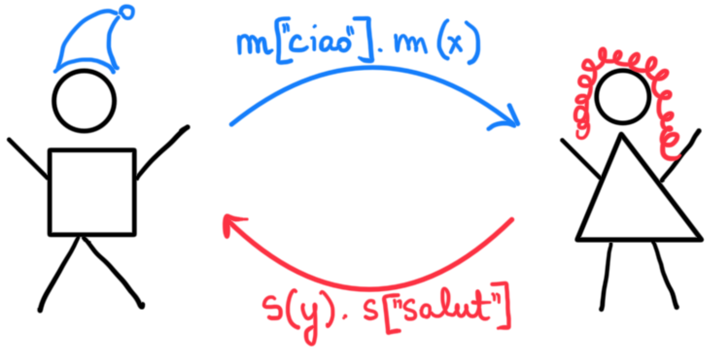
SESSION TYPES

Lecture 2 : Type Safety

MGS 2024

Matteo Acclavio ⊗ Sonia Marin



Yesterday we introduced the basic concepts of session types:

— the language of the π-calculus with sessions for message-passing and terminated processes
— its operational semantics to describe valid communication behaviour
— a session type system that carves out a (hopefully well-behaved) subset of the processes

Today we will discuss the properties that are (and are not) guaranteed by the proposed type system.

Then we will consider some ways for extending the basic language with more and more realistic constructions.

Acknowledgements Thank you to S. Gay and V. Vasconcelos for sharing their book draft with us. Many presentational choices follow theirs ♡
Thank you to M. Carbone for being a great teacher and always on hand for session type questions ♡

When do processes get stuck?

$$(\nu uv)\,(\overset{\text{close}}{u[\,].P} \mid \overset{\text{wait}}{v(\,).Q}) \longrightarrow \; ?$$

$$(\nu uv)\,(\overset{\text{receive/input}}{u(x).P} \mid \overset{\text{send/output}}{v[n].Q}) \longrightarrow \; ?$$

$$(\nu uv)\,(u[\,].P \mid v[n].Q) \longrightarrow \; ?$$

$$(\nu uv)\,(u(\,).P \mid v[\,].Q \mid v[\,].R) \longrightarrow \; ? \quad \longrightarrow \; ?$$

$$(\nu uv)(\nu wz)\quad (u[n].w(x).P \mid z[n].v(y).Q) \longrightarrow \; ?$$

$$(\nu uv)(\nu wz)\quad (w(x).u[n].P \mid v(y).z[n].Q) \longrightarrow \; ?$$

Formally, a process P is reducible if $P \longrightarrow Q$ for some Q, irreducible otherwise

Runtime errors and Races

threads that do not contain any restrictions (νuv) but can contain parallel $|$

The subject of a prefix is the channel endpoint that it owns

$$\mathrm{subj}(\underset{\text{receive/input}}{u(x)}.P) = \mathrm{subj}(\underset{\text{send/output}}{u[e]}.P) = \mathrm{subj}(\underset{\text{wait}}{u()}.P) = \mathrm{subj}(\underset{\text{close}}{u[\,]}.P) = u$$

Process $(\nu u_1 v_1)\dots(\nu u_n v_n)\underbrace{(P_1 \mid \dots \mid P_m)}_{\text{if } m=0:\ \mathbf{inact}}$ is in canonical form

(prefixes: $P_1, \dots, P_m$)

Exercise Every process is structurally congruent to a canonical form.

need to complete the definition of $\equiv$ with an axiom $\alpha.(\nu uv)P \equiv (\nu uv)\alpha.P$ for $\alpha \in \{w(), w[\,], w(x), w[e]\}$ and $w \notin fv(P)$

Processes of the form
$$\left\{\begin{array}{l} (\nu uv)(\overset{\text{close}}{u[\,]}.P \mid \overset{\text{wait}}{v()}.Q) \\ (\nu uv)(\overset{\text{receive/input}}{u(x)}.P \mid \overset{\text{send/output}}{v[n]}.Q) \end{array}\right\}$$
are redexes

A process in canonical form $(\nu u_1 v_1)\dots(\nu u_n v_n)(P_1 \mid \dots \mid P_m)$

- contains a race: if there are $i \neq j$ such that $\mathrm{subj}(P_i) = \mathrm{subj}(P_j)$

  Example: $(\nu uv)(u().P \mid v[\,].Q \mid v[\,].R)$

- is a runtime error: if there are i, j, k such that

$\mathrm{subj}(P_i) = u_k$, $\mathrm{subj}(P_j) = v_k$ but $(\nu u_k v_k)(P_i \mid P_j)$ not redex

Example: $(\nu uv)\,(u[\,].\,P \mid v[n].Q)$

* Can a reducible process reduce to an irreducible one?

* Are all irreducible processes runtime errors?

$$(\nu uv)\,(\nu wz)\,(u[n].w(x).\,P \mid z[n].v(y).\,Q)$$

$$(\nu uv)\,(\nu wz)\,(w(x).v[n].\,P \mid v(y).z[n].\,Q)$$

A process is deadlocked if it is irreducible but neither runtime error, nor terminated ($\equiv$ inact)

When are processes typable?

A process P is typable if there exists a context Γ such that $\Gamma \vdash P$ can be obtained as the root of a typing derivation built from the rules:

$$\frac{}{\cdot \vdash \mathrm{inact}}\ (\mathrm{INACT})$$

$$\frac{\Gamma \vdash P}{\Gamma, u:\mathrm{wait} \vdash u().P}\ (\mathrm{WAIT}) \qquad \frac{\Gamma,\ y:T,\ u:S \vdash P}{\Gamma,\ u:(T) \blacktriangleleft S \vdash u(y).P}\ (\mathrm{RECV})$$

$$\frac{\Gamma \vdash P}{\Gamma, u:\mathrm{close} \vdash u[\,].P}\ (\mathrm{CLOSE}) \qquad \frac{\Gamma \Vdash e:T \qquad \Gamma,\ u:S \vdash P}{\Gamma,\ u:[T] \blacktriangleleft S \vdash u[e].P}\ (\mathrm{SEND})$$

$$\frac{\Gamma \vdash P \qquad \Gamma' \vdash Q}{\Gamma, \Gamma' \vdash (P \mid Q)}\ (\mathrm{PAR}) \qquad \frac{\Gamma,\ u:S,\ v:S^{\perp} \vdash P}{\Gamma \vdash (\nu uv)P}\ (\mathrm{RES})$$

Supposing P, Q, R are typable processes (in appropriate contexts)

$$\dfrac{\dfrac{\dfrac{\dfrac{\nabla}{\Gamma \vdash P}}{\Gamma, u:\mathsf{close} \vdash u[\,].P}\;(\mathrm{CLOSE}) \qquad \dfrac{\dfrac{\nabla}{\Gamma' \vdash Q}}{\Gamma', v:\mathsf{wait} \vdash v(\,).Q}\;(\mathrm{WAIT})}{\Gamma, \Gamma', u:\mathsf{close}, v:\mathsf{wait} \vdash u[\,].P \mid v(\,).Q}\;(\mathrm{PAR})}{\Gamma, \Gamma' \vdash (\nu uv)\,(u[\,].P \mid v(\,).Q)}\;(\mathrm{RES})$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\nabla}{\Gamma, u:S, x:\mathsf{nat} \vdash P}}{\Gamma, u:(\mathsf{nat})\blacktriangleleft S \vdash u(x).P}\;(\mathrm{RECV}) \qquad \dfrac{\dfrac{n \in \mathbb{N}}{\Gamma' \Vdash n:\mathsf{nat}} \quad \dfrac{\nabla}{\Gamma', v:S^{\perp} \vdash Q}}{\Gamma', v:[\mathsf{nat}]\blacktriangleleft S^{\perp} \vdash v[n].Q}\;(\mathrm{SEND})}{\Gamma, \Gamma', u:(\mathsf{nat})\blacktriangleleft S, v:[\mathsf{nat}]\blacktriangleleft S^{\perp} \vdash u(x).P \mid v[n].Q}\;(\mathrm{PAR})}{\Gamma, \Gamma' \vdash (\nu uv)\,(u(x).P \mid v[n].Q)}\;(\mathrm{RES})$$

$$\dfrac{\dfrac{\dfrac{S = \mathsf{close}}{\Gamma, u:S \vdash u[\,].P}\;(\mathrm{CLOSE}) \qquad \dfrac{S^{\perp} = [\mathsf{nat}]\blacktriangleleft S'}{\Gamma', v:S^{\perp} \vdash v[n].Q}\;(\mathrm{SEND})}{\Gamma, \Gamma', u:S, v:S^{\perp} \vdash u[\,].P \mid v[n].Q}\;(\mathrm{PAR})}{\Gamma, \Gamma' \vdash (\nu uv)\,(u[\,].P \mid v[n].Q)}\;(\mathrm{RES})$$

linearity forbids

$$\dfrac{\dfrac{\dfrac{S = \mathsf{wait}}{\Gamma, u:S \vdash u(\,).P} \qquad \dfrac{\dfrac{S = \mathsf{close}}{\Gamma', v:S^{\perp} \vdash v[\,].Q} \qquad \Gamma'' \vdash v[\,].R}{\Gamma', \Gamma'', v:S^{\perp} \vdash v[\,].Q \mid v[\,].R}}{\Gamma, \Gamma', \Gamma'', u:S, v:S^{\perp} \vdash u(\,).P \mid (v[\,].Q \mid v[\,].R)}}{\Gamma, \Gamma', \Gamma'' \vdash (\nu uv)(u(\,).P \mid (v[\,].Q \mid v[\,].R))}\;(\mathrm{RES})$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash n:\mathsf{nat}} \quad \dfrac{\dfrac{\nabla}{\Gamma, u:S, w:S', x:\mathsf{nat} \vdash P}}{\Gamma, u:S, w:(\mathsf{nat})\blacktriangleleft S' \vdash w(x).P}}{\Gamma, u:[\mathsf{nat}]\blacktriangleleft S, w:(\mathsf{nat})\blacktriangleleft S' \vdash u[n].w(x).P} \qquad \dfrac{\dfrac{}{\Gamma' \vdash n:\mathsf{nat}} \quad \dfrac{\dfrac{\nabla}{\Gamma', v:S^{\perp}, y:\mathsf{nat}, z:S'^{\perp} \vdash Q}}{\Gamma', v:(\mathsf{nat})\blacktriangleleft S^{\perp}, z:S'^{\perp} \vdash v(y).Q}}{\Gamma', v:(\mathsf{nat})\blacktriangleleft S^{\perp}, z:[\mathsf{nat}]\blacktriangleleft S'^{\perp} \vdash z[n].v(y).Q}}{\Gamma, \Gamma', u:[\mathsf{nat}]\blacktriangleleft S, v:(\mathsf{nat})\blacktriangleleft S^{\perp}, w:(\mathsf{nat})\blacktriangleleft S', z:[\mathsf{nat}]\blacktriangleleft S'^{\perp} \vdash u[n].w(x).P \mid z[n].v(y).Q}}{\Gamma, \Gamma', u:[\mathsf{nat}]\blacktriangleleft S, v:(\mathsf{nat})\blacktriangleleft S^{\perp} \vdash (\nu wz)\,(u[n].w(x).P \mid z[n].v(y).Q)}}{\Gamma, \Gamma' \vdash (\nu uv)\,(\nu wz)\,(u[n].w(x).P \mid z[n].v(y).Q)}$$

$$\dfrac{\dfrac{\dfrac{\Gamma, u:S, w:S', x:\mathrm{nat} \vdash P}{\Gamma, u:[\mathrm{nat}]\blacktriangleleft S,\ w:(\mathrm{nat})\blacktriangleleft S' \vdash w(x).u[n].P} \qquad \dfrac{\Gamma', v:S^{\perp}, y:\mathrm{nat}, z:S'^{\perp} \vdash Q}{\Gamma', v:(\mathrm{nat})\blacktriangleleft S^{\perp}, z:[\mathrm{nat}]\blacktriangleleft S'^{\perp} \vdash z[n].v(y).Q}}{\Gamma, \Gamma', u:[\mathrm{nat}]\blacktriangleleft S, v:(\mathrm{nat})\blacktriangleleft S^{\perp}, w:(\mathrm{nat})\blacktriangleleft S', z:[\mathrm{nat}]\blacktriangleleft S'^{\perp} \vdash w(x).u[n].P \mid v(y).z[n].Q}}{(\nu uv)(\nu wz)(w(x).u[n].P \mid v(y).z[n].Q)}$$

We observe on these examples that:

- processes with races do not seem typable
- runtime errors do not seem typable
- some deadlocked processes are typable

Typing Guarantees

Absence of races:

Theorem: Typable processes do not contain races

Proof: By contradiction, suppose a process P contains a race and there is a Γ for which we can derive $\Gamma \vdash P$.

needs Preservation for $\equiv$: $P \equiv Q$ implies $\Gamma \vdash P$ iff $\Gamma \vdash Q$

we can assume P in canonical form $(\nu u_1 v_1) \dots (\nu u_n v_n)(P_1 \mid \dots \mid P_m)$ s.t. there are $i \neq j$ with $\mathrm{subj}(P_i) = \mathrm{subj}(P_j) = w$

needs a technical Inversion Lemma

hence, we can derive $\Delta_i \vdash P_i$ and $\Delta_j \vdash P_j$ but Δ_i and Δ_j cannot contain $w:S_i$ and $w:S_j$ by linearity

contradicts

1. If $\Gamma \vdash \mathrm{inact}$ then $\Gamma = \emptyset$
2. If $\Gamma \vdash u().P$ then $\Gamma = \Gamma', u:\mathrm{wait}$ and $\Gamma' \vdash P$
3. If $\Gamma \vdash u[].P$ then $\Gamma = \Gamma', u:\mathrm{close}$ and $\Gamma' \vdash P$
4. If $\Gamma \vdash u(x).P$ then $\Gamma = \Gamma', u:(T)\blacktriangleleft S$ and $\Gamma', x:T, u:S \vdash P$
5. If $\Gamma \vdash u[e].P$ then $\Gamma = \Gamma', u:[T]\blacktriangleleft S$ and $\Gamma', u:S \vdash P$ and $\Gamma' \vdash e:T$

6. If $\Gamma \vdash (P|Q)$ then $\Gamma = \Gamma', \Gamma''$ and $\Gamma' \vdash P$ and $\Gamma'' \vdash Q$
7. If $\Gamma \vdash (\nu uv)P$ then $\Gamma, u: S, v: S^\perp \vdash P$ for some S.

Absence of immediate errors:

Theorem: Typable processes are not runtime errors

Proof: By contradiction, suppose a process P is a runtime error and there is a Γ for which we can derive $\Gamma \vdash P$

needs Preservation for $\equiv$: $P \equiv Q$ implies $\Gamma \vdash P$ iff $\Gamma \vdash Q$

we can assume P in canonical form $(\nu u_1 v_1) \dots (\nu u_n v_n)(P_1 | \cdots | P_m)$ with i, j, k s.t. $\boxed{subj(P_i) = u_k,\ subj(P_j) = v_k}$ but $(\nu u_k v_k)(P_i | P_j)$ not redex

needs technical Inversion Lemma

we can derive $\Delta_i, u_k : S_k \vdash P_i$ and $\Delta_j, v_k : S_k^\perp \vdash P_j$

the communication of P_i (resp P_j) follows the structure of S_k (resp $S_k^\perp$)

contradicts

Preservation:

Theorem: If $\Gamma \vdash P$ and $P \longrightarrow Q$ then $\Gamma \vdash Q$

Proof: By induction on the definition of $\longrightarrow$ with base cases:

- $(\nu uv)(u().P \mid v[\,].Q) \longrightarrow (P \mid Q)$

Suppose $\Gamma \vdash (\nu uv)(u().P | v[\,].Q)$ derivable

By Inversion Lemma: $\Gamma = \Gamma', \Gamma''$ and $\Gamma' \vdash P$ and $\Gamma'' \vdash Q$, hence $\dfrac{\Gamma' \vdash P \qquad \Gamma'' \vdash Q}{\Gamma \vdash (P|Q)}$

- $(\nu uv)(u(x).P \mid v[e].Q) \longrightarrow (\nu uv)(P[c/x] \mid Q)$ if $e \downarrow c$

and induction cases:

$$-\ \frac{P \longrightarrow Q}{P \mid R \longrightarrow Q \mid R} \qquad -\ \frac{P \longrightarrow Q}{(\nu u v) P \longrightarrow (\nu u v) Q} \qquad -\ \underbrace{\frac{P \equiv P' \quad P \longrightarrow Q \quad Q \equiv Q'}{P' \longrightarrow Q'}}$$

needs Preservation for $\equiv$:
$P \equiv Q$ implies $\Gamma \vdash P$ iff $\Gamma \vdash Q$

Type safety

$P \longrightarrow^* Q$

A process P reduces to Q when $P \equiv P_0 \longrightarrow P_1 \longrightarrow P_2 \longrightarrow \dots \longrightarrow P_n \equiv Q$ for $n \geq 0$.

Corollary If P is typable and $P \longrightarrow^* Q$, then Q is not a runtime error.

$\boxed{\text{Choice}}$

The session types introduced so far have a simple structure:
a finite sequence of messages, sent or received.

More realistic protocols allow choices to be made, e.g., to let a client choose among the services offered by a server.

Our base sets now also contain labels denoted $k, l \dots$
and L for a finite, non-empty set

And processes are extended by:

$P ::= \dots \mid u \triangleright \{ l : P_l \}$ $\longleftarrow$ $\boxed{\text{external choice}}$ (branching)

$\mid \ldots \{\ldots\}_{l \in L}$

offers a fixed range of alternatives to continue as one of the P_l

$\mid \; u \triangleleft k : P$ ⟵ $\boxed{\text{internal choice}}$ (selection)

select one of the label $k \in L$ and continue as P

Example:
$$P = u \triangleright \{\text{init} : u[1].\ \text{inact}$$
$$\text{incr} : u(x).u[x+1].\ \text{inact}$$
$$\text{sum} : u(x).\ u(y).\ u[x+y].\ \text{inact}\}$$

$$Q = v \triangleleft \text{incr} : v[2].v(z).\ Q'$$

The <u>operational semantics</u> is also extended

$$(\nu uv)\,(\overbrace{u \triangleright \{l : P_l\}_{l\in L}}^{\text{offers a choice}} \mid \overbrace{v \triangleleft k : Q}^{\text{select an option in } L}) \longrightarrow (\nu uv)\,(P_k \mid Q) \quad \text{if } k \in L$$

endpoints u and v are co-variables

Example: $(\nu uv)\,(P \mid Q) \longrightarrow (\nu uv)\,(u(x).u[x+1].\ \text{inact} \mid v[2].v(z).\ Q')$

$$\longrightarrow (\nu uv)\,(u[2+1].\ \text{inact} \mid v(z).\ Q') \longrightarrow Q'[3/z]$$

We add corresponding <u>dual types</u>:

$$S ::= \ldots \mid \& \{l : S_l\}_{l\in L} \mid \oplus \{l : S_l\}_{l\in L}$$

external/branching   internal/selection

Finally, the two new <u>typing rules</u> for them:

$$\{\Gamma, x : S_l \vdash P_l\}_{l\in L}$$

$$\frac{}{\Gamma,\ x : \&\{l : S_l\}_{l \in L} \vdash x \triangleright \{l : P_l\}_{l \in L}}\ (\text{BRA})$$

$$\frac{\Gamma,\ x : S_k \vdash P}{\Gamma,\ x : \oplus\{l : S_l\}_{l \in L} \vdash x \triangleleft k . P}\ (\text{SEL})_{k \in L}$$

Example:

$$\begin{aligned} P = u \triangleright \{ & init : u[1].\, u[\,].\, inact \\ & incr : u(x).\, u[x+1].\, u[\,].\, inact \\ & sum : u(x).\, u(y).\, u[x+y].\, u[\,].\, inact \} \end{aligned}$$

$$Q = v \triangleleft incr : v[2].\, v(z).\, v().\, Q'$$

$$\begin{aligned} u : \&\{ & init : [nat] \triangleleft close,\ incr : (nat) \triangleleft [nat] \triangleleft close, \\ & sum : (nat) \triangleleft (nat) \triangleleft [nat] \triangleleft close \} \vdash P \end{aligned}$$

What would be a suitable typing for Q?

Next steps:

Choice	✓
Shared Channels	?
Infinite behaviour	?
Asynchrony	?
Multiparty	Tomorrow?
Curry-Howard (LL)	Friday?