Lambda Calculus + Binary Sessions (à la FuSe)

Hernán Melgratti

ICC University of Buenos Aires-Conicet

Formal FuSe

Syntax

Expression	e	$::=$	x	variable
		$\mid$	$\lambda x.e$	abstraction
		$\mid$	$e_1\, e_2$	application
		$\mid$	$\mathtt{let}\ x = e_1\ \mathtt{in}\ e_2$	let
		$\mid$	$\mathtt{c}$	constant
		$\mid$	a^p	endpoint
Process	P, Q	$::=$	$\langle e \rangle$	thread
		$\mid$	$P \parallel Q$	composition
		$\mid$	$(\nu a)P$	session
		$\mid$	$\mathtt{error}$	run-time error

Constant

$$\mathtt{c} ::= () \mid \mathtt{pair} \mid \mathtt{fst} \mid \mathtt{snd} \mid \mathtt{inl} \mid \mathtt{inr} \mid \mathtt{cases} \mid \mathtt{fix} \mid \mathtt{fork} \mid \mathtt{create} \mid \mathtt{send} \mid \mathtt{receive} \mid \mathtt{branch} \mid \mathtt{left} \mid \mathtt{right}$$

Polarity

$$p ::= + \mid - \mid \star$$

$\star$ denotes *invalid endpoint* that should not be used. Moreover, $\overline{\star} = \star$.

Formal FuSe

Values

Values $v, w ::= \lambda x \mathbin{.} e \mid \mathtt{c} \mid \mathtt{c}^1 v \mid \mathtt{c}^2 v\ w \mid a^p$

$\mathtt{c}^n$	n **max**	**Sugared**	**Description**
`()`	0		unit
`pair`	2	(v, w)	pair creation
`fst`, `snd`	0		pair projection
`inl`, `inr`	1		left/right injection
`cases`	2		sum deconstructor
`fix`	0		fixpoint combinator
`fork`	1		forking a process
`create`	0		session creation
`close`	0		terminate session
`receive`	0		input
`send`	1		output
`branch`	0		offer choice
`left`	0		choose left
`right`	0		choose right

Formal FuSe

Reduction of expressions (1)

$$
\begin{array}{lrcl}
[\mathrm{r}-\mathrm{beta}] & (\lambda x.e)\,v & \rightarrow & e\{v/x\} \\
[\mathrm{r}-\mathtt{let}] & \mathtt{let}\ x = v\ \mathtt{in}\ e & \rightarrow & e\{v/x\} \\
[\mathrm{r}-\mathtt{fix}] & \mathtt{fix}\ v & \rightarrow & v(\lambda x.\mathtt{fix}\ v\ x) \\
[\mathrm{r}-\mathtt{fst}] & \mathtt{fst}\ (\mathtt{pair}\ v\ w) & \rightarrow & v \\
[\mathrm{r}-\mathtt{snd}] & \mathtt{snd}\ (\mathtt{pair}\ v\ w) & \rightarrow & w \\
[\mathrm{r}-\mathtt{inl}] & \mathtt{cases}\ (\mathtt{inl}\ v)\ v\ w & \rightarrow & v \\
[\mathrm{r}-\mathtt{inr}] & \mathtt{cases}\ (\mathtt{inr}\ v)\ v\ w & \rightarrow & w
\end{array}
$$

Formal FuSe

Evaluation Contexts

- Contextual rules are handled by defining **Evaluation Contexts**

$$\mathscr{E} ::= [\,] \mid \mathscr{E}\ e \mid v\ \mathscr{E} \mid \texttt{let}\ x = \mathscr{E}\ \texttt{in}\ e$$

- $\mathscr{E}[e]$ stands for the result of replacing the hole $[\,]$ in $\mathscr{E}$ with e

Semantics

Reduction of processes(1)

$$P \xrightarrow{\ell} Q$$

where

$$
\begin{array}{lcll}
\textbf{Labels } \ell & ::= & \tau & \text{internal action} \\
& | & \mathsf{c}a & \text{session } a \text{ has been closed} \\
& | & \mathsf{m}ap & \text{message exchange from } a^{p} \text{ to } a^{\overline{p}}
\end{array}
$$

Enpoints associated with a label $\quad \mathrm{ep}(\_)$

$$
\begin{aligned}
\mathrm{ep}(\tau) &= \{\} \\
\mathrm{ep}(\mathsf{c}a) &= \{a^{+}, a^{-}\} \\
\mathrm{ep}(\mathsf{m}ap) &= \{a^{+}, a^{-}\}
\end{aligned}
$$

Semantics

Reduction of processes(2)

$$
\begin{array}{lrl}
[\mathrm{r}-\mathtt{fork}] & \langle \mathscr{E}[\mathtt{fork}\ v\ w]\rangle \xrightarrow{\tau} \langle \mathscr{E}[()]\rangle \parallel \langle v\ w\rangle & \\
[\mathrm{r}-\mathtt{create}] & \langle \mathscr{E}[\mathtt{create}\ ()]\rangle \xrightarrow{\tau} (\nu a)\langle \mathscr{E}[\mathtt{pair}\ a^{+}\ a^{-}]\rangle & a \text{ fresh} \\
[\mathrm{r}-\mathtt{close}] & \langle \mathscr{E}[\mathtt{close}\ a^{p}]\rangle \parallel \langle \mathscr{E}'[\mathtt{close}\ a^{\overline{p}}]\rangle \xrightarrow{\mathsf{c}a} \langle \mathscr{E}_{\mathsf{c}a}[()]\rangle \parallel \langle \mathscr{E}'_{\mathsf{c}a}[()]\rangle &
\end{array}
$$

$$\mathscr{E}_{\mathsf{c}a} = \mathscr{E}\{a^{\star}/a^{+}, a^{\star}/a^{-}\}$$

Semantics

Reduction of processes(3)

$$[\mathrm{r-comm}] \quad \langle \mathscr{E}[\texttt{send}\ a^p\ v] \rangle \parallel \langle \mathscr{E}'[\texttt{receive}\ a^{\overline{p}}] \rangle \xrightarrow{map} \langle \mathscr{E}_{\mathrm{map}}[a^p] \rangle \parallel \langle \mathscr{E}'_{\mathrm{map}}[\texttt{pair}\ v_{\mathrm{map}}\ a^{\overline{p}}] \rangle$$

$$[\mathrm{r-\texttt{left}}] \quad \langle \mathscr{E}[\texttt{left}\ a^p] \rangle \parallel \langle \mathscr{E}'[\texttt{branch}\ a^{\overline{p}}] \rangle \xrightarrow{map} \langle \mathscr{E}_{\mathrm{map}}[a^p] \rangle \parallel \langle \mathscr{E}'_{\mathrm{map}}[\texttt{inl}\ a^{\overline{p}}] \rangle$$

$$[\mathrm{r-\texttt{rigth}}] \quad \langle \mathscr{E}[\texttt{right}\ a^p] \rangle \parallel \langle \mathscr{E}'[\texttt{branch}\ a^{\overline{p}}] \rangle \xrightarrow{map} \langle \mathscr{E}_{\mathrm{map}}[a^p] \rangle \parallel \langle \mathscr{E}'_{\mathrm{map}}[\texttt{inr}\ a^{\overline{p}}] \rangle$$

$$\begin{aligned} \mathscr{E}_{\mathrm{map}} &= \mathscr{E}\{a^\star/a^+, a^\star/a^-\} \\ v_{\mathrm{map}} &= v\{a^\star/a^+, a^\star/a^-\} \end{aligned}$$

Semantics

Reduction of processes(4)

$$[\mathsf{r-thread}] \quad \frac{e \xrightarrow{\tau} e'}{\langle \mathscr{E}[e] \rangle \xrightarrow{\tau} \langle \mathscr{E}[e'] \rangle}$$

$$[\mathsf{r-par}] \quad \frac{P \xrightarrow{\ell} Q}{P \parallel R \xrightarrow{\ell} Q \parallel R}$$

$$[\mathsf{r-new}] \quad \frac{P \xrightarrow{\ell} Q}{(\nu a)P \xrightarrow{\ell \backslash a} (\nu a)Q}$$

$$[\mathsf{r-struct}] \quad \frac{P \equiv P' \qquad P' \xrightarrow{\ell} Q' \qquad Q' \equiv Q}{P \xrightarrow{\ell} Q}$$

$$\ell \backslash a = \begin{cases} \tau & \textit{if } \ell = \mathsf{map} \textit{ or } \ell = \mathsf{ca} \\ \ell & \textit{otherwise} \end{cases}$$

$\equiv$ is an equivalence relation such that:

- $\parallel$ is associative and commutative, with $\langle () \rangle$ as the identity
- $(\nu a)\langle () \rangle = \langle () \rangle$
- $(\nu a)P \parallel Q = (\nu a)(P \parallel Q)$ if $a \notin \mathrm{fn}(Q)$

Semantics

Reduction of processes(5)

$$[\mathrm{r}-\texttt{error}]\ \langle \mathscr{E}[K\ a^\star] \rangle \xrightarrow{\tau} \texttt{error}$$

$$K\ ::=\ \texttt{close} \mid \texttt{send}\ v \mid \texttt{receive} \mid \texttt{left} \mid \texttt{right} \mid \texttt{branch}$$

Typing

Types

Type Schemes	σ	$::=$	$t \mid \forall\alpha.\sigma \mid \forall A.\sigma$
Types	t, s	$::=$	$\alpha \mid \mathtt{unit} \mid t \rightarrow s \mid t + s \mid t \times s \mid T$
Session Types	T, S	$::=$	$A \mid \overline{A} \mid \mathtt{end} \mid \mathtt{!}t\,\mathtt{.}\,T \mid \mathtt{?}t\,\mathtt{.}\,T \mid T \oplus S \mid T \,\&\, S$

Equations are interpreted *coinductively*: Infinite terms instead of concrete syntax for infinite types

Instantiation of schemes

$$t \succ t \qquad\qquad \frac{\sigma \succ t}{\forall\alpha.\sigma \succ t\{s/\alpha\}}$$

Typing

Type schemes of constants

$$
\begin{aligned}
() &: \texttt{unit} \\
\texttt{pair} &: \forall\alpha,\beta.\alpha \to \beta \to \alpha \times \beta \\
\texttt{fst} &: \forall\alpha,\beta.\alpha \times \beta \to \alpha \\
\texttt{snd} &: \forall\alpha,\beta.\alpha \times \beta \to \beta \\
\texttt{inl} &: \forall\alpha,\beta.\alpha \to \alpha + \beta \\
\texttt{inr} &: \forall\alpha,\beta.\beta \to \alpha + \beta \\
\texttt{cases} &: \forall\alpha,\beta,\gamma.(\alpha + \beta) \to (\alpha \to \gamma) \to (\beta \to \gamma) \to \gamma \\
\texttt{fix} &: \forall\alpha,\beta.((\alpha \to \beta) \to \alpha \to \beta) \to \alpha \to \beta \\
\texttt{fork} &: \forall\alpha.((\alpha \to ()) \to \alpha \to \texttt{unit} \\
\texttt{create} &: \forall A.() \to A \times \overline{A} \\
\texttt{close} &: \texttt{end} \to \texttt{unit} \\
\texttt{send} &: \forall\alpha, A.\alpha \to \texttt{!}\alpha\texttt{.}A \to A \\
\texttt{receive} &: \forall\alpha, A.\texttt{?}\alpha\texttt{.}A \to \alpha \times A \\
\texttt{left} &: \forall A, B.A \oplus B \to A \\
\texttt{right} &: \forall A, B.A \oplus B \to B \\
\texttt{branch} &: \forall A, B.A \,\&\, B \to A + B
\end{aligned}
$$

Typing

Typing

Typing rules for expressions $\boxed{\Gamma \vdash e : t}$

[t-const]
$$\frac{\text{typeof}(\mathtt{c}) \succ t}{\Gamma \vdash \mathtt{c} : t}$$

[t-name]
$$\frac{\sigma \succ t}{\Gamma, u : \sigma \vdash u : t}$$

[t-fun]
$$\frac{\Gamma, x : t \vdash e : s}{\Gamma \vdash \lambda x.\, e : t \to^{\iota} s}$$

[t-app]
$$\frac{\Gamma \vdash e_1 : t \to^{\iota} s \qquad \Gamma \vdash e_2 : t}{\Gamma \vdash e_1 e_2 : s}$$

[t-let]
$$\frac{\Gamma \vdash e_1 : t_1 \qquad \Gamma, x : \text{Close}(t_1, \Gamma) \vdash e_2 : t}{\Gamma \vdash \mathtt{let}\ x = e_1\ \mathtt{in}\ e_2 : t}$$

$\text{Close}(t, \Gamma) = \forall \alpha_1, \ldots, \alpha_n$ where $\{\alpha_1, \ldots, \alpha_n\} = fv(t) \setminus fv(\Gamma)$

Typing

Typing

Typing rules for processes $\boxed{\Gamma \vdash P}$

[t-thread]
$$\frac{\Gamma \vdash e : \mathtt{unit}}{\Gamma \vdash \langle e \rangle}$$

[t-par]
$$\frac{\Gamma \vdash P_1 \qquad \Gamma \vdash P_2}{\Gamma \vdash P_1 \parallel P_2}$$

[t-session]
$$\frac{\Gamma, a^{+} : T, a^{-} : \overline{T}, a^{\star} : \forall A.A \vdash P}{\Gamma \vdash (\nu a)P}$$

Properties

Endpoint affine/linear process

Let $\mathscr{A}$, $\mathscr{L}$ be the largest predicates such that:

- if either $\mathscr{A}(P)$ or $\mathscr{L}(P)$ and $P \equiv (\nu a_1 \dots a_n)(\mathscr{E}[K\ a^p] \parallel Q)$, then $p \in \{+, -\}$;
- if $\mathscr{L}(P)$ and $P \equiv (\nu a)Q$ and $a^p \in \mathsf{fv}(Q)$ and $p \in \{+, -\}$, then $a^{\overline{p}} \in \mathsf{fv}(Q)$
- if $\mathscr{L}(P)$ and $P \xrightarrow{\ell} Q$, then $\mathscr{L}(Q)$;
- if $\mathscr{A}(P)$ and $P \xrightarrow{\ell} Q$, then $\mathscr{A}(Q)$;

Properties

Balanced type environment

Γ is *balanced* if:

- for every $a^p \in \mathrm{dom}(\Gamma)$ with $p \in \{+, -\}$ we have $a^{\overline{p}}, a^\star \in \mathrm{dom}(\Gamma)$ and $\Gamma(a^p) \perp \Gamma(a^{\overline{p}})$
- for every $a^\star \in \mathrm{dom}(\Gamma)$ we have $\Gamma(a^p) = \forall A.A$

Properties

Subject reduction

- If $\Gamma \vdash P$ with Γ balanced and $\mathscr{A}(P)$ and $P \xrightarrow{\ell} Q$, then there exists Γ' such that $\Gamma \xrightarrow{\ell} \Gamma'$ and $\Gamma' \vdash Q$.

Semantics of types

$$
\begin{array}{rcl}
\Gamma & \xrightarrow{\ell} & \Gamma \\
\Gamma, a^p : \mathtt{!}t\mathtt{.}T, a^{\overline{p}} : \mathtt{?}t\mathtt{.}\overline{T} & \xrightarrow{\mathsf{map}} & \Gamma, a^p : T, a^{\overline{p}} : \overline{T} \\
\Gamma, a^p : T \oplus S, a^{\overline{p}} : \overline{T} \mathbin{\&} \overline{S} & \xrightarrow{\mathsf{map}} & \Gamma, a^p : T, a^{\overline{p}} : \overline{T} \\
\Gamma, a^p : T \oplus S, a^{\overline{p}} : \overline{T} \mathbin{\&} \overline{S} & \xrightarrow{\mathsf{map}} & \Gamma, a^p : S, a^{\overline{p}} : \overline{S} \\
\Gamma, a^p : \mathsf{end}, a^{\overline{p}} : \mathsf{end} & \xrightarrow{\mathsf{ca}} & \Gamma
\end{array}
$$

Properties

Type safety

Let $\Gamma \vdash P$ and $\mathscr{A}$. Then,

- If $P \equiv (\nu a_1 \dots a_n)(\mathscr{E}[\texttt{send}\ v\ w] \parallel Q)$, then there exists Γ' such that $\Gamma, \Gamma' \vdash v : t$ and $\Gamma, \Gamma' \vdash w : \,!t\,.\,T$.
- If $P \equiv (\nu a_1 \dots a_n)(\mathscr{E}[\texttt{left}\ v\ ] \parallel Q)$ or $P \equiv (\nu a_1 \dots a_n)(\mathscr{E}[\texttt{right}\ v\ ] \parallel Q)$, then there exists Γ' such that $\Gamma, \Gamma' \vdash v : S \oplus T$