\,.\,P)\{v/y\} &=& x^p\{v/y\}\,?(z{:}\mathtt{t})\,.\,P\{v/y\} & \text{if } z \notin \mathrm{fn}(v) \cup \{y\} \\
(x^p\,!\,w\,.\,P)\{v/y\} &=& x^p\{v/y\}\,!\,w\{v/y\}\,.\,P\{v/y\} & \\
(x^p \triangleright [\mathtt{l}_i : P_i]_{i\in I})\{v/y\} &=& x^p\{v/y\} \triangleright [\mathtt{l}_i : P_i\{v/y\}]_{i\in I} & \\
(x^p \triangleleft \mathtt{l}\,.\,P)\{v/y\} &=& x^p\{v/y\} \triangleleft \mathtt{l}\,.\,P\{v/y\} & \\
((\nu x{:}S)P)\{v/y\} &=& (\nu x{:}S)P\{v/y\} & \text{if } x \notin \mathrm{fn}(v) \cup \{y\} \\
(!\,P)\{v/y\} &=& !\,(P\{v/y\}) &
\end{array}
$$

Operational semantics

$$x^p\,!\,v\,.\,P \mid x^{\overline{p}}?(y{:}\mathtt{t})\,.\,Q \xrightarrow{x,-} P \mid Q\{v/y\} \text{ [R-Comm]}$$

$$\frac{p \in \{+,-\} \qquad i \in I}{x^p \triangleleft \mathtt{l}_i\,.\,P \mid x^{\overline{p}} \triangleright [\mathtt{l}_j : Q_j]_{j\in I} \xrightarrow{x,\mathtt{l}_i} P \mid Q_i} \text{ [R-Select]}$$

$$\frac{P \xrightarrow{x,\mathtt{l}} P' \qquad S \xrightarrow{\mathtt{l}} T}{(\nu x{:}S)P \xrightarrow{\tau,-} (\nu x{:}T)P'} \text{ [R-NewS]}$$

Semantics of Types

$$?t\,.\,S \xrightarrow{-} S \qquad !t\,.\,S \xrightarrow{-} S$$

$$\&[\mathtt{l}_i : T_i]_{i\in I} \xrightarrow{\mathtt{l}_i} T_i \qquad \oplus[\mathtt{l}_i : T_i]_{i\in I} \xrightarrow{\mathtt{l}_i} T_i$$

$$\frac{S\{\mu X\,.\,S/X\} \xrightarrow{\beta} T}{\mu X\,.\,S \xrightarrow{\beta} T}$$

Operational semantics

$$x^{p}\,!\,v\,.\,P \mid x^{\overline{p}}?(y{:}\mathtt{t})\,.\,Q \xrightarrow{x,-} P \mid Q\{v/y\} \quad \text{[R-Comm]}$$

$$\frac{i \in I}{x^{p} \triangleleft \mathtt{l}_i\,.\,P \mid x^{\overline{p}} \triangleright [\mathtt{l}_j : Q_j]_{j\in I} \xrightarrow{x,\mathtt{l}_i} P \mid Q_i} \text{[R-Select]}$$

$$\frac{P \xrightarrow{x,\mathtt{l}} P' \qquad S \xrightarrow{x,\mathtt{l}} T}{(\nu x{:}S)P \xrightarrow{\tau,-} (\nu x{:}T)P'} \text{[R-NewS]}$$

$$\frac{P \xrightarrow{\alpha,\mathtt{l}} P' \qquad \alpha \neq x}{(\nu x{:}S)P \xrightarrow{\alpha,\mathtt{l}} (\nu x{:}S)P'} \text{[R-New]}$$

$$\frac{P \xrightarrow{\alpha,\mathtt{l}} P'}{P|Q \xrightarrow{\alpha,\mathtt{l}} P'|Q} \text{[R-Par]}$$

$$\frac{P \equiv Q \qquad Q \xrightarrow{\alpha,\mathtt{l}} Q' \qquad Q' \equiv P'}{P \xrightarrow{\alpha,\mathtt{l}} P'} \text{[R-Cong]}$$

Structural equivalence

$$
\begin{aligned}
P|0 &\equiv P \\
P|Q &\equiv Q|P \\
(P|Q)|R &\equiv Q|(P|R) \\
(\nu x{:}S)(\nu y{:}T)P &\equiv (\nu y{:}T)(\nu x{:}S)P \\
(\nu x{:}S)P|Q &\equiv (\nu x{:}S)(P|Q) && \text{if } x^p \notin \mathrm{fn}(Q) \\
(\nu x{:}S)0 &\equiv 0 && \text{if } S = \mathbf{end} \\
!P &\equiv P|\ !P
\end{aligned}
$$

$$P = \,!(loop?(x{:}S).x \triangleright [next : loop!x.0, end : 0]) \mid loop!x.Q$$

$$S = \mu X.\&[next : X, end : \mathsf{end}]$$

Would it be possible to assign a session type to loop?

Typing

Type Judgement

$$\Gamma \vdash P$$

P uses channels as specified by Γ

Environments Γ

- Partial function from polarized names to types
- Written $x_1^{p_1} : t_1, x_2^{p_2} : t_2, \ldots, x_n^{p_n} : t_n$
- Its satisfies one of the following conditions
  - $x^+, x^-, x \notin dom(\Gamma)$
  - $x \in dom(\Gamma)$ and $x^+, x^- \notin dom(\Gamma)$
  - $x^p \in dom(\Gamma)$ and $p \in \{+, -\}$ and $x^{\overline{p}}, x \notin dom(\Gamma)$
  - $x^+, x^- \in dom(\Gamma)$ and $x \notin dom(\Gamma)$

Typing

$$\frac{\Gamma_1 \vdash P_1 \qquad \Gamma_2 \vdash P_2}{\Gamma_1 + \Gamma_2 \vdash P_1 | P_2}\text{[T-Par]}$$

Context split

$$\begin{array}{rcll}
\Gamma + x^{+} : t & = & \Gamma, x^{+} : t & \text{if } x, x^{+} \notin \mathrm{dom}(\Gamma) \\
\Gamma + x^{-} : t & = & \Gamma, x^{-} : t & \text{if } x, x^{-} \notin \mathrm{dom}(\Gamma) \\
\Gamma + \quad x : t & = & \Gamma, x : t & \text{if } x, x^{+}, x^{-} \notin \mathrm{dom}(\Gamma) \\
(\Gamma, x : t) + \quad x : t & = & \Gamma, x : t & \text{if } t \text{ is not a session type}
\end{array}$$

Extended on context as

$$\begin{array}{rcl}
\Gamma + \qquad \emptyset & = & \Gamma \\
\Gamma + (x^{p} : t, \Delta) & = & (\Gamma + x^{p} : t) + \Delta
\end{array}$$

Linear usage of session endpoints

Typing

$$\frac{\Gamma_1 \vdash P_1 \qquad \Gamma_2 \vdash P_2}{\Gamma_1 + \Gamma_2 \vdash P_1 | P_2}\text{[T-Par]}$$

$$\frac{\Gamma, x^+ : S, x^- : \overline{S} \vdash P}{\Gamma \vdash (\nu x{:}S)P}\text{[T-Res]}$$

$$\frac{\Gamma, x^p : S, y : t \vdash P}{\Gamma, x^p : ?t.S \vdash x^p?(y{:}\mathtt{t}).P}\text{[T-In]}$$

$$\frac{\Gamma_1 \vdash v : t \qquad \Gamma_2, x^p : S \vdash P}{\Gamma_1 + (\Gamma_2, x^p : !t.S) \vdash x^p!v.P}\text{[T-Out]}$$

$$\frac{\Gamma, x : [t], y : t \vdash P}{\Gamma, x : [t] \vdash x?(y{:}\mathtt{t}).P}\text{[T-In-Un]}$$

$$\frac{\Gamma_1 \vdash v : t \qquad \Gamma_2, x : [t] \vdash P}{\Gamma_1 + (\Gamma_2, x : [t]) \vdash x!v.P}\text{[T-Out-Un]}$$

$$\frac{\Gamma, x^p : S_j \vdash P \qquad j \in I}{\Gamma, x^p : \oplus[\mathtt{l}_i : S_i]_{i \in I} \vdash x^p \triangleleft \mathtt{l}_j.P}\text{[T-Choice]}$$

$$\frac{\Gamma, x^p : S_i \vdash P_i \quad \forall i \in I}{\Gamma, x^p : \&[\mathtt{l}_i : S_i]_{i \in I} \vdash x^p \triangleright [\mathtt{l}_i : P_i]_{i \in I}}\text{[T-Branch]}$$

$$\frac{\Gamma \text{ completed}}{\Gamma \vdash 0}\text{[T-Nil]}$$

$$\frac{\Gamma \vdash P \quad \Gamma \text{ Unlimited}}{\Gamma \vdash !P}\text{[T-Rep]}$$

Γ Unlimited iff $\forall x \in \mathrm{dom}(\Gamma).\Gamma(x) \notin T$

On linearity

- Consider $P = x^+ ! y^+ . y^+ ! 1 . 0$.
- Does the following hold?

$$\Gamma, x^+ : !(!\mathtt{int}.\mathtt{end}).\mathtt{end}, y^+ : !\mathtt{int}.\mathtt{end} \vdash P$$

$$\frac{\Gamma_1 \vdash v : t \qquad \Gamma_2, x^p : S \vdash P}{\Gamma_1 + (\Gamma_2, x^p : !t.S) \vdash x^p ! v . P}\,\text{[T-Out]}$$

- No. Why does [T-Out] ban P?
- Take

$$Q = (\nu y{:}!\mathtt{int}.\mathtt{end})(\nu x{:}!(!\mathtt{int}.\mathtt{end}).\mathtt{end})(\\ P \mid x^- ?(z{:}!\mathtt{int}.\mathtt{end}).z ! 2 . 0 \mid y^- ?(z{:}\mathtt{int}).0)$$

- $Q \xrightarrow{\tau,-} Q'$ where

$$Q' = (\nu y{:}!\mathtt{int}.\mathtt{end})(\nu x{:}\mathtt{end})(y^+ ! 1 . 0 \mid y^+ ! 2 . 0 \mid y^- ?(z{:}\mathtt{int}).0)$$

  where two processes concurrently send on y^+
- A process does not use a session endpoint after *delegating* it (i.e., sending it over a different session endpoint)

Results

Theorem (Type Preservation)

- *If* $\Gamma \vdash P$ *and* $P \xrightarrow{\tau,-} Q$ *then* $\Gamma \vdash Q$.
- *If* $\Gamma, x^p : S, x^{\overline{p}} : \overline{S} \vdash P$ *and* $P \xrightarrow{x,\mathtt{l}} Q$ *then* $S \xrightarrow{\mathtt{l}} T$ *and* $\Gamma, x^p : T, x^{\overline{p}} : \overline{T} \vdash Q$.

Theorem (Type Safety)

Let $\Gamma \vdash P$ *where* Γ *balanced* [2]

- *If* $P \equiv (\nu\tilde{z}{:}\tilde{S})(x^p\,!\,v\,.\,P_1 \mid x^{\overline{p}}?(y{:}\mathtt{t})\,.\,P_2 \mid Q)$ *with* $p \in \{+, -\}$ *then* $x, x^+, x^- \notin \mathrm{fn}(Q)$ *and* $\Gamma, \tilde{z} : \tilde{S} \vdash v : t$
- *If* $P \equiv (\nu\tilde{z}{:}\tilde{S})(x^p \triangleleft \mathtt{l}_j\,.\,R \mid x^{\overline{p}} \triangleright [\mathtt{l}_i : P_i]_{i\in I} \mid Q)$ *with* $p \in \{+, -\}$ *then* $x, x^+, x^- \notin \mathrm{fn}(Q)$ *and* $j \in I$.

[2] Γ is balanced if $x^p : S$ and $x^{\overline{p}} : T$ implies $S = \overline{T}$

Properties

Does the following hold?

$$\vdash (\nu x{:}?\mathtt{int}.\mathtt{end})(x^{+}?(z{:}\mathtt{int}).x^{-}!z.0)$$

Yes!

- The process is well-typed and deadlocked

The type system ensures

- *Type Safety* in communication (e.g., received values are of the expected type)
- *Session Fidelity* (e.g., communication follows the flow described by the session type)
- The type system does not ensure deadlock-freedom

Deadlock

Deadlocked Process

$$P = x^{+}?(z{:}\mathtt{int}).y^{-}!1.0 \quad | \quad y^{+}?(z{:}\mathtt{int}).x^{-}!1.0$$

Is P well-typed?

$$\vdash (\nu x{:}?\mathtt{int}.\mathtt{end})(\nu y{:}?\mathtt{int}.\mathtt{end})P$$

Yes!

- The process is well-typed and deadlocked
- The type system does not check the dependencies between different sessions