On the Expressiveness of Higher Order Sessions

No Author Given

Imperial College London

1 A Higher Order Session Calculus

We define a session calculus augmented with higher order semantics.

1.1 Syntax

We assume the countable sets:

$$\begin{array}{llll} S = \{s, s_1, \dots\} & \text{Sessions} & \overline{S} = \{\overline{s} \mid s \in S\} & \text{Dual Sessions} \\ \mathcal{V} = \{x, y, z, \dots\} & \text{Variables} & \mathcal{PV} = \{X, Y, Z, \dots\} & \text{Process Variables} \\ \mathcal{R} = \{r, r_1, \dots\} & \text{Recursive Variables} & & \end{array}$$

with the set of names $\mathcal{N} = S \cup \overline{S}$ and let $k \in \mathcal{N} \cup \mathcal{V}$. Also for convenience we sometimes denote shared names with $a, b, \dots$ although $a \in N$.

Processes The syntax of processes follows:

$$\begin{array}{lll} P & ::= & k!\langle k' \rangle; P \mid k?(x); P \\ & \mid & k!\langle (x)Q \rangle; P \mid k?(X); P \\ & \mid & s \triangleleft l; P \mid s \triangleright \{l_i : P_i\}_{i \in I} \mid P_1 \mid P_2 \mid (\nu\ s)P \mid \mathbf{0} \mid r \mid \mu r.P \end{array}$$

1.2 Reduction Relation

Structural Congruence

$$P \mid \mathbf{0} \equiv P \qquad P_1 \mid P_2 \equiv P_2 \mid P_1 \qquad P_1 \mid (P_2 \mid P_3) \qquad (P_1 \mid P_2) \mid P_3 \qquad (\nu\ s)\mathbf{0} \equiv \mathbf{0}$$
$$s \notin \mathtt{fn}(P_1) \Rightarrow P_1 \mid (\nu\ s)P_2 \equiv (\nu\ s)(P_1 \mid P_2)$$

Process Variable Substitution

$$\begin{array}{rcl} (s!\langle (y)P_1 \rangle; P_2)\{(x)Q/X\} & = & s!\langle (y)P_1\{(x)Q/X\} \rangle; (P_2\{(x)Q/X\}) \\ (s?(Y); P)\{(x)Q/X\} & = & s?(Y); (P\{(x)Q/X\}) \\ (s \triangleleft l; P)\{(x)Q/X\} & = & s \triangleleft l; (P\{(x)Q/X\}) \\ (s \triangleright \{l_i : P_i\}_{i \in I})\{(x)Q/X\} & = & s \triangleright \{l_i : P_i\{(x)Q/X\}\}_{i \in I} \\ (P_1 \mid P_2)\{(x)Q/X\} & = & P_1\{(x)Q/X\} \mid P_2\{(x)Q/X\} \\ ((\nu\ s)P)\{(x)Q/X\} & = & (\nu\ s)(P\{(x)Q/X\}) \\ \mathbf{0}\{(x)Q/X\} & = & \mathbf{0} \\ X\langle k \rangle\{(x)Q/X\} & = & Q\{k/x\} \end{array}$$

Operational Semantics

$$
\begin{array}{rcl}
& & s!\langle (x)P\rangle; P_1 \mid s?(X); P_2 \longrightarrow P_1 \mid P_2\{(x)P/X\} \\
& & s!\langle s'\rangle; P_1 \mid s?(x); P_2 \longrightarrow P_1 \mid P_2\{s'/x\} \\
& & s \triangleleft l_k; P \mid s \triangleright \{l_i : P_i\}_{i\in I} \longrightarrow P \mid P_k \qquad k \in I \\
P_1 \longrightarrow P_1' & \Rightarrow & P_1 \mid P_2 \longrightarrow P_1' \mid P_2 \\
P \longrightarrow P' & \Rightarrow & (\nu\ s)P \longrightarrow (\nu\ s)P' \\
P \equiv\longrightarrow\equiv P' & \Rightarrow & P \longrightarrow P'
\end{array}
$$

1.3 Subcalculi

We identify two subcalculi of the Higher Order Session Calculus:

1. pure HO uses only the semantics that allow abstraction passing.
2. session π uses only the semantics that allow name passing.

Later in this paper we will identify a third typed subcalculi derived from session π that is defined on the non-usage of shared sessions.

Proposition 1.1 (Normalisation). *Let P a Higher Orser Session Calculus process, then $P \equiv (\nu\ \tilde{s})(P_1 \mid \ldots \mid P_n)$ with $P_1, \ldots, P_n$ session prefixed processes, recursion $\mu r.P$ or application process $X\langle k\rangle$.*

Proof. The proof is a simple induction on the syntax of P.

2 Types

2.1 Session Types

$$
\begin{array}{rcl}
H & ::= & T \multimap \diamond \quad | \quad T \to \diamond \\
U & ::= & H \quad | \quad T \quad | \quad \langle T\rangle \\
T & ::= & \mathsf{end} \quad | \quad \mu \mathsf{t}.T \quad | \quad \mathsf{t} \quad | \quad !\langle U\rangle;T \quad | \quad ?(U);T \quad | \quad \oplus\{l_i : T_i\}_{i\in I} \quad | \quad \&\{l_i : T_i\}_{i\in I}
\end{array}
$$

2.2 Subtyping

Definition 2.1 (Session Subtyping). *Let $\mathcal{T}$ to be the set of all session types. Define the monotone function $F : \mathcal{T} \longrightarrow \mathcal{T}$:*

$$
\begin{array}{rl}
F(R) = & \{\mathsf{end}, \mathsf{end}\} \cup \{\langle T\rangle, \langle T\rangle\} \\
& \cup\ \{(T \to \diamond, T \to \diamond)\} \cup \{(T \multimap \diamond, T \multimap \diamond)\} \cup \{(T \multimap \diamond, T \to \diamond)\} \\
& \cup\ \{(!\langle U_1\rangle; T_1, !\langle U_2\rangle; T_2) \ \mid\ T_1\ R\ T_2, U_2\ R\ U_1\} \\
& \cup\ \{(?(U_1); T_1, ?(U_2); T_2) \ \mid\ T_1\ R\ T_2, U_1\ R\ U_2\} \\
& \cup\ \{(\oplus\{l_i : T_i\}_{i\in I}, \oplus\{l_i : T_i'\}_{i\in I}) \ \mid\ \forall k \in I, T_k\ R\ T_k'\} \\
& \cup\ \{(\&\{l_i : T_i\}_{i\in I}, \&\{l_i : T_i'\}_{i\in I}) \ \mid\ \forall k \in I, T_k\ R\ T_k'\} \\
& \cup\ \{(\mu \mathsf{t}.T_1, \mu \mathsf{t}.T_2) \ \mid\ T_1\ R\ T_2\} \\
& \cup\ \{(T_1\{\mu \mathsf{t}.T_1/t\}, \mu \mathsf{t}.T_2) \ \mid\ T_1\ R\ \mu \mathsf{t}.T_2\} \cup \{(\mu \mathsf{t}.T_1, T_2\{\mu \mathsf{t}.T_2/t\}) \ \mid\ \mu \mathsf{t}.T_1\ R\ T_2\}
\end{array}
$$

$\leq = \nu R.F(R)$.

2.3 Duality

Definition 2.2 (Session Duality). *Define the monotone function $F : \mathcal{T} \longrightarrow \mathcal{T}$:*

$$
\begin{aligned}
F(R) = \{&\mathsf{end}, \mathsf{end}\} \\
&\cup \{(!\langle U_1 \rangle; T_1, ?(U_2); T_2) \mid T_1 \mathrel{R} T_2, U_2 \leq U_1\} \\
&\cup \{(?(U_1); T_1, !\langle U_2 \rangle; T_2) \mid T_1 \mathrel{R} T_2, U_1 \leq U_2\} \\
&\cup \{(\oplus\{l_i : T_i\}_{i \in I}, \&\{l_i : T'_i\}_{i \in I}) \mid \forall k \in I, T_k \mathrel{R} T'_k\} \\
&\cup \{(\&\{l_i : T_i\}_{i \in I}, \oplus\{l_i : T'_i\}_{i \in I}) \mid \forall k \in I, T_k \mathrel{R} T'_k\} \\
&\cup \{(\mu \mathsf{t}.T_1, \mu \mathsf{t}.T_2) \mid T_1 \mathrel{R} T_2\} \\
&\cup \{(T_1\{\mu \mathsf{t}.T_1/t\}, \mu \mathsf{t}.T_2) \mid T_1 \mathrel{R} \mu \mathsf{t}.T_2\} \cup \{(\mu \mathsf{t}.T_1, T_2\{\mu \mathsf{t}.T_2/t\}) \mid \mu \mathsf{t}.T_1 \mathrel{R} T_2\}
\end{aligned}
$$

$\mathsf{dual} = \nu R.F(R)$.

2.4 Typing System

$$\Gamma ::= \Gamma \cdot X : H \mid \Gamma \cdot r : \Delta \mid \emptyset$$

$$\Delta ::= \Delta \cdot k : T \mid \Delta \cdot k : \langle T \rangle \mid \Delta \cdot X \mid \emptyset$$

$$\Gamma \vdash P \triangleright \Delta$$

$$\Gamma \cdot X : H \vdash X \triangleright H \quad \text{[PVar]} \qquad \Gamma \cdot r : \Delta \vdash r \triangleright \Delta \quad \text{[RVar]}$$

$$\Gamma \vdash \mathbf{0} \triangleright \emptyset \quad \text{[Inact]} \qquad \frac{\Gamma \vdash P \triangleright \Delta \quad k \notin \mathsf{dom}(\Delta)}{\Gamma \vdash P \triangleright \Delta \cdot k : \mathsf{end}} \quad \text{[Comp]}$$

$$\Gamma \cdot X : T \multimap \diamond \vdash X\langle k \rangle \triangleright k : T \cdot X \quad \text{[LAppl]} \qquad \Gamma \cdot X : T \to \diamond \vdash X\langle k \rangle \triangleright k : T \quad \text{[SAppl]}$$

$$\frac{\Gamma \vdash P \triangleright \Delta \cdot k : T' \quad T' \leq T}{\Gamma \vdash P \triangleright \Delta \cdot k : T} \quad \text{[Subs]} \qquad \frac{\Gamma \vdash P_1 \triangleright \Delta_1 \quad \Gamma \vdash P_2 \triangleright \Delta_2 \quad \mathsf{dom}(\Delta_1) \cap \mathsf{dom}(\Delta_2) = \emptyset}{\Gamma \vdash P_1 \mid P_2 \triangleright \Delta_1 \cdot \Delta_2} \quad \text{[Par]}$$

$$\frac{\Gamma \vdash P \triangleright \Delta \quad k \notin \mathsf{dom}(\Delta)}{\Gamma \vdash k!\langle k' \rangle; P \triangleright \Delta \cdot k' : T \cdot k : \langle T \rangle} \quad \text{[Req]} \qquad \frac{\Gamma \vdash P \triangleright \Delta \cdot x : T' \quad k \notin \mathsf{dom}(\Delta) \quad T \ \mathsf{dual}\ T'}{\Gamma \vdash k?(x); P \triangleright \Delta \cdot k : \langle T \rangle} \quad \text{[Acc]}$$

$$\frac{\Gamma \vdash P \triangleright \Delta \cdot k : \langle T \rangle}{\Gamma \vdash k!\langle k' \rangle; P \triangleright \Delta \cdot k' : T \cdot k : \langle T \rangle} \quad \text{[ShReq]} \qquad \frac{\Gamma \vdash P \triangleright \Delta \cdot x : T' \cdot k : \langle T \rangle \quad T \ \mathsf{dual}\ T'}{\Gamma \vdash k?(x); P \triangleright \Delta \cdot k : \langle T \rangle} \quad \text{[ShAcc]}$$

$$\frac{\Gamma \vdash P \triangleright \Delta \cdot k : T}{\Gamma \vdash k!\langle k' \rangle; P \triangleright \Delta \cdot k : !\langle T' \rangle; T \cdot k' : T'} \quad \text{[NOut]} \qquad \frac{\Gamma \vdash P \triangleright \Delta \cdot k : T \cdot x : T'}{\Gamma \vdash k?(x); P \triangleright \Delta \cdot k : ?(T'); T} \quad \text{[NIn]}$$

$$\frac{\Gamma \vdash Q \triangleright x : T' \quad \Gamma \vdash P \triangleright \Delta \cdot k : T}{\Gamma \vdash k!\langle (x)Q \rangle; P \triangleright \Delta \cdot k : !\langle T' \to \diamond \rangle; T} \quad \text{[SHOut]}$$

$$\frac{\Gamma \vdash Q \triangleright \Delta_1 \cdot x : T' \quad \Gamma \vdash P \triangleright \Delta_2 \cdot k : T \quad \Delta_1 \neq \emptyset \quad \mathsf{dom}(\Delta_1) \cap \mathsf{dom}(\Delta_2 \cdot k : T) = \emptyset}{\Gamma \vdash k!\langle (x)Q \rangle; P \triangleright \Delta_1 \cdot \Delta_2 \cdot k : !\langle T' \multimap \diamond \rangle; T} \quad \text{[LHOut]}$$

$$\frac{\Gamma \cdot X : T' \to \diamond \vdash P \triangleright \Delta \cdot k : T \quad X \notin \Delta}{\Gamma \vdash k?(X); P \triangleright \Delta \cdot k : ?(T' \to \diamond); T} \quad \text{[SHIn]}$$

$$\frac{\Gamma \cdot X : T' \multimap \diamond \vdash P \triangleright \Delta \cdot k : T \cdot X}{\Gamma \vdash k?(X); P \triangleright \Delta \cdot k : ?(T' \multimap \diamond); T} \quad \text{[LHIn]}$$

$$\frac{\Gamma \vdash P \triangleright \Delta \cdot s : T_1 \cdot \overline{s} : T_2 \quad T_1 \ \mathsf{dual}\ T_2}{\Gamma \vdash (\nu\ s)P \triangleright \Delta} \quad \text{[NRes]} \qquad \frac{\Gamma \vdash P \triangleright \Delta \cdot a : T}{\Gamma \vdash (\nu\ a)P \triangleright \Delta} \quad \text{[ShRes]}$$

$$\frac{\Gamma \vdash P \triangleright \Delta \cdot k : T}{\Gamma \vdash P \triangleright \Delta \cdot k : \oplus\{l : T\}} \quad \text{[Sel]} \qquad \frac{\forall i \in I, \Gamma \vdash P_i \triangleright \Delta \cdot k : T_i}{\Gamma \vdash k \triangleright \{l_i : P_i\}_{i \in I} \triangleright \Delta \cdot k : \&\{l_i : T_i\}_{i \in I}} \quad \text{[Bra]}$$

$$\frac{\Gamma \vdash P \triangleright \Delta \qquad \Gamma \vdash r \triangleright \Delta}{\Gamma \vdash \mu r.P \triangleright \Delta} \quad \text{[Repl]}$$

Definition 2.3 (pure session π). *Let P be a* session π *process with $\Gamma \vdash P \triangleright \Delta$. If the typing derivation does not use the rules* [*Req*], [*Acc*], [*ShReq*], [*ShAcc*], [*ShRes*] *then P is a* pure session π *process.*

2.5 Examples

Example 2.1.

$$P = s?(X);(X\langle s_1\rangle \mid X\langle s_2\rangle)$$

is untypable under environment $\Gamma = X : T \multimap \diamond$, since:

$$\Gamma \vdash X\langle s_1\rangle \triangleright s_1 : T \cdot X \quad \Gamma \vdash X\langle s_2\rangle \triangleright s_2 : T \cdot X$$

We cannot apply rule [Par] to get:

$$\Gamma \vdash X\langle s_1\rangle \mid X\langle s_2\rangle \triangleright s_1 : T \cdot s_2 : T$$

because $\mathtt{dom}(s_1 : T \cdot X) \cap \mathtt{dom}(s_2 : T \cdot X) = X$.

It is though typable under environment $\Gamma = X : T \rightarrow \diamond$, since:

$$\dfrac{\dfrac{\Gamma \vdash X\langle s_1\rangle \triangleright s_1 : T \quad \Gamma \vdash X\langle s_2\rangle \triangleright s_2 : T \quad \mathtt{dom}(s_1 : T) \cup \mathtt{dom}(s_2 : T) = \emptyset}{\Gamma \vdash X\langle s_1\rangle \mid X\langle s_2\rangle \triangleright s_1 : T \cdot s_2 : T}}{\vdash s?(X);(X\langle s_1\rangle \mid X\langle s_2\rangle) \triangleright s :?(T \rightarrow \diamond);\mathtt{end} \cdot s_1 : T \cdot s_2 : T}$$

Now let

$$Q_1 = \overline{s}!\langle (x)x!\langle \mathbf{0}\rangle;\mathbf{0}\rangle;\mathbf{0}$$
$$Q_2 = \overline{s}!\langle (x)x!\langle \mathbf{0}\rangle;s'!\langle \mathbf{0}\rangle;\mathbf{0}\rangle;\mathbf{0}$$

Process $(\nu\ s)(Q_1 \mid P)$ is typable, whereas $(\nu\ s)(Q_2 \mid P)$ is not. This is due to the fact that abstraction $(x)x!\langle \mathbf{0}\rangle;s'!\langle \mathbf{0}\rangle;\mathbf{0}$ contains linear session s' and should not be duplicated:

$$P \mid Q_2 \longrightarrow s_1!\langle \mathbf{0}\rangle;s'!\langle \mathbf{0}\rangle;\mathbf{0} \mid s_2!\langle \mathbf{0}\rangle;s'!\langle \mathbf{0}\rangle;\mathbf{0}$$

The last process should not be typable because name s' is appeared twice.

The type system avoids the above situation on rule [Out] and the duality relation:

$$\dfrac{\vdash x!\langle \mathbf{0}\rangle;s'!\langle \mathbf{0}\rangle;\mathbf{0} \triangleright x : T \cdot s' : T' \vdash \mathbf{0} \triangleright \overline{s} : \mathtt{end}}{\vdash Q_2 \triangleright s' : T' \cdot \overline{s} :!\langle T \multimap \diamond\rangle;\mathtt{end}}$$

We then apply rule [Par] to get:

$$\vdash P \mid Q_2 \triangleright s' : T' \cdot \overline{s} :!\langle T \multimap \diamond\rangle;\mathtt{end} \cdot s :?(T \rightarrow \diamond);\mathbf{0} s_1 : T \cdot s_2 : T$$

On this typing node, rule [Res] is not applicable since $!\langle T \multimap \diamond\rangle;\mathtt{end}$ is not dual with $?(T \rightarrow \diamond);\mathbf{0}$.

2.6 Soundness

Definition 2.4 (Environment Reduction).

1. $\Delta \cdot s :!\langle U\rangle;T_1 \cdot \overline{s} :?(U);T_2 \longrightarrow \Delta \cdot s : T_1 \cdot \overline{s} : T_2$
2. $\Delta \cdot s : \oplus\{l_i : T_i\}_{i\in I} \cdot \overline{s} : \oplus\{l_i : T_i'\}_{i\in I} \longrightarrow \Delta \cdot s : T_k \cdot \overline{s} : T_k'$, $k \in I$.

Definition 2.5 (Well Typed Environment). *Environment Δ is well typed if whenever $s : T_1, \overline{s} : T_2 \in \Delta$ then T_1* dual *T_2.*

Lemma 2.1 (Substitution). *Jorge TODO*

Theorem 2.1. *Let $\Gamma \vdash P \triangleright \Delta$ with Δ well typed. If $P \longrightarrow P'$ then $\Gamma \vdash P \triangleright \Delta'$ and $\Delta \longrightarrow \Delta'$ or $\Delta = \Delta'$.*

3 Observational Semantics

We give the observational semantics for the pure HO.

3.1 Labelled Transition Semantics

$$
\begin{array}{lcl}
\lambda & ::= & \tau \mid s!\langle (x)P \rangle \mid s?\langle (x)P \rangle \mid s \oplus l \mid s \& l \mid o \\
o & ::= & (\nu\, s)s!\langle (x)P \rangle \mid (\nu\, s)o
\end{array}
$$

$$
\begin{array}{c}
\mathtt{fn}(s \oplus l) = \mathtt{fn}(s \& l) = \{s\} \qquad \mathtt{fn}(\tau) = \emptyset \\
\mathtt{fn}(s!\langle (x)P \rangle) = \mathtt{fn}(s!\langle (x)P \rangle) = \{s\} \cup \mathtt{fn}((x)P) \\
\mathtt{bn}(\tau) = \mathtt{bn}(s \oplus l) = \mathtt{bn}(s \& l) = \mathtt{bn}(s?\langle (x)P \rangle) = \emptyset \\
\mathtt{bn}((\nu\, \tilde{s})s!\langle (x)P \rangle) = \tilde{s}
\end{array}
$$

$$
s \oplus l \asymp s \& l \qquad (\nu\, \tilde{s})s!\langle (x)P \rangle \asymp s?\langle (x)P \rangle
$$

$$
s!\langle (x)Q \rangle; P \stackrel{s!\langle (x)Q \rangle}{\longrightarrow} P \qquad s?(X); P \stackrel{s?\langle (x)Q \rangle}{\longrightarrow} P\{(x)Q/X\}
$$

$$
s \triangleleft l; P \stackrel{s \oplus l}{\longrightarrow} P \qquad s \triangleright \{l_i : P_i\}_{i \in I} \stackrel{s \& l_k}{\longrightarrow} P_k \quad k \in I
$$

$$
\frac{P \stackrel{\lambda}{\longrightarrow} P' \quad s \notin \mathtt{fn}(\lambda)}{(\nu\, s)P \stackrel{\lambda}{\longrightarrow} (\nu\, s)P'} \qquad \frac{P \stackrel{(\nu\, \tilde{s})s!\langle (x)Q \rangle}{\longrightarrow} P' \quad s' \in \mathtt{fn}((x)Q)}{(\nu\, s')P \stackrel{(\nu\, s' \cdot \tilde{s})s!\langle (x)Q \rangle}{\longrightarrow} P'}
$$

$$
\frac{P \stackrel{\lambda}{\longrightarrow} P' \quad \mathtt{bn}(\lambda) \cap \mathtt{fn}(Q) = \emptyset}{P \mid Q \stackrel{\lambda}{\longrightarrow} P' \mid Q} \qquad \frac{Q \stackrel{\lambda}{\longrightarrow} Q' \quad \mathtt{bn}(\lambda) \cap \mathtt{fn}(P) = \emptyset}{P \mid Q \stackrel{\lambda}{\longrightarrow} P \mid Q'}
$$

$$
\frac{P \stackrel{\lambda_1}{\longrightarrow} P' \qquad Q \stackrel{\lambda_2}{\longrightarrow} Q'}{P \mid Q \stackrel{\tau}{\longrightarrow} (\nu\, \mathtt{bn}(\lambda_1) \cup \mathtt{bn}(\lambda_2))(P' \mid Q')} \qquad \frac{P \equiv_\alpha P'' \quad P'' \stackrel{\lambda}{\longrightarrow} P'}{P \stackrel{\lambda}{\longrightarrow} P'}
$$

3.2 LTS for Types

$$
\frac{\overline{s} \notin \mathtt{dom}(\Delta) \quad \Gamma \vdash P \triangleright \Delta' \cdot x : T' \quad H = T' \to \diamond \vee H = T' \multimap \diamond}{(\Gamma, \Delta \cdot s : !\langle H \rangle; T) \stackrel{s!\langle (x)P \rangle}{\longrightarrow} (\Gamma, \Delta \cdot s : T)}
$$

$$
\frac{\overline{s} \notin \mathtt{dom}(\Delta) \quad \Gamma \vdash P \triangleright \Delta' \cdot x : T' \quad H = T' \to \diamond \vee H = T' \multimap \diamond}{(\Gamma, \Delta \cdot s : ?(H); T) \stackrel{s?\langle (x)P \rangle}{\longrightarrow} (\Gamma, \Delta \cdot s : T)}
$$

$$
\frac{\overline{s} \notin \mathtt{dom}(\Delta) \quad k \in I}{(\Gamma, \Delta \cdot s : \oplus\{l_i : T_i\}_{i \in I}) \stackrel{s \oplus l_k}{\longrightarrow} (\Gamma, \Delta \cdot s : T_k)} \quad \frac{\overline{s} \notin \mathtt{dom}(\Delta) \quad k \in I}{(\Gamma, \Delta \cdot s : \&\{l_i : T_i\}_{i \in I}) \stackrel{s \& l_k}{\longrightarrow} (\Gamma, \Delta \cdot s : T_k)}
$$

$$
\frac{(\Gamma, \Delta) \stackrel{(\nu\, \tilde{s})s!\langle (x)P \rangle}{\longrightarrow} (\Gamma, \Delta') \quad \Gamma \vdash P \triangleright \Delta'' \cdot s' : T}{(\Gamma, \Delta) \stackrel{(\nu\, s' \cdot \tilde{s}')s!\langle (x)P \rangle}{\longrightarrow} (\Gamma, \Delta \cdot s' : T)} \qquad \frac{\Delta \longrightarrow \Delta'}{(\Gamma, \Delta) \stackrel{\tau}{\longrightarrow} (\Gamma, \Delta')}
$$

Definition 3.1 (Typed Transition). *We define*

$$\Gamma \vdash P \triangleright \Delta \xrightarrow{\lambda} P' \triangleright \Delta'$$

if

1. $P \xrightarrow{\lambda} P'$
2. $(\Gamma, \Delta) \xrightarrow{\lambda} (\Gamma, \Delta')$

3.3 Barbed Congruence

Definition 3.2 (Barbs). *Let* pure HO *process* P.

1. *We write* $P \downarrow_s$ *if* $P \equiv (\nu\ \tilde{s})(s!\langle (x)P_1 \rangle; P_2 \mid P_3), s \notin \tilde{s}$. *We write* $P \Downarrow_s$ *if* $P \longrightarrow^* \downarrow_s$.
2. *We write* $\Gamma \vdash P \triangleright \Delta \downarrow_s$ *if* $P \downarrow_s$ *and* $\overline{s} \notin \Delta$. *We write* $\Gamma \vdash P \triangleright \Delta \Downarrow_s$ *if* $\Gamma \vdash P \triangleright \Delta \Longrightarrow P' \triangleright \Delta' \downarrow_s$.

Definition 3.3 (Context). *C is a context defined on the grammar:*
$C = - \mid P \mid k!\langle (x)P \rangle; C \mid k?(X); C \mid (\nu\ s)C \mid C \mid C \mid k \triangleleft l; C \mid k \triangleright \{l_i : C_i\}_{i \in I}$
Notation $C[P]$ *replaces every* $-$ *in* C *with* P.

Definition 3.4 (Typed Congruence). *Relation* $\Gamma \vdash P_1 \triangleright \Delta_1\ R\ P_2 \triangleright \Delta_2$ *is a typed congruence if* $\forall C$ *such that* $\Gamma \vdash C[P_1] \triangleright \Delta_1'$ *and* $\Gamma \vdash C[P_2] \triangleright \Delta_2'$ *then* $\Gamma \vdash C[P_1] \Delta_1'\ R\ C[P_2] \triangleright \Delta_2'$.

Definition 3.5 (Barbed Congruence). *Relation* $\Gamma \vdash P_1 \triangleright \Delta_1\ R\ P_2 \triangleright \Delta_2$ *is a barbed congruence whenever:*

1. – *If* $P_1 \longrightarrow P_1'$ *then* $\exists P_2', P_2 \longrightarrow^* P_1'$ *and* $\Gamma \vdash P_1' \triangleright \Delta_1'\ R\ P_2' \triangleright \Delta_2'$.
   – *If* $P_2 \longrightarrow P_2'$ *then* $\exists P_1', P_1 \longrightarrow^* P_1'$ *and* $\Gamma \vdash P_1' \triangleright \Delta_1'\ R\ P_2' \triangleright \Delta_2'$.
2. – *If* $\Gamma \vdash P_1 \triangleright \Delta_1 \downarrow_s$ *then* $\Gamma \vdash P_2 \triangleright \Delta_2 \Downarrow_s$.
   – *If* $\Gamma \vdash P_2 \triangleright \Delta_2 \downarrow_s$ *then* $\Gamma \vdash P_1 \triangleright \Delta_1 \Downarrow_s$.
3. *R is a typed congruence.*

The largest such congruence is denote with $\cong$.

3.4 Bisimulation

Definition 3.6 (Barbed congruence). Let relation $\mathcal{R}$ such that $\Gamma \vdash P_1 \triangleright \Delta \mathcal{R} Q_1 \triangleright \Delta$. $\mathcal{R}$ is a barbed congruence if whenever:

- $\forall (\nu\ \tilde{s})s!\langle (x)P \rangle$ such that $\Gamma \vdash P_1 \triangleright \Delta \xrightarrow{(\nu\ \tilde{s})s!\langle (x)P \rangle} P_2 \triangleright \Delta', \exists Q_2$ such that $\Gamma \vdash Q_1 \triangleright \Delta \xrightarrow{(\nu\ \tilde{s})s!\langle (x)Q \rangle} Q_2 \triangleright \Delta'$ and $\forall C, s'$ such that $\Gamma \vdash (\nu\ \tilde{s})(P_2 \mid C[P\{s'/x\}]) \triangleright \Delta''$ and $\Gamma \vdash (\nu\ \tilde{s})(Q_2 \mid C[Q\{s'/x\}]) \triangleright \Delta''$ then $\Gamma \vdash (\nu\ \tilde{s})(P_2 \mid C[P\{s'/x\}]) \triangleright \Delta'' \mathcal{R} (\nu\ \tilde{s})(Q_2 \mid C[Q\{s'/x\}]) \triangleright \Delta''$.
- $\forall \lambda \neq (\nu\ \tilde{s})s!\langle (x)P \rangle$ such that $\Gamma \vdash P_1 \triangleright \Delta \xrightarrow{\lambda} P_2 \triangleright \Delta', \exists Q_2$ such that $\Gamma \vdash Q_1 \triangleright \Delta \xrightarrow{\lambda} Q_2 \triangleright \Delta'$ and $\Gamma \vdash P_2 \triangleright \Delta' \mathcal{R} Q_2 \triangleright \Delta'$.
- The symmetric cases of 1 and 2.

The largest barbed congruence is denoted by $\approx^c$.

Definition 3.7 (Bisimulation). Let relation $\mathcal{R}$ such that $\Gamma \vdash P_1 \triangleright \Delta \mathcal{R} Q_1 \triangleright \Delta$. $\mathcal{R}$ is a bisimulation if whenever:

- $\forall (\nu\ \tilde{s}) s!\langle (x)P\rangle$ such that $\Gamma \vdash P_1 \triangleright \Delta \stackrel{(\nu\ \tilde{s}) s!\langle (x)P\rangle}{\longrightarrow} P_2 \triangleright \Delta', \exists Q_2$ such that $\Gamma \vdash Q_1 \triangleright \Delta \stackrel{(\nu\ \tilde{s}) s!\langle (x)Q\rangle}{\longrightarrow} Q_2 \triangleright \Delta'$ and s' such that $\Gamma \vdash (\nu\ \tilde{s})(P_2 \mid P\{s'/x\}) \triangleright \Delta''$ and $\Gamma \vdash (\nu\ \tilde{s})(Q_2 \mid Q\{s'/x\}) \triangleright \Delta''$ then $\Gamma \vdash (\nu\ \tilde{s})(P_2 \mid P\{s'/x\}) \triangleright \Delta'' \mathcal{R} (\nu\ \tilde{s})(Q_2 \mid Q\{s'/x\}) \triangleright \Delta''$.
- $\forall \lambda \neq (\nu\ \tilde{s}) s!\langle (x)P\rangle$ such that $\Gamma \vdash P_1 \triangleright \Delta \stackrel{\lambda}{\longrightarrow} P_2 \triangleright \Delta', \exists Q_2$ such that $\Gamma \vdash Q_1 \triangleright \Delta \stackrel{\lambda}{\longrightarrow} Q_2 \triangleright \Delta'$ and $\Gamma \vdash P_2 \triangleright \Delta' \mathcal{R} Q_2 \triangleright \Delta'$.
- The symmetric cases of 1 and 2.

The largest barbed congruence is denoted by $\approx$.

Theorem 3.1.
- *$\approx^c$ is a congruence.*
- *$\cong$ implies $\approx^c$*
- *$\approx^c = \approx$*

4 Encoding

Before we proceed with encodings we define some properties that encodings may respect:

Definition 4.1. *Given a mapping $[\![\cdot]\!] : L_1 \longrightarrow L_2$ we define the following:*

1. *Operational Correspondence.*
   - *$P \longrightarrow Q$ implies $[\![P]\!] \longrightarrow^* [\![Q]\!]$.*
   - *$[\![P]\!] \longrightarrow R$ implies $\exists Q$ such that $P \longrightarrow Q$ and $R \longrightarrow^* [\![Q]\!]$.*
2. *Typability. If $\Gamma \vdash P \triangleright \Delta$ then $\Gamma \vdash [\![P]\!] \triangleright \Delta'$.*
3. *$\mid$-preservation. $[\![P \mid Q]\!] = [\![P]\!] \mid [\![Q]\!]$.*
4. *Full Abstraction. $P \cong Q$ if and only if $[\![P]\!] \cong [\![Q]\!]$.*

4.1 Encode the non-recursive pure session π into pure HO

In this section provide an encoding of the pure session π with no recursion into the pure HO.

$$[\![k!\langle k'\rangle; P]\!] \stackrel{\mathsf{def}}{=} k!\langle (z)z?(X); X\langle k'\rangle\rangle; [\![P]\!]$$
$$[\![k?(x); P]\!] \stackrel{\mathsf{def}}{=} k?(X); (\nu\ s)(X\langle s\rangle \mid \overline{s}!\langle (x)[\![P]\!]\rangle; \mathbf{0})$$

The rest of the operators, except the recursive constructs, are encoded in an isomorphic way:

$$[\![\mathbf{0}]\!] \stackrel{\mathsf{def}}{=} \mathbf{0} \qquad [\![P \mid Q]\!] \stackrel{\mathsf{def}}{=} [\![P]\!] \mid [\![Q]\!] \qquad [\![(\nu\ s)P]\!] \stackrel{\mathsf{def}}{=} (\nu\ s)[\![P]\!]$$
$$[\![k \triangleleft l; P]\!] \stackrel{\mathsf{def}}{=} k \triangleleft l; [\![P]\!] \qquad [\![k \triangleright \{l_i : P_i\}_{i\in I}]\!] \stackrel{\mathsf{def}}{=} k \triangleright \{l_i : [\![P_i]\!]\}_{i\in I}$$

We can also encode the polyadic version of the send and receive primitives.

$$[\![k!\langle k' \cdot \tilde{k}\rangle; P]\!] \stackrel{\mathsf{def}}{=} [\![k!\langle k'\rangle; k!\langle \tilde{k}\rangle; P]\!]$$
$$[\![k?(x \cdot \tilde{x}); P]\!] \stackrel{\mathsf{def}}{=} [\![k?(x); k?(\tilde{x}); P]\!]$$

Unlike the classic π calculus we do not need to create a new channel because typable terms quarranty no race conditions on the two session endpoints.

4.2 Extend the pure HO

We extend the pure HO with process variable abstraction and process variable application, as well as polyadic abstractions and polyadic applications to define the pure HO^+ (pure Higher Order plus) calculus. We show that all of the constructs are encodable in the pure HO.

$$
\begin{array}{rcll}
P & ::= & k!\langle (X)P_1 \rangle; P_2 & \text{Process Abstraction} \\
& \mid & X\langle (x)P \rangle & \text{Process Application} \\
& \mid & k!\langle (\tilde{x})P_1 \rangle; P_2 & \text{Polyadic Abstraction} \\
& \mid & X\langle \tilde{k} \rangle & \text{Polyadic Application}
\end{array}
$$

Operational Semantics In order to define the operational semantics of the pure HO^+, we extend the operational semantics of pure HO with the rules:

$$
\begin{array}{c}
s!\langle (Y)P \rangle; P_1 \mid s?(X); X\langle (x)P_2 \rangle \longrightarrow P_1 \mid P\{(x)P_2/Y\} \\
s!\langle (\tilde{x})P_1 \rangle; P_2 \mid s?(X); X\langle \tilde{k} \rangle \longrightarrow P_2 \mid P_1\{\tilde{k}/\tilde{x}\}
\end{array}
$$

Encoding of pure HO^+ to pure HO

$$
\begin{array}{rcl}
[\![k!\langle (X)Q \rangle; P]\!] & \stackrel{\mathsf{def}}{=} & k!\langle (z)z?(X); [\![Q]\!] \rangle; [\![P]\!] \\
[\![X\langle (x)P \rangle]\!] & \stackrel{\mathsf{def}}{=} & (\nu\ s)(X\langle s \rangle \mid \overline{s}!\langle (x)[\![P]\!] \rangle; \mathbf{0}) \\
[\![k!\langle (\tilde{x})P_1 \rangle; P_2]\!] & \stackrel{\mathsf{def}}{=} & k!\langle (z)[\![z?(\tilde{x}); P_1]\!] \rangle; [\![P_2]\!] \\
[\![X\langle \tilde{k} \rangle]\!] & \stackrel{\mathsf{def}}{=} & (\nu\ s)(X\langle s \rangle \mid [\![\overline{s}!\langle \tilde{k} \rangle; \mathbf{0}]\!])
\end{array}
$$

We are not ready yet to encode recursion. In an iterative process we require subject abstractions to be non-linear due to the fact that the receiver should apply an abstraction more than once to achieve iteration, i.e. as we have seen in Example 2.1 a process:

$$
s!\langle ()P \rangle; P_1 \mid s?(X); (X\langle\rangle \mid X\langle\rangle)
$$

with $\mathtt{fs}(P) \neq \emptyset$ is not typable, since abstraction $()P$ can only be applied in a linear way.

Encode linear pure HO processes into non-linear pure HO abstractions. Therefore it is convenient to have an encoding from a process to an abstraction with no free names, that can be used a shared value:

$$
\begin{array}{c}
\mathcal{A}[\![\cdot]\!] : \mathcal{P} \longrightarrow \mathcal{V} \\
\mathcal{A}[\![P]\!] \ ::= \ ((\!| \mathtt{fn}(P) |\!)^v)\mathcal{A}[\![P]\!]^{\emptyset}
\end{array}
$$

where

Function $(\!| \cdot |\!)^s : 2^{\mathcal{N}} \longrightarrow \mathcal{N}^{\omega}$ orders lexicographically a set of names, function $(\!| \cdot |\!)^v : 2^{\mathcal{N}} \longrightarrow \mathcal{V}^{\omega}$ maps a set of names to variables:

$$
\begin{array}{rcl}
(\!| \{s_i\}_{i\in I} |\!)^v & = & (\!| (\!| \{s_i\}_{i\in I} |\!)^s |\!)^{s\to v} \\
(\!| s \cdot \tilde{s} |\!)^{s\to v} & = & x_s \cdot (\!| \tilde{s} |\!)^{s\to v} \\
(\!| s |\!)^{s\to v} & = & x_s
\end{array}
$$

$$
\begin{array}{rcl}
\mathcal{A}[\![s!\langle (x)P'\rangle;P]\!]^{\sigma} & ::= & \begin{cases} x_s!\langle ((\!|x|\!)^{v}P)\mathcal{A}[\![P']\!]^{\emptyset}\rangle;\mathcal{A}[\![P]\!]^{\sigma} & s\notin\sigma \\ s!\langle ((\!|x|\!)^{v}P)\mathcal{A}[\![P']\!]^{\emptyset}\rangle;\mathcal{A}[\![P]\!]^{\sigma} & s\in\sigma \end{cases} \\
\mathcal{A}[\![s?(X);P]\!]^{\sigma} & ::= & \begin{cases} x_s?(X);\mathcal{A}[\![P]\!]^{\sigma} & s\notin\sigma \\ s?(X);\mathcal{A}[\![P]\!]^{\sigma} & s\in\sigma \end{cases} \\
\mathcal{A}[\![s\triangleleft l;P]\!]^{\sigma} & ::= & \begin{cases} x_s\triangleleft l;\mathcal{A}[\![P]\!]^{\sigma} & s\notin\sigma \\ s\triangleleft l;\mathcal{A}[\![P]\!]^{\sigma} & s\in\sigma \end{cases} \\
\mathcal{A}[\![s\triangleright\{l_i:P_i\}_{i\in I}]\!]^{\sigma} & ::= & \begin{cases} x_s\triangleright\{l_i:\mathcal{A}[\![P_i]\!]^{\sigma}\}_{i\in I} & s\notin\sigma \\ s\triangleright\{l_i:\mathcal{A}[\![P_i]\!]^{\sigma}\}_{i\in I} & s\in\sigma \end{cases} \\
\mathcal{A}[\![P_1\mid P_2]\!]^{\sigma} & ::= & \mathcal{A}[\![P_1]\!]^{\sigma}\mid\mathcal{A}[\![P_2]\!]^{\sigma} \qquad s\notin\sigma \\
\mathcal{A}[\![(\nu\ s)P]\!]^{\sigma} & ::= & (\nu\ s)\mathcal{A}[\![P]\!]^{\sigma\cdot s} \\
\mathcal{A}[\![\mathbf{0}]\!]^{\sigma} & ::= & \mathbf{0} \\
\mathcal{A}[\![X\langle s\rangle]\!]^{\sigma} & ::= & \begin{cases} X\langle x_s\rangle & s\notin\sigma \\ X\langle s\rangle & s\in\sigma \end{cases}
\end{array}
$$

A basic property of the $\mathcal{A}[\![\cdot]\!]$ function is the restoration of the original process when we apply its free names to the resulting abstraction.

Proposition 4.1. *Let P be a* pure HO *process, then*

$$(\nu\ s)(s?(X);X\langle (\!|P|\!)^{s}\rangle \mid \overline{s}!\langle\mathcal{A}[\![P]\!]\rangle;\mathbf{0}) \longrightarrow P$$

Proof. doit

Encode Recursion We are ready now to encode Recursion.

$$
\begin{array}{rcl}
[\![\mu r.P]\!] & = & (\nu\ s)(s?(X);[\![P]\!]\mid\overline{s}!\langle (z\cdot(\!|\mathtt{fn}(P)|\!)^{v})z?(X);\mathcal{A}[\![P]\!]^{\emptyset}\rangle;\mathbf{0}) \\
[\![r]\!] & = & (\nu\ s)(X\langle s\cdot(\!|\mathtt{fn}(P)|\!)^{s}\rangle\mid\overline{s}!\langle (z\cdot(\!|\mathtt{fn}(P)|\!)^{v})X\langle z\cdot(\!|\mathtt{fn}(P)|\!)^{v}\rangle\rangle;\mathbf{0})
\end{array}
$$

A different process constructor for recursion is the constructor of replication:

$$*\,P$$

with

$$*\,P \equiv P \mid *\,P$$

We show that process constructors $\mu r.P$ can encode process constructor $*\,P$.

$$[\![*\,P]\!] \stackrel{\mathtt{def}}{=} \mu r.[\![P]\!]\mid r.$$

The other direction is encodable when P is guarded on a shared input:

$$[\![\mu r.a?(x);P]\!] \stackrel{\mathtt{def}}{=} *\,a?(x);[\![C[a!\langle x\rangle;\mathbf{0}]]\!]$$

where C being the context that results by replacing the recursive variable r with a $-$ in P.

4.3 Properties of the Encodings

Proposition 4.2 (Operational Correspondence). *Let P pure session π or a pure HO^+ process.*

1. *If $P \longrightarrow Q$ then $[\![P]\!] \longrightarrow^* [\![Q]\!]$.*
2. *If $[\![P]\!] \longrightarrow R$ then $\exists Q$ such that $P \longrightarrow Q$ and $R \longrightarrow^* [\![Q]\!]$.*

Proof. Part 1 is proved by induction on the reduction rules. The basic step consider all leaf reductions.

$$\begin{array}{rcl}
[\![s!\langle k'\rangle; P_1 \mid s?(x); P_2]\!] & ::= & s!\langle (z)z?(X); X\langle k'\rangle\rangle; [\![P_1]\!] \mid s?(X); (\nu\, s')(X\langle s'\rangle \mid \overline{s'}!\langle (x)[\![P_2]\!]\rangle; \mathbf{0}) \\
& \longrightarrow & [\![P_1]\!] \mid (\nu\, s')(s?(X); X\langle k'\rangle \mid \overline{s'}!\langle (x)[\![P_2]\!]\rangle; \mathbf{0}) \\
& \longrightarrow & [\![P_1]\!] \mid [\![P_2]\!]\{k'/x\}
\end{array}$$

$$\begin{array}{rcl}
[\![s!\langle (Y)P\rangle; P_1 \mid s?(X); X\langle (x)P_2\rangle]\!] & ::= & s!\langle (z)z?(Y); [\![P]\!]\rangle; [\![P_1]\!] \mid s?(X); (\nu\, s')(X\langle s'\rangle \mid \overline{s'}!\langle (x)[\![P_2]\!]\rangle; \mathbf{0}) \\
& \longrightarrow & [\![P_1]\!] \mid (\nu\, s')(s'?(Y); [\![P]\!] \mid \overline{s'}!\langle (x)[\![P_2]\!]\rangle; \mathbf{0}) \\
& \longrightarrow & [\![P_1]\!] \mid [\![P]\!]\{(x)[\![P_2]\!]/Y\}
\end{array}$$

$$\begin{array}{rcl}
[\![s!\langle (\tilde{x})P_1\rangle; P_2 \mid s?(X); X\langle \tilde{k}\rangle]\!] & ::= & s!\langle (z)[\![z?(\tilde{x}); P_1]\!]\rangle; [\![P_2]\!] \mid s?(X); (\nu\, s')(X\langle s'\rangle \mid [\![\overline{s}!\langle \tilde{k}\rangle; \mathbf{0}]\!]) \\
& \longrightarrow & [\![P_2]\!] \mid (\nu\, s')([\![s'?(\tilde{x}); P_1]\!] \mid [\![\overline{s}!\langle \tilde{k}\rangle; \mathbf{0}]\!]) \\
& \longrightarrow^* & [\![P_2]\!] \mid [\![P_1]\!]\{\tilde{k}/\tilde{x}\}
\end{array}$$

Operational Correspondence for Recursion TODO

The inductive step is trivial since the rest of the reduction cases make use of the isomorphic encoding rules.

Part 2 TODO

An important result is that of the typability of the encodings.

Proposition 4.3 (Typable Encodings). *Let P be a pure session π or pure HO^+ process and $\Gamma \vdash P \triangleright \Delta$, then $\Gamma \vdash [\![P]\!] \triangleright \Delta$ for some environments Γ and Δ.*

Proof. 1. $s!\langle k\rangle; P$

$$\dfrac{\Gamma \vdash [\![P]\!] \triangleright \Delta \cdot s : T \qquad \dfrac{\dfrac{\Gamma \cdot X : T' \multimap \diamond \vdash X\langle k\rangle \triangleright k : T' \cdot X}{\Gamma \cdot X : T' \multimap \diamond \vdash X\langle k\rangle \triangleright k : T' \cdot X \cdot z : \mathsf{end}}}{\Gamma \vdash z?(X); X\langle k\rangle \triangleright z : ?(T' \multimap \diamond); \mathsf{end}}}{\Gamma \vdash s!\langle (z)z?(X); X\langle k\rangle\rangle; [\![P]\!] \triangleright s : \Delta \cdot s : !\langle ?(T' \multimap \diamond); \mathsf{end} \multimap \diamond\rangle; T}$$

2. $s?(x); P$ with $\Gamma' = \Gamma \cdot X : ?(T' \multimap \diamond); \mathsf{end}$

$$\dfrac{\dfrac{\dfrac{\Gamma' \vdash X\langle s'\rangle \triangleright s' : ?(T' \multimap \diamond); \mathsf{end} \cdot X \qquad \dfrac{\dfrac{\dfrac{\Gamma' \vdash \mathbf{0} \triangleright \emptyset}{\Gamma' \vdash \mathbf{0} \triangleright \overline{s'} : \mathsf{end}} \qquad \Gamma' \vdash [\![P]\!] \triangleright \Delta \cdot x : T' \cdot s : T}{\Gamma' \vdash \overline{s'}!\langle (x)[\![P]\!]\rangle; \mathbf{0} \triangleright \Delta \cdot \overline{s'} : !\langle T' \multimap \diamond\rangle; \mathsf{end} \cdot s : T}}{}}{\Gamma' \vdash X\langle s'\rangle \mid \overline{s'}!\langle (x)[\![P]\!]\rangle; \mathbf{0} \triangleright \Delta \cdot s' : ?(T' \multimap \diamond); \mathsf{end} \cdot \overline{s'} : !\langle T' \multimap \diamond\rangle; \mathsf{end} \cdot s : T \cdot X}}{\Gamma' \vdash (\nu\, s')(X\langle s'\rangle \mid \overline{s'}!\langle (x)[\![P]\!]\rangle; \mathbf{0}) \triangleright \Delta \cdot s : T \cdot X}}{\Gamma \vdash s?(X); (\nu\, s')(X\langle s'\rangle \mid \overline{s'}!\langle (x)[\![P]\!]\rangle; \mathbf{0}) \triangleright \Delta \cdot s : ?(?(T' \multimap \diamond); \mathsf{end} \multimap \diamond); T}$$

3. $s!\langle (Y)P_2\rangle; P_1$

$$\dfrac{\Gamma \vdash [\![P_1]\!] \triangleright \Delta_1 \cdot s : T \qquad \dfrac{\dfrac{\Gamma \cdot Y : T' \multimap \diamond \vdash [\![P_2]\!] \triangleright \Delta_2}{\Gamma\cdot Y : T' \multimap\diamond \vdash [\![P_2]\!]\triangleright \Delta_2 \cdot z : \mathsf{end}}}{\Gamma \vdash z?(Y); [\![P_2]\!] \triangleright \Delta_2 \backslash Y \cdot z : ?(T' \multimap \diamond);\mathsf{end}}}{\Gamma \vdash s!\langle (z) z?(Y); [\![P_2]\!]\rangle; [\![P_1]\!] \triangleright \Delta_1\cdot\Delta_2\backslash Y \cdot z : !\langle ?(T'\multimap\diamond);\mathsf{end}\multimap\diamond\rangle; T}$$

4. $X\langle (x)P\rangle$

$$\dfrac{\dfrac{\Gamma\cdot X : ?(T'\multimap\diamond);\mathbf{0}\multimap\diamond \vdash X\langle s\rangle \triangleright \Delta_1 \cdot s : ?(T'\multimap\diamond);\mathbf{0} \qquad \dfrac{\Gamma'\vdash [\![P]\!]\triangleright\Delta_2\cdot x : T' \qquad \dfrac{\Gamma'\vdash \mathbf{0}\triangleright\emptyset}{\Gamma'\vdash\mathbf{0}\triangleright\overline{s'}:\mathsf{end}}}{\Gamma'\vdash \overline{s}!\langle (x)P\rangle;\mathbf{0}\triangleright\Delta_2\cdot\overline{s} : !\langle T'\multimap\diamond\rangle;\mathsf{end}}}{\Gamma\cdot X : ?(T'\multimap\diamond);\mathbf{0}\multimap\diamond \vdash X\langle s\rangle \mid \overline{s}!\langle (x)P\rangle;\mathbf{0}\triangleright \Delta_1\cdot\Delta_2\cdot s: ?(T'\multimap\diamond);\mathbf{0}\cdot\overline{s}:!\langle T'\multimap\diamond\rangle;\mathsf{end}}}{\Gamma\cdot X : ?(T'\multimap\diamond);\mathbf{0}\multimap\diamond\vdash (\nu\, s)(X\langle s\rangle \mid \overline{s}!\langle (x)P\rangle;\mathbf{0})\triangleright\Delta_1\cdot\Delta_2}$$

5. $\mu r.P$

$$\dfrac{\Gamma\cdot X : ?(T'\to\diamond);\mathsf{end}\to\diamond \vdash [\![P]\!]\triangleright\Delta\cdot s : T}{\Gamma\vdash s?(X);[\![P]\!]\triangleright\Delta\cdot s : ?(?(T'\to\diamond);\mathsf{end}\to\diamond);T}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\Gamma\cdot X : T'\to\diamond\vdash\mathcal{A}[\![P]\!]^{\emptyset}\triangleright z:\mathsf{end}\cdot\tilde{y}:\tilde{T}}{\Gamma\vdash z?(X);\mathcal{A}[\![P]\!]^{\emptyset}\triangleright z : ?(T'\to\diamond);\mathsf{end}\cdot\tilde y:\tilde T} \qquad \dfrac{\Gamma\vdash\mathbf{0}\triangleright\emptyset}{\Gamma\vdash\mathbf{0}\triangleright\overline{s}:\mathsf{end}}}{\Gamma\vdash\overline{s}!\langle (z\tilde y) z?(X);\mathcal{A}[\![P]\!]^{s}\rangle;\mathbf{0}\triangleright\overline{s}:!\langle ?(T'\to\diamond);\mathsf{end}\to\diamond\rangle;\mathsf{end}}}{\Gamma\vdash s?(X);[\![P]\!]\mid\overline{s}!\langle(z\tilde y)z?(X);\mathcal{A}[\![P]\!]^s\rangle;\mathbf{0}\triangleright\Delta\cdot s : ?(?(T'\to\diamond);\mathsf{end}\to\diamond);T\cdot\overline{s}:!\langle ?(T'\to\diamond);\mathsf{end}\to\diamond\rangle;\mathsf{end}}}{\Gamma\vdash(\nu\, s)(s?(X);[\![P]\!]\mid\overline{s}!\langle(z\tilde y)z?(X);\mathcal{A}[\![P]\!]^s\rangle;\mathbf{0})\triangleright\Delta}$$

6. $[\![r]\!]$

4.4 Encode pure HO processes into pure session π.

First Approach

$$\begin{aligned}
[\![k?(X);P]\!] &\stackrel{\mathsf{def}}{=} k?(x);[\![P]\!] \\
[\![X\langle k\rangle]\!] &\stackrel{\mathsf{def}}{=} x!\langle k\rangle;\mathbf{0} \\
[\![k!\langle (x)Q\rangle;P]\!] &\stackrel{\mathsf{def}}{=} (\nu\, s)(k!\langle s\rangle;[\![P]\!] \mid \overline{s}?(x);[\![Q]\!])
\end{aligned}$$

Proposition 4.4. *Let P be a* pure HO *process with $\Gamma\vdash P\triangleright\Delta$ and with the typing derivation to contain only linear abstractions.* $[\![P]\!]$

- *is typable.*
- *enjoys operational correspondence.*

Proof. TODO

Nevertheless the above encoding is not typable and does not respect operational correspondence for processes that require shared abstractions.

Proposition 4.5. *Let P be a* pure HO *process with $\Gamma \vdash P \triangleright \Delta$ and with the typing derivation to contain shared abstractions.* $[\![P]\!]$

- *is not typable.*
- *does not enjoy operational correspondence.*

Proof. Let process $P = \overline{s}!\langle ()\mathbf{0}\rangle; \mathbf{0} \mid s?(X); (X\langle\rangle \mid X\langle\rangle)$. The typing of such process requires in its derivation to check process variable X against a shared type. We get

$$[\![P]\!] \stackrel{\mathsf{def}}{=} (\nu\, s')(\overline{s}!\langle s'\rangle; \mathbf{0} \mid \overline{s'}?(\mathbf{0});) \mid s?(x); (x!\langle\rangle; \mathbf{0} \mid x!\langle\rangle; \mathbf{0})$$

The derivation on the subprocess $x!\langle\rangle; \mathbf{0} \mid x!\langle\rangle; \mathbf{0}$ uses the [Par] rule which in turn checks for the disjointness of the two linear environemnts. But both environments contain variable x making the mapping untypable.

Furthermore in the untyped setting

$$[\![P]\!] \longrightarrow^{*} \mathbf{0} \mid x!\langle\rangle; \mathbf{0}$$

when

$$P \longrightarrow \mathbf{0}$$

providing evidence for no operational correspondence.

As a consequence of the last two proposition the provided encoding allows only for a limited set of processes (namely purely linear processes) to be encoded in a sound way.

Nevertheless we claim that there is a sound encoding from pure HO to pure session π, although its definition should be complicated. We give the basic intuition through an example.

Example 4.1. Let process

$$P = \overline{s}!\langle ()\mathbf{0}\rangle; \mathbf{0} \mid s?(X); (\mu r. X\langle\rangle \mid r \mid X\langle\rangle)$$

A sound mapping for this process should be

$$[\![P]\!] \stackrel{\mathsf{def}}{=} (\nu\, s_1, s_2)(\overline{s}!\langle s_1, s_2\rangle; \mathbf{0} \mid \mu r.(\overline{s_1}?(); r) \mid \overline{s_2}?(); \mathbf{0}) \mid s?(x_1, x_2); (\mu r.(x_1!\langle\rangle; r) \mid x_2!\langle\rangle; \mathbf{0})$$

To formalise the above intuition we should use a mapping with complex side conditions that tracks the entire structure of the process.

4.5 Encode pure HO processes into session π.

However we can easily provide a sound encoding for pure HO process into session π processes, by exploiting shared channels to represent shared abstractions.

$$\begin{aligned}
[\![k?(X); P]\!] &\stackrel{\mathsf{def}}{=} k?(x); [\![P]\!] \\
[\![X\langle k\rangle]\!] &\stackrel{\mathsf{def}}{=} (\nu\, s)(x!\langle s\rangle; \overline{s}!\langle k\rangle; \mathbf{0}) \\
[\![k!\langle (x)Q\rangle; P]\!] &\stackrel{\mathsf{def}}{=} (\nu\, a)(k!\langle a\rangle; [\![P]\!] \mid * a?(y); \overline{y}?(x); [\![Q]\!])
\end{aligned}$$

Operational Correspondence TODO

4.6 Negative Result

A good encoding of the session π calculus to the pure HO calculus should respect the represenation of race conditions over shared channels. This represantation can be captured by the $|$-preservation property.

In this section we prove that the pure HO calculus cannot represent session π processes that model race conditions.

First we prove an auxiliary result:

Lemma 4.1. *Let $P \mid Q$ a pure HO process with $\Gamma \vdash P \mid Q \triangleright \Delta$ and Δ well typed. If $P \mid P' \longrightarrow P' \mid Q'$ and $P \mid Q \longrightarrow P' \mid Q''$ then $Q' \equiv Q''$.*

Proof. We write $P \mid Q$ using the normal form (Proposition 1.1).

$$P \mid Q \equiv (\nu\ \tilde{s})(P_1 \mid \ldots \mid P_n) \mid (\nu\ \tilde{s'})(Q_1 \mid \ldots \mid Q_m)$$

We do a case analysis on the possible reductions:
Case:

$$(\nu\ \tilde{s})(P_1 \mid \ldots P_i \mid \ldots \mid P_j \mid \ldots \mid P_n) \mid Q \longrightarrow (\nu\ \tilde{s})(P_1 \mid \ldots P_i' \mid \ldots \mid P_j' \mid \ldots \mid P_n) \mid Q$$
$$(\nu\ \tilde{s})(P_1 \mid \ldots P_i \mid \ldots \mid P_j \mid \ldots \mid P_n) \mid Q \longrightarrow (\nu\ \tilde{s})(P_1 \mid \ldots P_i' \mid \ldots \mid P_j' \mid \ldots \mid P_n) \mid Q$$

The proof is trivial.
Case:

$$(\nu\ \tilde{s})(P_1 \mid \ldots P_i \mid \ldots \mid P_n) \mid (\nu\ \tilde{s'})(Q_1 \mid \ldots \mid Q_j \mid \ldots \mid Q_m) \longrightarrow (\nu\ \tilde{s})(P_1 \mid \ldots P_i' \mid \ldots \mid P_n) \mid (\nu\ \tilde{s'})(Q_1 \mid \ldots \mid Q_j' \mid \ldots \mid Q_m)$$
$$(\nu\ \tilde{s})(P_1 \mid \ldots P_i \mid \ldots \mid P_n) \mid (\nu\ \tilde{s'})(Q_1 \mid \ldots \mid Q_k \mid \ldots \mid Q_m) \longrightarrow (\nu\ \tilde{s})(P_1 \mid \ldots P_i' \mid \ldots \mid P_n) \mid (\nu\ \tilde{s'})(Q_1 \mid \ldots \mid Q_k' \mid \ldots \mid Q_m)$$

By normalisation (Lemma 1.1) we get that P_i and Q_j are session prefixed, so we can assume that they are prefixed on session s. By the well typeness condition of Δ we get that $s, \overline{s} \in \mathtt{dom}(\Delta)$, with $\Gamma \vdash P_i \triangleright \Delta \cdot \overline{s} : T_i$ If we assume that $k \neq j$ then $\Gamma \vdash Q_j \triangleright \Delta_j \cdot s : T_j$ and $\Gamma \vdash Q_k \triangleright \Delta_k \cdot s : T_k$, because the two processes should interact with endpoint $\overline{s}$ in process P_i. Furthermore typing rule [Par] cannot be applied to type $Q_j \mid Q_k$ because Δ_j and Δ_k are not disjoint. So it has to be that $k = j$ that results to:

$$(\nu\ \tilde{s'})(Q_1 \mid \ldots \mid Q_j' \mid \ldots \mid Q_m) \equiv (\nu\ \tilde{s'})(Q_1 \mid \ldots \mid Q_k' \mid \ldots \mid Q_m)$$

as required.

Theorem 4.1. *Mapping $[\![\cdot]\!]$: pure HO $\longrightarrow$ session π that enjoys operational correspondence and $|$-preservation does not exist.*

Proof. Let $[\![\cdot]\!]$: $\mathsf{pure\ HO} \longrightarrow \mathsf{session}\ \pi$ that respects operational correspondence and $|$-preservation and $\mathsf{pure\ HO}$ process: $P = a!\langle s\rangle; P_1 \mid a?(x); P_2 \mid a?(x); P_3$ with $P_1 \not\equiv P_2$ and $\Gamma \vdash P \triangleright \Delta$.

$|$-preservation implies

$$[\![P]\!] \stackrel{\mathsf{def}}{=} [\![a!\langle s\rangle; P_1]\!] \mid [\![a?(x); P_2]\!] \mid [\![a?(x); P_3]\!]$$

and operational correspondence implies

$$P \longrightarrow P_1 \mid P_2\{s/x\} \mid a?(x); P_3 \Rightarrow [\![P]\!] \longrightarrow [\![P_1 \mid P_2\{s/x\} \mid a?(x); P_3]\!] \tag{1}$$
$$P \longrightarrow P_1 \mid a?(x); P_1 \mid P_3\{s/x\} \Rightarrow [\![P]\!] \longrightarrow [\![P_1 \mid a?(x); P_2 \mid P_3\{s/x\}]\!] \tag{2}$$

By the $|$-preservation property we get that

$$[\![a!\langle s\rangle; P_1]\!] \mid [\![a?(x); P_2]\!] \mid [\![a?(x); P_3]\!] \longrightarrow [\![P_1]\!] \mid [\![P_2\{s/x\}]\!] \mid [\![a?(x); P_3]\!] \tag{3}$$
$$[\![a!\langle s\rangle; P_1]\!] \mid [\![a?(x); P_2]\!] \mid [\![a?(x); P_3]\!] \longrightarrow [\![P_1]\!] \mid [\![a?(x); P_2]\!] \mid [\![P_3\{s/x\}]\!] \tag{4}$$

By Lemma 4.1 we get that $[\![P_1]\!] \mid [\![P_2\{s/x\}]\!] \mid [\![a?(x); P_3]\!] \equiv [\![P_1]\!] \mid [\![a?(x); P_2]\!] \mid [\![P_3\{s/x\}]\!]$ which implies contradiction since $P_1\{s/x\} \not\equiv P_2\{s/x\}$.

5 Encode the λ-calculus

$$\begin{aligned}
[\![x]\!] &\stackrel{\mathsf{def}}{=} (z)(X\langle z\rangle) \\
[\![\lambda x.M]\!]^w &\stackrel{\mathsf{def}}{=} (z)z?(X); [\![M]\!]\langle w\rangle \\
[\![MN]\!] &\stackrel{\mathsf{def}}{=} (z)(\nu\ s)([\![M]\!]^z\langle s\rangle \mid \overline{s}!\langle [\![N]\!]\rangle; \mathbf{0}) \\
[\![M]\!]\langle z\rangle &\stackrel{\mathsf{def}}{=} (\nu\ s)(s?(X); X\langle z\rangle \mid \overline{s}!\langle [\![M]\!]\rangle; \mathbf{0}) \\
\lambda[M] &\stackrel{\mathsf{def}}{=} [\![M]\!]\langle - \rangle
\end{aligned}$$

$$
\begin{aligned}
& \lambda[(\lambda x.(\lambda x.x)x)y] \\
\stackrel{\texttt{def}}{=}\ & [\![(\lambda x.(\lambda x.x)x)y]\!]\langle w\rangle \\
\stackrel{\texttt{def}}{=}\ & (\nu\ s_1)(s_1?(X);X\langle w\rangle \mid \overline{s_1}!\langle [\![(\lambda x.(\lambda x.x)x)y]\!]\rangle;\mathbf{0}) \\
\stackrel{\texttt{def}}{=}\ & (\nu\ s_1)(s_1?(X);X\langle w\rangle \mid \overline{s_1}!\langle (z_1)(\nu\ s_2)([\![(\lambda x.(\lambda x.x)x]\!]^{z_1}\langle s_2\rangle \mid \overline{s_2}!\langle [\![y]\!]\rangle;\mathbf{0})\rangle;\mathbf{0}) \\
\stackrel{\texttt{def}}{=}\ & (\nu\ s_1)(s_1?(X);X\langle w\rangle \mid \overline{s_1}!\langle (z_1)(\nu\ s_2)((\nu\ s_3)(s_3?(X);X\langle s_2\rangle \mid \overline{s_3}!\langle [\![(\lambda x.(\lambda x.x)x]\!]^{z_1}\rangle;\mathbf{0}) \mid \overline{s_2}!\langle [\![y]\!]\rangle;\mathbf{0})\rangle;\mathbf{0}) \\
\stackrel{\texttt{def}}{=}\ & (\nu\ s_1)(s_1?(X);X\langle w\rangle \mid \overline{s_1}!\langle (z_1)(\nu\ s_2)((\nu\ s_3)(s_3?(X);X\langle s_2\rangle \mid \overline{s_3}!\langle (z_2)z_2?(X);[\![(\lambda x.x)x]\!]\langle z_1\rangle\rangle;\mathbf{0}) \mid \overline{s_2}!\langle [\![y]\!]\rangle;\mathbf{0})\rangle;\mathbf{0}) \\
\stackrel{\texttt{def}}{=}\ & (\nu\ s_1)(s_1?(X);X\langle w\rangle \mid \overline{s_1}!\langle (z_1)(\nu\ s_2)((\nu\ s_3)(s_3?(X);X\langle s_2\rangle \mid \\
& \overline{s_3}!\langle (z_2)z_2?(X);(\nu\ s_4)(s_4?(X);X\langle z_1\rangle \mid \overline{s_4}!\langle [\![(\lambda x.x)x]\!]\rangle;\mathbf{0})\rangle;\mathbf{0}) \mid \overline{s_2}!\langle [\![y]\!]\rangle;\mathbf{0})\rangle;\mathbf{0}) \\
\stackrel{\texttt{def}}{=}\ & (\nu\ s_1)(s_1?(X);X\langle w\rangle \mid \overline{s_1}!\langle (z_1)(\nu\ s_2)((\nu\ s_3)(s_3?(X);X\langle s_2\rangle \mid \\
& \overline{s_3}!\langle (z_2)z_2?(X);(\nu\ s_4)(s_4?(X);X\langle z_1\rangle \mid \overline{s_4}!\langle (z_3)(\nu\ s_5)([\![(\lambda x.x)]\!]^{z_3}\langle s_5\rangle \mid \overline{s_5}!\langle [\![x]\!]\rangle;\mathbf{0})\rangle;\mathbf{0})\rangle;\mathbf{0}) \mid \overline{s_2}!\langle [\![y]\!]\rangle;\mathbf{0})\rangle;\mathbf{0}) \\
\stackrel{\texttt{def}}{=}\ & (\nu\ s_1)(s_1?(X);X\langle w\rangle \mid \overline{s_1}!\langle (z_1)(\nu\ s_2)((\nu\ s_3)(s_3?(X);X\langle s_2\rangle \mid \\
& \overline{s_3}!\langle (z_2)z_2?(X);(\nu\ s_4)(s_4?(X);X\langle z_1\rangle \mid \overline{s_4}!\langle (z_3)(\nu\ s_5)((\nu\ s_6)(s_6?(X);X\langle s_5\rangle \mid \overline{s_6}!\langle [\![(\lambda x.x)]\!]^{z_3}\rangle;\mathbf{0}) \mid \\
& \overline{s_5}!\langle [\![x]\!]\rangle;\mathbf{0})\rangle;\mathbf{0})\rangle;\mathbf{0}) \mid \overline{s_2}!\langle [\![y]\!]\rangle;\mathbf{0})\rangle;\mathbf{0}) \\
\stackrel{\texttt{def}}{=}\ & (\nu\ s_1)(s_1?(X);X\langle w\rangle \mid \overline{s_1}!\langle (z_1)(\nu\ s_2)((\nu\ s_3)(s_3?(X);X\langle s_2\rangle \mid \\
& \overline{s_3}!\langle (z_2)z_2?(X);(\nu\ s_4)(s_4?(X);X\langle z_1\rangle \mid \overline{s_4}!\langle (z_3)(\nu\ s_5)((\nu\ s_6)(s_6?(X);X\langle s_5\rangle \mid \\
& \overline{s_6}!\langle (z_4)z_4?(X);[\![x]\!]\langle z_3\rangle\rangle;\mathbf{0}) \mid \\
& \overline{s_5}!\langle [\![x]\!]\rangle;\mathbf{0})\rangle;\mathbf{0})\rangle;\mathbf{0}) \mid \overline{s_2}!\langle [\![y]\!]\rangle;\mathbf{0})\rangle;\mathbf{0}) \\
\stackrel{\texttt{def}}{=}\ & (\nu\ s_1)(s_1?(X);X\langle w\rangle \mid \overline{s_1}!\langle (z_1)(\nu\ s_2)((\nu\ s_3)(s_3?(X);X\langle s_2\rangle \mid \\
& \overline{s_3}!\langle (z_2)z_2?(X);(\nu\ s_4)(s_4?(X);X\langle z_1\rangle \mid \overline{s_4}!\langle (z_3)(\nu\ s_5)((\nu\ s_6)(s_6?(X);X\langle s_5\rangle \mid \\
& \overline{s_6}!\langle (z_4)z_4?(X);(\nu\ s_7)(s_7?(X);X\langle z_3\rangle \mid \overline{s_7}!\langle [\![x]\!]\rangle;\mathbf{0})\rangle;\mathbf{0}) \mid \\
& \overline{s_5}!\langle [\![x]\!]\rangle;\mathbf{0})\rangle;\mathbf{0})\rangle;\mathbf{0}) \mid \overline{s_2}!\langle [\![y]\!]\rangle;\mathbf{0})\rangle;\mathbf{0}) \\
\stackrel{\texttt{def}}{=}\ & (\nu\ s_1)(s_1?(X);X\langle w\rangle \mid \overline{s_1}!\langle (z_1)(\nu\ s_2)((\nu\ s_3)(s_3?(X);X\langle s_2\rangle \mid \\
& \overline{s_3}!\langle (z_2)z_2?(X);(\nu\ s_4)(s_4?(X);X\langle z_1\rangle \mid \overline{s_4}!\langle (z_3)(\nu\ s_5)((\nu\ s_6)(s_6?(X);X\langle s_5\rangle \mid \\
& \overline{s_6}!\langle (z_4)z_4?(X);(\nu\ s_7)(s_7?(X);X\langle z_3\rangle \mid \overline{s_7}!\langle (z_5)(X\langle z_5\rangle)\rangle;\mathbf{0})\rangle;\mathbf{0}) \mid \\
& \overline{s_5}!\langle (z_6)(X\langle z_6\rangle)\rangle;\mathbf{0})\rangle;\mathbf{0})\rangle;\mathbf{0}) \mid \overline{s_2}!\langle (z_7)(Y\langle z_7\rangle)\rangle;\mathbf{0})\rangle;\mathbf{0})
\end{aligned}
$$

$$
\begin{aligned}
& (\nu\ s_1)(s_1?(X); X\langle w\rangle \mid \overline{s_1}!\langle (z_1)(\nu\ s_2)((\nu\ s_3)(s_3?(X); X\langle s_2\rangle \mid \\
& \overline{s_3}!\langle (z_2) z_2?(X); (\nu\ s_4)(s_4?(X); X\langle z_1\rangle \mid \overline{s_4}!\langle (z_3)(\nu\ s_5)((\nu\ s_6)(s_6?(X); X\langle s_5\rangle \mid \\
& \overline{s_6}!\langle (z_4) z_4?(X); (\nu\ s_7)(s_7?(X); X\langle z_3\rangle \mid \overline{s_7}!\langle (z_5)(X\langle z_5\rangle)\rangle; \mathbf{0})\rangle; \mathbf{0}) \mid \\
& \overline{s_5}!\langle (z_6)(X\langle z_6\rangle)\rangle; \mathbf{0})\rangle; \mathbf{0})\rangle; \mathbf{0}) \mid \overline{s_2}!\langle (z_7)(Y\langle z_7\rangle)\rangle; \mathbf{0})\rangle; \mathbf{0}) \\
\longrightarrow\ & (\nu\ s_2)((\nu\ s_3)(s_3?(X); X\langle s_2\rangle \mid \\
& \overline{s_3}!\langle (z_2) z_2?(X); (\nu\ s_4)(s_4?(X); X\langle w\rangle \mid \overline{s_4}!\langle (z_3)(\nu\ s_5)((\nu\ s_6)(s_6?(X); X\langle s_5\rangle \mid \\
& \overline{s_6}!\langle (z_4) z_4?(X); (\nu\ s_7)(s_7?(X); X\langle z_3\rangle \mid \overline{s_7}!\langle (z_5)(X\langle z_5\rangle)\rangle; \mathbf{0})\rangle; \mathbf{0}) \mid \\
& \overline{s_5}!\langle (z_6)(X\langle z_6\rangle)\rangle; \mathbf{0})\rangle; \mathbf{0})\rangle; \mathbf{0}) \mid \overline{s_2}!\langle (z_7)(Y\langle z_7\rangle)\rangle; \mathbf{0}) \\
\longrightarrow\ & (\nu\ s_4)(s_4?(X); X\langle w\rangle \mid \overline{s_4}!\langle (z_3)(\nu\ s_5)((\nu\ s_6)(s_6?(X); X\langle s_5\rangle \mid \\
& \overline{s_6}!\langle (z_4) z_4?(X); (\nu\ s_7)(s_7?(X); X\langle z_3\rangle \mid \overline{s_7}!\langle (z_5)(X\langle z_5\rangle)\rangle; \mathbf{0})\rangle; \mathbf{0}) \mid \\
& \overline{s_5}!\langle (z_6)(Y\langle z_6\rangle)\rangle; \mathbf{0})\rangle; \mathbf{0}) \\
\longrightarrow\ & (\nu\ s_5)((\nu\ s_6)(s_6?(X); X\langle s_5\rangle \mid \\
& \overline{s_6}!\langle (z_4) z_4?(X); (\nu\ s_7)(s_7?(X); X\langle w\rangle \mid \overline{s_7}!\langle (z_5)(X\langle z_5\rangle)\rangle; \mathbf{0})\rangle; \mathbf{0}) \mid \\
& \overline{s_5}!\langle (z_6)(Y\langle z_6\rangle)\rangle; \mathbf{0}) \\
\longrightarrow\ & (\nu\ s_5)(s_5?(X); (\nu\ s_7)(s_7?(X); X\langle w\rangle \mid \overline{s_7}!\langle (z_5)(X\langle z_5\rangle)\rangle; \mathbf{0}) \mid \\
& \overline{s_5}!\langle (z_6)(Y\langle z_6\rangle)\rangle; \mathbf{0}) \\
\longrightarrow\ & (\nu\ s_7)(s_7?(X); X\langle w\rangle \mid \overline{s_7}!\langle (z_5)(Y\langle z_5\rangle)\rangle; \mathbf{0}) \\
\longrightarrow\ & Y\langle w\rangle
\end{aligned}
$$

TODO
$((\lambda x.x)\lambda x.x)x$
$((\lambda x.x)\lambda z.z)\lambda y.y$

$$
\begin{aligned}
& \lambda[((\lambda x.x)\lambda z.z)\lambda y.y] \\
\stackrel{\text{def}}{=}\ & [\![((\lambda x.x)\lambda z.z)\lambda y.y]\!]\langle -\rangle \\
\stackrel{\text{def}}{=}\ & (\nu\ s_1)(s_1?(X); X\langle -\rangle \mid \overline{s_1}!\langle [\![((\lambda x.x)\lambda z.z)\lambda y.y]\!]\rangle; \mathbf{0}) \\
\stackrel{\text{def}}{=}\ & (\nu\ s_1)(s_1?(X); X\langle -\rangle \mid \overline{s_1}!\langle (z_1)(\nu\ s_2)([\![((\lambda x.x)\lambda z.z)]\!]^{z_1}\langle s_2\rangle \mid \overline{s_2}!\langle [\![\lambda y.y]\!]\rangle; \mathbf{0})\rangle; \mathbf{0})
\end{aligned}
$$