Core Higher-Order Session Processes: Tractable Equivalences and Relative Expressiveness

Abstract

This work proposes tractable bisimulations for the higher-order π-calculus with session primitives ($\mathsf{HO}\pi$) and offers a complete study of the expressivity of its most significant subcalculi. First we develop three typed bisimulations, which are shown to coincide with contextual equivalence. These characterisations demonstrate that observing as inputs only a specific finite set of higher-order values (which inhabit session types) suffices to reason about $\mathsf{HO}\pi$ processes. Next, we identify HO, a minimal, second-order subcalculus of $\mathsf{HO}\pi$ in which higher-order applications/abstractions, name-passing, and recursion are absent. We show that HO can encode $\mathsf{HO}\pi$ extended with higher-order applications and abstractions and that a first-order session π-calculus can encode $\mathsf{HO}\pi$. Both encodings are fully abstract. We also prove that the session π-calculus with passing of shared names cannot be encoded into $\mathsf{HO}\pi$ without shared names. We show that $\mathsf{HO}\pi$, HO, and π are equally expressive; the expressivity of HO enables effective reasoning about typed equivalences for higher-order processes.

1. Introduction

By combining features from the λ-calculus and the π-calculus, in *higher-order process calculi* exchanged values may contain processes. In this paper, we consider higher-order calculi with *session primitives*, thus enabling the specification of reciprocal exchanges (protocols) for higher-order mobile processes, which can be verified via type-checking using *session types* [10]. The study of higher-order concurrency has received significant attention, from untyped and typed perspectives (see, e.g., [11, 13, 17, 18, 20, 25, 28, 32, 34]). Although models of session-typed communication with features of higher-order concurrency exist [8, 19], their *tractable behavioural equivalences* and *relative expressiveness* remain little understood. Clarifying their status is not only useful for, e.g., justifying non-trivial mobile protocol optimisations, but also for transferring key reasoning techniques between (higher-order) session calculi. Our discovery is that *linearity* of session types plays a vital role to offer new equalities and fully abstract encodability, which to our best knowledge have not been proposed before.

The main higher-order language in our work, denoted $\mathsf{HO}\pi$, extends the higher-order π-calculus [25] with session primitives: it contains constructs for synchronisation on shared names, recursion, name abstractions (i.e., functions from name identifiers to processes, denoted $\lambda x.P$) and applications (denoted $(\lambda x.P)a$); and session communication (value passing and labelled choice using linear names). We study two significant subcalculi of $\mathsf{HO}\pi$, which distil higher- and first-order mobility: the HO-calculus, which is $\mathsf{HO}\pi$ without recursion and name passing, and the session π-calculus (here denoted π), which is $\mathsf{HO}\pi$ without abstractions and applications. While π is, in essence, the calculus in [10], this paper shows that HO is a new core calculus for higher-order session concurrency.

In the first part of the paper, we address tractable behavioural equivalences for $\mathsf{HO}\pi$. A well-studied behavioural equivalence in the higher-order setting is *context bisimilarity* [27], a labelled characterisation of reduction-closed, barbed congruence, which offers an appropriate discriminative power at the price of heavy universal quantifications in output clauses. Obtaining alternative characterisations is thus a recurring issue in the study of higher-order calculi. Our approach shows that protocol specifications given by session types are essential to limit the behaviour of higher-order session processes. Exploiting elementary processes inhabiting session types, this limitation is formally enforced by a refined (typed) labelled transition system (LTS) that narrows down the spectrum of allowed process behaviours, thus enabling tractable reasoning techniques. Two tractable characterisations of bisimilarity are then introduced. Remarkably, using session types we prove that these bisimilarities coincide with context bisimilarity, without using operators for name-matching.

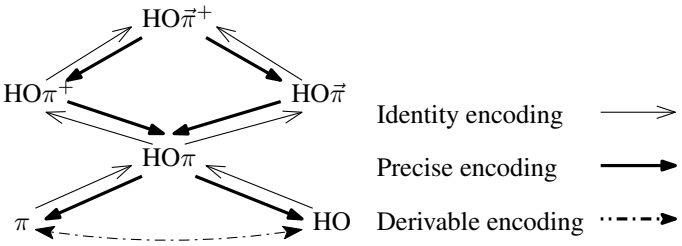

Figure 1: Encodability in Higher-Order Session Calculi. Precise encodings are defined in Def. 6.5.

We then move on to assess the expressivity of $\mathsf{HO}\pi$, HO, and π as delineated by typing. We establish strong correspondences between these calculi via type-preserving, fully abstract encodings up to behavioural equalities. While encoding $\mathsf{HO}\pi$ into the π-calculus preserving session types (extending known results for untyped processes) is significant, our main contribution is an encoding of $\mathsf{HO}\pi$ into HO, where name-passing is absent.

We illustrate the essence of encoding name passing into HO: to encode name output, we "pack" the name to be passed around into a suitable abstraction; upon reception, the receiver must "unpack" this object following a precise protocol. More precisely, our encoding of name passing in HO is given as:

$$\begin{array}{lcl} [\![a!\langle b\rangle.P]\!] & = & a!\langle \lambda z.\, z?(x).(x\,b)\rangle.[\![P]\!] \\ [\![a?(x).Q]\!] & = & a?(y).(\nu\, s)(y\, s \mid \overline{s}!\langle \lambda x.[\![Q]\!]\rangle.\mathbf{0}) \end{array}$$

where a, b are names; s and $\overline{s}$ are linear names (called *session endpoints*); $a!\langle V\rangle.P$ and $a?(x).P$ denote an output and input at a; and $(\nu\, s)(P)$ is hiding. A (deterministic) reduction between endpoints s and $\overline{s}$ guarantees name b is properly unpacked. Encoding a recursive process $\mu X.P$ is also challenging, for the linearity of endpoints in P must be preserved. We encode recursion with non-tail recursive session types; for this we apply recent advances on the theory of session duality [3, 4].

We further extend our encodability results to i) $\mathsf{HO}\pi$ with *higher-order* abstractions (denoted $\mathsf{HO}\pi^+$) and to ii) $\mathsf{HO}\pi$ with polyadic name passing and abstraction ($\mathsf{HO}\vec{\pi}$); and to their supercalculus ($\mathsf{HO}\vec{\pi}^+$) (equivalent to the calculus in [19]). A further result shows that shared names strictly add expressive power to session calculi. Fig. 1 summarises these results.

Outline / Contributions. This paper is structured as follows:

- § **??** overviews key ideas of our tractable bisimulations.

$$u,w ::= n \mid x,y,z \quad n ::= a,b \mid s,\overline{s} \quad V,W ::= u \mid \lambda x.P$$

$$P,Q ::= u!\langle V\rangle.P \mid u?(x).P \mid u \triangleleft l.P \mid u \triangleright \{l_i : P_i\}_{i\in I} \mid X \mid \mu X.P \mid V\,u \mid P \mid Q \mid (\nu n)P \mid \mathbf{0}$$

Figure 2: Syntax of HOπ (HO lacks the constructs in grey).

- § 3 presents the higher-order session calculus HOπ and its subcalculi HO and π. Then, § **??** gives the session type system and states type soundness for HOπ and its variants.
- § **??** develops *higher-order* and *characteristic* bisimilarities, our two tractable characterisations of contextual equivalence which alleviate the issues of context bisimilarity [27]. These relations are shown to coincide in HOπ (Thm. **??**).
- § 6 defines *precise (typed) encodings* by extending encodability criteria studied for untyped processes (e.g. [9, 17]).
- § 7 gives encodings of HOπ into HO and of HOπ into π. These encodings are shown to be *precise* (Thms. 7.4 and 7.6). Mutual encodings between π and HO are derivable; all these calculi are thus equally expressive. Exploiting determinacy and typed equivalences, we also prove the non-encodability of shared names into linear names (Thm. 7.7).
- § 8 studies extensions of HOπ. We show that both HOπ^+ (the extension with higher-order applications) and HO$\vec{\pi}$ (the extension with polyadicity) are encodable in HOπ (Thms. 8.1 and 8.2). This connects our work to the existing higher-order session calculus in [19] (here denoted HO$\vec{\pi}^+$).
- § 9 concludes with related works. The appendix summarises the typing system.

The paper is self-contained. ***Additional related work, more examples, omitted definitions, and proofs are in [1].***

2. Overview

3. Higher-Order Session π-Calculi

We introduce the *Higher-Order Session π-Calculus* (HOπ). HOπ includes both name- and abstraction-passing as well as recursion; it is a subcalculus of the language studied in [19]. Following the literature [11], for simplicity of the presentation we concentrate on the second-order call-by-value HOπ. (In § 8 we consider extensions of HOπ with higher-order abstractions and polyadicity in name-passing/abstractions.)

3.1 Syntax of HOπ

Values The syntax of HOπ is defined in Fig. 2 We use $a,b,c,\ldots$ (resp. $s,\overline{s},\ldots$) to range over shared (resp. session) names. We use $m,n,t,\ldots$ for session or shared names. We define the dual operation over names n as $\overline{n}$ with $\overline{\overline{s}} = s$ and $\overline{a} = a$. Intuitively, names s and $\overline{s}$ are dual (two) *endpoints* while shared names represent shared (non-deterministic) points. Variables are denoted with $x,y,z,\ldots$, and recursive variables are denoted with $X,Y\ldots$. An abstraction $\lambda x.P$ is a process P with name parameter x. Values V,W include identifiers $u,v,\ldots$ and abstractions $\lambda x.P$ (first- and higher-order values, resp.).

Terms include the π-calculus prefixes for sending and receiving values V. Process $u!\langle V\rangle.P$ denotes the output of value V over name u, with continuation P; process $u?(x).P$ denotes the input prefix on name u of a value that will substitute variable x in continuation P. Recursion is expressed by $\mu X.P$, which binds the recursive variable X in process P. Process $V\,u$ is the application which substitutes name u on the abstraction V. Typing ensures that V is not a name.

$$\begin{array}{rcll}
(\lambda x.P)\,u & \longrightarrow & P\{u/x\} & \text{[App]}\\
n!\langle V\rangle.P \mid \overline{n}?(x).Q & \longrightarrow & P \mid Q\{V/x\} & \text{[Pass]}\\
n \triangleleft l_j.Q \mid \overline{n} \triangleright \{l_i : P_i\}_{i\in I} & \longrightarrow & Q \mid P_j \ \ (j \in I) & \text{[Sel]}\\
P \longrightarrow P' & \Rightarrow & (\nu n)P \longrightarrow (\nu n)P' & \text{[Res]}\\
P \longrightarrow P' & \Rightarrow & P \mid Q \longrightarrow P' \mid Q & \text{[Par]}\\
P \equiv Q \longrightarrow Q' \equiv P' & \Rightarrow & P \longrightarrow P' & \text{[Cong]}
\end{array}$$

$$P \mid \mathbf{0} \equiv P \quad P_1 \mid P_2 \equiv P_2 \mid P_1 \quad P_1 \mid (P_2 \mid P_3) \equiv (P_1 \mid P_2) \mid P_3$$
$$(\nu n)\mathbf{0} \equiv \mathbf{0} \quad P \mid (\nu n)Q \equiv (\nu n)(P \mid Q)\ (n \notin \mathtt{fn}(P)) \quad \mu X.P \equiv P\{\mu X.P/X\}$$
$$P \equiv Q \text{ if } P \equiv_\alpha Q$$

Figure 3: Operational Semantics of HOπ.

Prefix $u \triangleleft l.P$ selects label l on name u and then behaves as P. Process $u \triangleright \{l_i : P_i\}_{i\in I}$ offers a choice on labels l_i with continuation P_i, given that $i \in I$. Constructs for inaction $\mathbf{0}$, parallel composition $P_1 \mid P_2$, and name restriction $(\nu n)P$ are standard. Session name restriction $(\nu s)P$ simultaneously binds endpoints s and $\overline{s}$ in P. We use $\mathtt{fv}(P)$ and $\mathtt{fn}(P)$ to denote a set of free variables and names; and assume V in $u!\langle V\rangle.P$ does not include free recursive variables X. If $\mathtt{fv}(P) = \emptyset$, we call P *closed*.

3.2 Subcalculi of HOπ

We define some subcalculi of HOπ.

- The first subcalculus is the *core higher-order session calculus* (denoted HO), which lacks recursion and name passing; its formal syntax is obtained from Fig. 2 by excluding constructs in grey.
- The second subcalculus is the *session π-calculus* (denoted π), which lacks the higher-order constructs (i.e., abstraction passing and application), but includes recursion. Let $\mathsf{C} \in \{\text{HO}\pi, \text{HO}, \pi\}$. We write $\mathsf{C}^{-\mathsf{sh}}$ for C without shared names (we delete a,b from n). We shall demonstrate that HOπ, HO, and π have the same expressivity.
- The third sub-calculus, denoted π^{λ}, represents cloud Haskell:

$$\begin{array}{lcl}
V,W & ::= & u \mid \lambda x.P\\
P,Q & ::= & u!\langle m\rangle.P \mid u?(x).P \mid u \triangleleft l.P \mid u \triangleright \{l_i : P_i\}_{i\in I}\\
 & \mid & V\,V \mid P \mid Q \mid (\nu n)P \mid \mathbf{0}
\end{array}$$

3.3 Operational Semantics

Fig. 3 defines the operational semantics of HOπ. [App] is a name application; [Pass] defines a shared interaction at n (with $\overline{n} = n$) or a session interaction; [Sel] is the standard rule for labelled choice/selection: given an index set I, a process selects label l_j on name n over a set of labels $\{l_i\}_{i\in I}$ offered by a branching on the dual endpoint $\overline{n}$; and other rules are standard. Rules for *structural congruence* are defined in Fig. 3 (bottom). We assume the expected extension of $\equiv$ to values V. We write $\longrightarrow^*$ for a multi-step reduction.

4. Types and Typing

5. Higher-Order Session Bisimulation

6. The Notion of Typed Encoding

Here we define the formal notion of *encoding* by extending to a typed setting existing criteria for untyped processes (as in, e.g., [7, 9, 17, 21–23, 33]). We first define a typed calculus parameterised by a syntax, operational semantics, and typing.

Definition 6.1 (Typed Calculus). A *typed calculus* $\mathcal{L}$ is a tuple $\langle \mathsf{C}, \mathcal{T}, \overset{\tau}{\longmapsto}, \approx, \vdash\rangle$ where C and $\mathcal{T}$ are sets of processes and types,

respectively; and $\overset{\tau}{\longmapsto}$, $\approx$, and $\vdash$ denote a transition system, a typed equivalence, and a typing system for C, respectively.

As we explain later, we write $\overset{\tau}{\longmapsto}$ to denote an operational semantics defined in terms of τ-transitions (to characterise reductions). Based on this definition, later on we define concrete instances of (higher-order) typed calculi. Our notion of encoding considers mappings on processes, types, and transition labels:

Definition 6.2 (Typed Encoding). Let $\mathcal{L}_1 = \langle \mathsf{C}_1, \mathcal{T}_1, \overset{\tau}{\longmapsto}_1, \approx_1, \vdash_1 \rangle$ and $\mathcal{L}_2 = \langle \mathsf{C}_2, \mathcal{T}_2, \overset{\tau}{\longmapsto}_2, \approx_2, \vdash_2 \rangle$ be typed calculi. Given mappings $[\![\cdot]\!] : \mathsf{C}_1 \to \mathsf{C}_2$ and $(\!|\cdot|\!) : \mathcal{T}_1 \to \mathcal{T}_2$, we write $\langle [\![\cdot]\!], (\!|\cdot|\!) \rangle : \mathcal{L}_1 \to \mathcal{L}_2$ to denote the *typed encoding* of $\mathcal{L}_1$ into $\mathcal{L}_2$.

We will often assume that $(\!|\cdot|\!)$ extends to typing environments as expected. This way, e.g., $(\!|\Delta \cdot u : S|\!) = (\!|\Delta|\!) \cdot u : (\!|S|\!)$.

We introduce two classes of typed encodings, which serve different purposes. First, for stating stronger positive encodability results, we define the notion of *precise* encodings. Then, for proving the strong non-encodability result, precise encodings are relaxed into the weaker *minimal* encodings.

We first state the syntactic criteria. Let σ denote a substitution of names for names (a renaming, in the usual sense). Given environments Δ and Γ, we write $\sigma(\Delta)$ and $\sigma(\Gamma)$ to denote the effect of applying σ on the domains of Δ and Γ (clearly, $\sigma(\Gamma)$ concerns only shared names in Γ: process and recursive variables in Γ are not affected by σ).

Definition 6.3 (Syntax Preserving Encoding). The typed encoding $\langle [\![\cdot]\!], (\!|\cdot|\!) \rangle : \mathcal{L}_1 \to \mathcal{L}_2$ is *syntax preserving* if it is:

1. *Homomorphic wrt parallel*, if $(\!|\Gamma|\!); \emptyset; (\!|\Delta_1 \cdot \Delta_2|\!) \vdash_2 [\![P_1 \mid P_2]\!] \triangleright \diamond$ then $(\!|\Gamma|\!); \emptyset; (\!|\Delta_1|\!) \cdot (\!|\Delta_2|\!) \vdash_2 [\![P_1]\!] \mid [\![P_2]\!] \triangleright \diamond$.
2. *Compositional wrt restriction*, if $(\!|\Gamma|\!); \emptyset; (\!|\Delta|\!) \vdash_2 [\![(\nu n)P]\!] \triangleright \diamond$ then $(\!|\Gamma|\!); \emptyset; (\!|\Delta|\!) \vdash_2 (\nu n)[\![P]\!] \triangleright \diamond$.
3. *Name invariant*, if $(\!|\sigma(\Gamma)|\!); \emptyset; (\!|\sigma(\Delta)|\!) \vdash_2 [\![\sigma(P)]\!] \triangleright \diamond$ then $\sigma((\!|\Gamma|\!)); \emptyset; \sigma((\!|\Delta|\!)) \vdash_2 \sigma([\![P]\!]) \triangleright \diamond$, for any injective renaming of names σ.

Homomorphism wrt parallel (used in, e.g., [22, 23]) expresses that encodings should preserve the distributed topology of source processes. This criteria is appropriate for both encodability and non encodability results; in our setting, it is induced by rules for typed composition. Compositionality wrt restriction is also supported by typing and turns out to be useful in our encodability results (§ 7). The name invariance criteria follows [9, 17]. Next we define semantic criteria:

Definition 6.4 (Semantic Preserving Encoding). Consider typed calculi $\mathcal{L}_1 = \langle \mathsf{C}_1, \mathcal{T}_1, \overset{\tau}{\longmapsto}_1, \approx_1, \vdash_1 \rangle$ and $\mathcal{L}_2 = \langle \mathsf{C}_2, \mathcal{T}_2, \overset{\tau}{\longmapsto}_2, \approx_2, \vdash_2 \rangle$. We say that $\langle [\![\cdot]\!], (\!|\cdot|\!) \rangle : \mathcal{L}_1 \to \mathcal{L}_2$ is a *semantic preserving encoding* if it satisfies the properties below.

1. *Type Preservation*: if $\Gamma; \emptyset; \Delta \vdash_1 P \triangleright \diamond$ then $(\!|\Gamma|\!); \emptyset; (\!|\Delta|\!) \vdash_2 [\![P]\!] \triangleright \diamond$, for any P in C_1.
2. *Barb preserving*: if $\Gamma; \Delta \vdash_1 P \downarrow_n$ then $(\!|\Gamma|\!); (\!|\Delta|\!) \vdash_2 [\![P]\!] \Downarrow_n$
3. *Operational Correspondence*: If $\Gamma; \emptyset; \Delta \vdash_1 P \triangleright \diamond$ then
   (a) Completeness: If $\Gamma; \Delta \vdash_1 P \overset{\tau}{\longmapsto}_1 \Delta' \vdash_1 P'$ then $\exists Q, \Delta''$ s.t. (i) $(\!|\Gamma|\!); (\!|\Delta|\!) \vdash_2 [\![P]\!] \Longrightarrow_2 (\!|\Delta''|\!) \vdash_2 Q$ and (ii) $(\!|\Gamma|\!); (\!|\Delta''|\!) \vdash_2 Q \approx_2 (\!|\Delta'|\!) \vdash_2 [\![P']\!]$.
   (b) Soundness: If $(\!|\Gamma|\!); (\!|\Delta|\!) \vdash_2 [\![P]\!] \Longrightarrow_2 (\!|\Delta'|\!) \vdash_2 Q$ then $\exists P', \Delta''$ s.t. (i) $\Gamma; \Delta \vdash_1 P \overset{\tau}{\longmapsto}_1 \Delta'' \vdash_1 P'$ and (ii) $(\!|\Gamma|\!); (\!|\Delta''|\!) \vdash_2 [\![P']\!] \approx_2 (\!|\Delta'|\!) \vdash_2 Q$.
4. *Full Abstraction:* $\Gamma; \Delta_1 \vdash_1 P \approx_1 \Delta_2 \vdash_1 Q$ if and only if $(\!|\Gamma|\!); (\!|\Delta_1|\!) \vdash_2 [\![P]\!] \approx_2 (\!|\Delta_2|\!) \vdash_2 [\![Q]\!]$.

Type preservation is a distinguishing criterion in our notion of encoding: it enables us to focus on encodings which retain the communication structures denoted by session types. Operational correspondence, standardly divided into completeness and soundness, is based on [9, 17]; it relies on τ-labeled transitions (reductions). Completeness ensures that an action of the source process is mimicked by an action of its associated encoding; soundness is its converse. Above, operational correspondence is stated in generic terms. It is worth stressing that the operational correspondence statements for our encodings are tailored to the specifics of each encoding, and so they are actually stronger than the criteria given above (see [1] for details). Finally, following [23, 25, 36], we consider full abstraction as an encodability criterion: this results into stronger encodability results. From the criteria in Def. 6.3 and 6.4 we have the following derived criterion:

We may now define *precise* and *minimal* typed criteria:

Definition 6.5 (Typed Encodings: Precise and Minimal). We say that the typed encoding $\langle [\![\cdot]\!], (\!|\cdot|\!) \rangle : \mathcal{L}_1 \to \mathcal{L}_2$ is

- *precise*, if it is both syntax and semantic preserving (cf. Def. 6.3 and 6.4).
- *minimal*, if it is syntax preserving (cf. Def. 6.3), barb preserving (cf. Def. 6.4-2), and operationally complete (cf. Def. 6.4-3(a)).

Our encodability results rely on precise encodings; our non encodability result uses minimal encodings. Further we have:

Proposition 6.6 (Composability of Precise Encodings). Let $\langle [\![\cdot]\!]^1, (\!|\cdot|\!)^1 \rangle : \mathcal{L}_1 \to \mathcal{L}_2$ and $\langle [\![\cdot]\!]^2, (\!|\cdot|\!)^2 \rangle : \mathcal{L}_2 \to \mathcal{L}_3$ be two precise typed encodings. Then their composition, denoted $\langle [\![\cdot]\!]^2 \circ [\![\cdot]\!]^1, (\!|\cdot|\!)^2 \circ (\!|\cdot|\!)^1 \rangle : \mathcal{L}_1 \to \mathcal{L}_3$ is precise.

Concrete Typed Calculi: In § 7 we consider the following concrete instances of typed calculi (cf. Def. 6.1):
$\mathcal{L}_{\mathsf{HO}\pi} = \langle \mathsf{HO}\pi, \mathcal{T}_1, \overset{\tau}{\longmapsto}, \approx^H, \vdash \rangle$, $\mathcal{L}_{\mathsf{HO}} = \langle \mathsf{HO}, \mathcal{T}_2, \overset{\tau}{\longmapsto}, \approx^H, \vdash \rangle$, and $\mathcal{L}_\pi = \langle \pi, \mathcal{T}_3, \overset{\tau}{\longmapsto}, \approx^C, \vdash \rangle$ where: $\mathcal{T}_1$, $\mathcal{T}_2$, and $\mathcal{T}_3$ are sets of types of $\mathsf{HO}\pi$, HO, and π, resp.; LTSs are as in Def. **??**, the typing $\vdash$ is defined in Fig. 9; $\approx^H$ is as in Def. **??**; $\approx^C$ is as in Def. **??**.

7. Expressiveness Results

This section presents two encodability results: (1) The higher-order name-passing communication with recursion ($\mathsf{HO}\pi$) into the higher-order communication without name-passing nor recursion (HO) (§ 7.1); and (2) $\mathsf{HO}\pi$ into the first-order name-passing communication with recursion (π) (§ 7.2).

7.1 From $\mathsf{HO}\pi$ to HO

We show that HO is expressive enough to represent the full $\mathsf{HO}\pi$-calculus. The main challenge is to encode (1) name passing and (2) recursion, for which we only use abstraction passing. For (1), we pass an abstraction which enables to use the name upon application. For (2), we copy a process upon reception; the case of linear abstraction passing is delicate because linear abstractions cannot be copied. To handle linearity, we define a mapping from processes with free names to processes without free names (but with free variables instead). To this end, we require two auxiliary definitions.

Definition 7.1. Let $\|\cdot\| : 2^{\mathcal{N}} \longrightarrow \mathcal{V}^\omega$ be a map of sequences of lexicographically ordered names to sequences of variables, defined inductively as: $\|\epsilon\| = \epsilon$ and $\|n \cdot \tilde{m}\| = x_n \cdot \|\tilde{m}\|$.

Definition 7.2 (Auxiliary Mapping). *Let σ be a set of session names. Fig. 4 defines an auxiliary mapping $[\![\cdot]\!]_\sigma : \mathsf{HO} \to \mathsf{HO}$.*

$$\llbracket n!\langle\lambda x.\,Q\rangle.P\rrbracket_\sigma \stackrel{\text{def}}{=} u!\langle\lambda x.\llbracket Q\rrbracket_\sigma\rangle.\llbracket P\rrbracket_\sigma$$

$$\llbracket(\nu n)P\rrbracket_\sigma \stackrel{\text{def}}{=} (\nu n)\llbracket P\rrbracket_{\sigma\cdot n} \qquad \llbracket P \mid Q\rrbracket_\sigma \stackrel{\text{def}}{=} \llbracket P\rrbracket_\sigma \mid \llbracket Q\rrbracket_\sigma$$

$$\llbracket xn\rrbracket_\sigma \stackrel{\text{def}}{=} xu \qquad \llbracket(\lambda x.Q)n\rrbracket_\sigma \stackrel{\text{def}}{=} (\lambda x.\llbracket Q\rrbracket_\sigma)u$$

$$\llbracket\mathbf{0}\rrbracket_\sigma \stackrel{\text{def}}{=} \mathbf{0} \qquad \llbracket n?(x).P\rrbracket_\sigma \stackrel{\text{def}}{=} u?(x).\llbracket P\rrbracket_\sigma$$

$$\llbracket n \triangleleft l.P\rrbracket_\sigma \stackrel{\text{def}}{=} u \triangleleft l.\llbracket P\rrbracket_\sigma \qquad \llbracket n \triangleright \{l_i : P_i\}_{i\in I}\rrbracket_\sigma \stackrel{\text{def}}{=} u \triangleright \{l_i : \llbracket P_i\rrbracket_\sigma\}_{i\in I}$$

In all cases: $u = n$ if $n \in \sigma$; otherwise $u = x_n$.

Figure 4: Auxiliary mapping used to encode HOπ into HO.

Types :

$$\lfloor S\rfloor^1 \stackrel{\text{def}}{=} (?(\langle\!\langle S\rangle\!\rangle^1 \multimap \diamond); \mathsf{end}) \multimap \diamond$$

$$\lfloor\langle S\rangle\rfloor^1 \stackrel{\text{def}}{=} (?(\langle\langle\!\langle S\rangle\!\rangle^1\rangle \to \diamond); \mathsf{end}) \multimap \diamond$$

$$\lfloor\langle L\rangle\rfloor^1 \stackrel{\text{def}}{=} (?(\langle\langle\!\langle L\rangle\!\rangle^1\rangle \to \diamond); \mathsf{end}) \multimap \diamond$$

$$\lfloor C \multimap \diamond\rfloor^1 \stackrel{\text{def}}{=} \langle\!\langle C\rangle\!\rangle^1 \multimap \diamond \qquad \lfloor C \to \diamond\rfloor^1 \stackrel{\text{def}}{=} \langle\!\langle C\rangle\!\rangle^1 \to \diamond$$

$$\langle\!\langle\langle S\rangle\rangle\!\rangle^1 \stackrel{\text{def}}{=} \langle\langle\!\langle S\rangle\!\rangle^1\rangle \qquad \langle\!\langle\langle L\rangle\rangle\!\rangle^1 \stackrel{\text{def}}{=} \langle\langle\!\langle L\rangle\!\rangle^1\rangle$$

$$\langle\!\langle !\langle U\rangle; S\rangle\!\rangle^1 \stackrel{\text{def}}{=} !\langle\lfloor U\rfloor^1\rangle; \langle\!\langle S\rangle\!\rangle^1 \qquad \langle\!\langle ?(U); S\rangle\!\rangle^1 \stackrel{\text{def}}{=} ?(\lfloor U\rfloor^1); \langle\!\langle S\rangle\!\rangle^1$$

$$\langle\!\langle \oplus\{l_i : S_i\}_{i\in I}\rangle\!\rangle^1 \stackrel{\text{def}}{=} \oplus\{l_i : \langle\!\langle S_i\rangle\!\rangle^1\}_{i\in I}$$

$$\langle\!\langle \&\{l_i : S_i\}_{i\in I}\rangle\!\rangle^1 \stackrel{\text{def}}{=} \&\{l_i : \langle\!\langle S_i\rangle\!\rangle^1\}_{i\in I}$$

$$\langle\!\langle \mathsf{t}\rangle\!\rangle^1 \stackrel{\text{def}}{=} \mathsf{t} \qquad \langle\!\langle \mu\mathsf{t}.S\rangle\!\rangle^1 \stackrel{\text{def}}{=} \mu\mathsf{t}.\langle\!\langle S\rangle\!\rangle^1 \qquad \langle\!\langle \mathsf{end}\rangle\!\rangle^1 \stackrel{\text{def}}{=} \mathsf{end}$$

Terms :

$$\llbracket u!\langle w\rangle.P\rrbracket_f^1 \stackrel{\text{def}}{=} u!\langle\lambda z.\, z?(x).(x\,w)\rangle.\llbracket P\rrbracket_f^1$$

$$\llbracket u?(x:C).Q\rrbracket_f^1 \stackrel{\text{def}}{=} u?(y).(\nu\, s)(y\, s \mid \overline{s}!\langle\lambda x.\llbracket Q\rrbracket_f^1\rangle.\mathbf{0})$$

$$\llbracket u!\langle\lambda x.\,Q\rangle.P\rrbracket_f^1 \stackrel{\text{def}}{=} u!\langle\lambda x.\llbracket Q\rrbracket_f^1\rangle.\llbracket P\rrbracket_f^1$$

$$\llbracket u?(x:L).P\rrbracket_f^1 \stackrel{\text{def}}{=} u?(x).\llbracket P\rrbracket_f^1$$

$$\llbracket s \triangleleft l.P\rrbracket_f^1 \stackrel{\text{def}}{=} s \triangleleft l.\llbracket P\rrbracket_f^1$$

$$\llbracket s \triangleright \{l_i : P_i\}_{i\in I}\rrbracket_f^1 \stackrel{\text{def}}{=} s \triangleright \{l_i : \llbracket P_i\rrbracket_f^1\}_{i\in I}$$

$$\llbracket\mathbf{0}\rrbracket_f^1 \stackrel{\text{def}}{=} \mathbf{0} \qquad \llbracket(\nu n)P\rrbracket_f^1 \stackrel{\text{def}}{=} (\nu n)\llbracket P\rrbracket_f^1$$

$$\llbracket xu\rrbracket_f^1 \stackrel{\text{def}}{=} xu \qquad \llbracket(\lambda x.Q)u\rrbracket_f^1 \stackrel{\text{def}}{=} (\lambda x.\llbracket Q\rrbracket_f^1)u$$

$$\llbracket P \mid Q\rrbracket_f^1 \stackrel{\text{def}}{=} \llbracket P\rrbracket_f^1 \mid \llbracket Q\rrbracket_f^1$$

$$\llbracket\mu X.P\rrbracket_f^1 \stackrel{\text{def}}{=}$$
$$(\nu\, s)(\overline{s}!\langle\lambda(\|\tilde{n}\|, y).\, y?(z_X).\llbracket\llbracket P\rrbracket^1_{f,\{X\to\tilde{n}\}}\rrbracket_\emptyset\rangle.\mathbf{0}$$
$$\mid s?(z_X).\llbracket P\rrbracket^1_{f,\{X\to\tilde{n}\}}) \quad (\tilde{n} = \mathtt{fn}(P))$$

$$\llbracket X\rrbracket_f^1 \stackrel{\text{def}}{=} (\nu\, s)(z_X(\tilde{n}, s)$$
$$\mid \overline{s}!\langle\lambda(\|\tilde{n}\|, y).\, z_X(\|\tilde{n}\|, y)\rangle.\mathbf{0}) \quad (\tilde{n} = f(X))$$

Above $\mathtt{fn}(P)$ denotes a lexicographically ordered sequence of free names in P. The input bound variable x is annotated by a type to distinguish first- and higher-order cases.

Figure 5: Encoding of HOπ into HO.

This way, given an HO process P (with $\mathtt{fn}(P) = m_1, \cdots, m_k$), we are interested in its associated abstraction, defined as $\lambda x_1 \cdots x_n.\llbracket P\rrbracket_\emptyset$, where $\|m_j\| = x_j$, for all $j \in \{1, \ldots, k\}$. In the following we make this intuition precise.

Definition 7.3 (Typed Encoding of HOπ into HO). *Let f be a function from process variables to sequences of name variables. We define the typed encoding $\langle\llbracket\cdot\rrbracket_f^1, \langle\!\langle\cdot\rangle\!\rangle^1\rangle : \mathcal{L}_{\mathsf{HO}\pi} \to \mathcal{L}_{\mathsf{HO}}$ in Fig. 5. We assume that the mapping $\langle\!\langle\cdot\rangle\!\rangle^1$ on types is extended to session environments Δ and shared environments Γ homomorphically with:*

$$\langle\!\langle\Gamma \cdot X : \Delta\rangle\!\rangle^1 = \langle\!\langle\Gamma\rangle\!\rangle^1 \cdot z_X : (S_1, \ldots, S_m, S^*) \to \diamond$$

with $S^* = \mu\mathsf{t}.?((S_1, \ldots, S_m, \mathsf{t}) \to \diamond); \mathsf{end}$ and $\Delta = \{n_i : S_i\}_{1\le i\le m}$.

Note that Δ in $X : \Delta$ is mapped to a non-tail recursive session type with variable z_X (see recursion mappings in Fig. 5). Non-tail recursive session types have been studied in [3, 4]; to our knowledge, this is the first application in the context of higher-order session types. For simplicity of the presentation, we use name abstractions with polyadicity. A precise encoding of polyadicity into HO is given in § 8.2.

We explain the mapping in Fig. 5, focusing on *name passing* ($\llbracket u!\langle w\rangle.P\rrbracket_f^1$, $\llbracket u?(x).P\rrbracket_f^1$), and *recursion* ($\llbracket\mu X.P\rrbracket_f^1$ and $\llbracket X\rrbracket_f^1$).

Name passing. A name w is passed as an input-guarded abstraction; the abstraction receives a higher-order value and continues with the application of w over the received higher-order value. On the receiver side ($u?(x).P$) the encoding realises a mechanism that i) receives the input guarded abstraction, then ii) applies it on a fresh endpoint s, and iii) uses the dual endpoint $\overline{s}$ to send the continuation P as the abstraction $\lambda x.P$. Then name substitution is achieved via name application.

Recursion. The encoding of a recursive process $\mu X.P$ is delicate, for it must preserve the linearity of session endpoints. To this end, we first record a mapping from recursive variable X to process variable z_X. Then, we encode the recursion body P as a name abstraction in which free names of P are converted into name variables, using Def. 7.2. (Notice that P is first encoded into HO and then transformed using mapping $\llbracket\cdot\rrbracket_\sigma$.) Subsequently, this higher-order value is embedded in an input-guarded "duplicator" process [32]. Finally, we define the encoding of X in such a way that it simulates recursion unfolding by invoking the duplicator in a by-need fashion. That is, upon reception, the HO abstraction which encodes the recursion body P is duplicated: one copy is used to reconstitute the original recursion body P (through the application of $\mathtt{fn}(P)$); another copy is used to re-invoke the duplicator when needed. An example of this typed encoding is detailed in [1].

Theorem 7.4 (Precise Encoding of HOπ into HO). *The encoding from $\mathcal{L}_{\mathsf{HO}\pi}$ into $\mathcal{L}_{\mathsf{HO}}$ (cf. Def. 7.3) is precise.*

7.2 From HOπ to π

We now discuss the encodability of HO into π. We closely follow the encoding by Sangiorgi [26, 31], but casted in the setting of session-typed communications. Intuitively, such an encoding represents the exchange of a process with the exchange of a freshly generated *trigger name*. Trigger names may then be used to activate copies of the process, which becomes a persistent resource represented by an input-guarded replication. On the other hand, session names are linear resources and cannot be replicated. Our approach then uses replicated names as triggers for shared resources and non-replicated names for linear resources.

Definition 7.5 (Typed Encoding of HOπ into π). *We define the typed encoding $\langle\llbracket\cdot\rrbracket^2, \langle\!\langle\cdot\rangle\!\rangle^2\rangle : \mathcal{L}_{\mathsf{HO}\pi} \to \mathcal{L}_\pi$ in Fig. 6.*

Notice that $\langle\!\langle n?\langle\lambda x.P\rangle\rangle\!\rangle^2$ involves a fresh trigger name (linear or shared), which denotes the location of $\llbracket P\rrbracket^2$. Observe also how $\llbracket(\lambda x.P)u\rrbracket^2$ naturally induces a name substitution.

Theorem 7.6 (Precise Encoding of HOπ into π). *The encoding from $\mathcal{L}_{\mathsf{HO}\pi}$ into $\mathcal{L}_\pi$ (cf. Def. 7.5) is precise.*

7.3 A Negative Result

As most session calculi, HOπ includes communication on both shared and linear names. The former enables non determinism and unrestricted behaviour; the latter allows to represent deterministic and linear communication structures. The expressive power of

$$
\begin{array}{lrcl}
\textbf{Types}: & (\!|!\langle S\multimap\diamond\rangle;S_1|\!)^2 & \stackrel{\text{def}}{=} & !\langle\langle?((\!|S|\!)^2);\mathsf{end}\rangle\rangle;(\!|S_1|\!)^2 \\
& (\!|?(S\multimap\diamond);S_1|\!)^2 & \stackrel{\text{def}}{=} & ?(\langle?((\!|S|\!)^2);\mathsf{end}\rangle);(\!|S_1|\!)^2
\end{array}
$$

Terms:

$$
[\![u!\langle\lambda x.Q\rangle.P]\!]^2 \stackrel{\text{def}}{=} \begin{cases} (\nu a)(u!\langle a\rangle.([\![P]\!]^2 \mid *\,a?(y).y?(x).[\![Q]\!]^2)) \\ \qquad (s\notin \mathsf{fn}(Q)) \\ (\nu a)(u!\langle a\rangle.([\![P]\!]^2 \mid a?(y).y?(x).[\![Q]\!]^2)) \\ \qquad \text{(otherwise)} \end{cases}
$$

$$
\begin{array}{rcl}
[\![u?(x).P]\!]^2 & \stackrel{\text{def}}{=} & u?(x).[\![P]\!]^2 \\
[\![x\,u]\!]^2 & \stackrel{\text{def}}{=} & (\nu\, s)(x!\langle s\rangle.\overline{s}!\langle u\rangle.\mathbf{0}) \\
[\![(\lambda x.P)\,u]\!]^2 & \stackrel{\text{def}}{=} & (\nu\, s)(s?(x).[\![P]\!]^2 \mid \overline{s}!\langle u\rangle.\mathbf{0})
\end{array}
$$

Notice: $*\,P$ means $\mu X.(P \mid X)$. The rest of mappings are homomorphic.

Figure 6: Encoding of $\mathsf{HO}\pi$ into π.

shared names is also illustrated by our encoding from $\mathsf{HO}\pi$ into π (Thm. 7.6). This result begs the question: can we represent shared name interaction using session name interaction? Here we prove that shared names actually add expressiveness to $\mathsf{HO}\pi$: we show the non existence of a minimal encoding (cf. Def. 6.5) of shared name communication into linear communication.

Theorem 7.7. There is no minimal encoding from π to $\mathsf{HO}\pi^{-\mathsf{sh}}$. Hence, for any $\mathsf{C}_1,\mathsf{C}_2\in\{\mathsf{HO}\pi,\mathsf{HO},\pi\}$, there is no minimal encoding from $\mathcal{L}_{\mathsf{C}_1}$ into $\mathcal{L}_{\mathsf{C}_2^{-\mathsf{sh}}}$.

The proof of Thm. 7.7 relies on τ-inertness (Lem. **??**) and barb preservation (Prop. **??**). By Def. 7.3 and 7.5 and Prop. 7.4 and 7.6, we have:

Corollary 7.8. *Let* $\mathsf{C}_1,\mathsf{C}_2\in\{\mathsf{HO}\pi,\mathsf{HO},\pi\}$. *Then there exists a precise encoding from* $\mathcal{L}_{\mathsf{C}_1^{-\mathsf{sh}}}$ *into* $\mathcal{L}_{\mathsf{C}_2^{-\mathsf{sh}}}$.

8. Extensions

This section studies (i) the extension of $\mathsf{HO}\pi$ with higher-order applications/abstractions and (ii) the extension of $\mathsf{HO}\pi$ with polyadicity. In both cases, we detail required modifications in the syntax and types, and describe further encodability results.

8.1 Encoding $\mathsf{HO}\pi^+$ into $\mathsf{HO}\pi$

The calculus $\mathsf{HO}\pi^+$ extends $\mathsf{HO}\pi$ with higher-order abstractions and applications.

Syntax, Operational Semantics and Types. First, the syntax of Fig. 2 extends $V\,u$ to $V\,W$, including higher-order value W. The rule $(\lambda x.P)V\longrightarrow P\{^V/_x\}$ replaces rule [App] in Fig. 3. The syntax of types is modified as follows:

$$L \;::=\; U\to\diamond \;\mid\; U\multimap\diamond$$

These types can be easily accommodated in the type system: in Fig. 9, we replace C by U in [Abs] and C by U' in [App].

Behavioural Semantics. Labels remain the same. Rule $\langle\text{App}\rangle$ in the untyped LTS (Fig. **??**) is replaced with rule $(\lambda x.P)V\stackrel{\tau}{\longrightarrow}P\{^V/_x\}$. Def. **??** (characteristic processes) is extended with $(\!|U\to\diamond|\!)^x\stackrel{\text{def}}{=}(\!|U\multimap\diamond|\!)^x\stackrel{\text{def}}{=}x[\![U]\!]_{\mathsf{c}}$ and $(\!|U\to\diamond|\!)_{\mathsf{c}}\stackrel{\text{def}}{=}(\!|U\multimap\diamond|\!)_{\mathsf{c}}\stackrel{\text{def}}{=}\lambda x.[\![U]\!]^x$. We can then use the same definitions for $\cong$, $\approx$, $\approx^H$ and $\approx^C$.

Encoding $\mathsf{HO}\pi^+$ into $\mathsf{HO}\pi$. Let $\mathcal{L}_{\mathsf{HO}\pi^+}=\langle\mathsf{HO}\pi^+,\mathcal{T}_4,\stackrel{\ell}{\longmapsto},\approx_H,\vdash\rangle$ where $\mathcal{T}_4$ is a set of types of $\mathsf{HO}\pi^+$; the typing $\vdash$ is defined in Fig. 9 with extended rules [Abs] and [App]. We define the typed encoding

$$
\begin{array}{lrcl}
\textbf{Types}: & (\!|L\to\diamond|\!)^3 & \stackrel{\text{def}}{=} & ?((\!|L|\!)^3);\mathsf{end}\to\diamond \\
& (\!|!\langle L\to\diamond\rangle;S|\!)^3 & \stackrel{\text{def}}{=} & !\langle(\!|L\to\diamond|\!)^3\rangle;(\!|S|\!)^3 \\
& (\!|?(L\to\diamond);S|\!)^3 & \stackrel{\text{def}}{=} & ?((\!|L\to\diamond|\!)^3);(\!|S|\!)^3
\end{array}
$$

Terms:

$$
\begin{array}{rcl}
[\![x(\lambda y.P)]\!]^3 & \stackrel{\text{def}}{=} & (\nu\, s)(x\,s\mid\overline{s}!\langle\lambda y.[\![P]\!]^3\rangle.\mathbf{0}) \\
[\![u!\langle\lambda x:L.Q\rangle.P]\!]^3 & \stackrel{\text{def}}{=} & u!\langle\lambda z.z?(x).[\![Q]\!]^3\rangle.[\![P]\!]^3 \\
[\![(\lambda x.P)\,(\lambda x.Q)]\!]^3 & \stackrel{\text{def}}{=} & (\nu\, s)(s?(x).[\![P]\!]^3\mid\overline{s}!\langle\lambda x.[\![Q]\!]^3\rangle.\mathbf{0})
\end{array}
$$

$(\!|L\multimap\diamond|\!)^3$ is defined as $(\!|L\to\diamond|\!)^3$ by replacing $L\to\diamond$ with $L\multimap\diamond$. Label and term mappings for $\lambda x:C.P$ are as in Fig. 5, replacing $(\!|\cdot|\!)^1$, $\{\!|\cdot|\!\}^1$, and $[\![\cdot]\!]^1_f$, by $(\!|\cdot|\!)^3$, $\{\!|\cdot|\!\}^3$, and $[\![\cdot]\!]^3$.

The other mappings for processes, types and labels are homomorphic.

Figure 7: Encoding of $\mathsf{HO}\pi^+$ into $\mathsf{HO}\pi$.

$\langle[\![\cdot]\!]^3,(\!|\cdot|\!)^3\rangle:\mathsf{HO}\pi^+\to\mathsf{HO}\pi$ in Fig. 7. By Prop. 6.6, we derive the following theorem.

Theorem 8.1 (Encoding $\mathsf{HO}\pi^+$ into $\mathsf{HO}\pi$). *The encoding from* $\mathcal{L}_{\mathsf{HO}\pi^+}$ *into* $\mathcal{L}_{\mathsf{HO}\pi}$ *(cf. Fig. 7) is precise. Hence, the encodings from* $\mathcal{L}_{\mathsf{HO}\pi^+}$ *to* $\mathcal{L}_{\mathsf{HO}}$ *and* $\mathcal{L}_\pi$ *are also precise.*

8.2 Encoding Polyadic $\mathsf{HO}\pi$ into $\mathsf{HO}\pi$

Embeddings of polyadic name passing into monadic name passing are well-studied. Using a linear typing, precise encodings (including full abstraction) can be obtained [36]. Here we summarise how $\mathsf{HO}\vec{\pi}$ can be encoded into $\mathsf{HO}\pi$. The syntax of $\mathsf{HO}\pi$ is extended with polyadic name passing $\tilde{n}$ and $\lambda\tilde{x}.Q$ in the syntax of value V. The type syntax is extended to:

$$L ::= \tilde{C}\to\diamond \mid \tilde{C}\multimap\diamond \qquad S \;::=\; !\langle\tilde{U}\rangle;S \;\mid\; ?(\tilde{U});S \;\mid\; \cdots$$

The type system disallows a shared name that directly carries polyadic shared names as in [19, 20]. Other definitions are straightforwardly extended. We slightly modify Def. 6.2 to capture that a label ℓ may be mapped into a sequence of labels $\tilde{\ell}$. Also, Def. 6.4 is kept unchanged, assuming that if $P\stackrel{\ell}{\longmapsto}P'$ and $\{\!|\ell|\!\}=\ell_1,\ell_2,\cdots,\ell_m$ then $[\![P]\!]\stackrel{\{\!|\ell|\!\}}{\Longmapsto}[\![P']\!]$ should be understood as $[\![P]\!]\stackrel{\ell_1}{\Longmapsto}P_1\stackrel{\ell_2}{\Longmapsto}P_2\cdots\stackrel{\ell_m}{\Longmapsto}P_m=[\![P']\!]$, for some $P_1,P_2,\ldots,P_m$.

Let $\mathcal{L}_{\mathsf{HO}\vec{\pi}}=\langle\mathsf{HO}\vec{\pi},\mathcal{T}_5,\stackrel{\ell}{\longmapsto},\approx_H,\vdash\rangle$ where $\mathcal{T}_5$ is a set of types of $\mathsf{HO}\pi^+$; the typing $\vdash$ is defined in Fig. 9 with polyadic types. We define the typed encoding $\langle[\![\cdot]\!]^4,(\!|\cdot|\!)^4\rangle:\mathsf{HO}\vec{\pi}\to\mathsf{HO}\pi$ in Fig. 8. We give the dyadic case (tuples of length 2), for simplicity; the general case is as expected. Then we have:

Theorem 8.2 (Encoding of $\mathsf{HO}\vec{\pi}$ into $\mathsf{HO}\pi$). *The encoding from* $\mathcal{L}_{\mathsf{HO}\vec{\pi}}$ *into* $\mathcal{L}_{\mathsf{HO}\pi}$ *(cf. Fig. 8) is precise. Hence, the encodings from* $\mathcal{L}_{\mathsf{HO}\vec{\pi}}$ *to* $\mathcal{L}_{\mathsf{HO}}$ *and* $\mathcal{L}_\pi$ *are also precise.*

By combining Thms. 8.1 and 8.2, we can easily extend the preciseness to $\mathsf{HO}\vec{\pi}^+$, the super-calculus of $\mathsf{HO}\pi^+$ and $\mathsf{HO}\vec{\pi}$.

9. Related Work

Expressiveness. There is a vast literature on expressiveness studies for process calculi. For space reasons here we concentrate on closely related work; see [1] for more detailed comparisons with other literature. The encoding of process-passing into name-passing is well-known [25]; an encoding in the reverse direction is given in [31] for an asynchronous, localised π-calculus (only the output capability of names can be sent around). The work [28] studies hierarchies for calculi with *internal* first-order mobility and with

$$
\begin{array}{rrcl}
\textbf{Types}: & (\!|!\langle S_1, S_2\rangle; S|\!)^4 & \stackrel{\text{def}}{=} & !\langle (\!|S_1|\!)^4\rangle; !\langle (\!|S_2|\!)^4\rangle; (\!|S|\!)^4 \\
& (\!|!\langle L\rangle; S|\!)^4 & \stackrel{\text{def}}{=} & !\langle (\!|L|\!)^4\rangle; (\!|S|\!)^4 \\
& (\!|(C_2, C_2)\to\diamond|\!)^4 & \stackrel{\text{def}}{=} & (?(\!|C_1|\!)^4); ?(\!|C_2|\!)^4); \mathsf{end})\to\diamond \\
& (\!|(C_1, C_2)\multimap\diamond|\!)^4 & \stackrel{\text{def}}{=} & (?(\!|C_1|\!)^4); ?(\!|C_2|\!)^4); \mathsf{end})\multimap\diamond \\
\hline
\textbf{Terms}: & [\![u!\langle u_1, u_2\rangle.P]\!]^4 & \stackrel{\text{def}}{=} & u!\langle u_1\rangle.u!\langle u_2\rangle.[\![P]\!]^4 \\
& [\![u!\langle\lambda(x_1, x_2).Q\rangle.P]\!]^4 & \stackrel{\text{def}}{=} & u!\langle\lambda z.z?(x_1).z?(x_2).[\![Q]\!]^4\rangle.[\![P]\!]^4 \\
& [\![x(u_1, u_2)]\!]^4 & \stackrel{\text{def}}{=} & (\nu\, s)(x\, s \mid \overline{s}!\langle u_1\rangle.\overline{s}!\langle u_2\rangle.\mathbf{0}) \\
& [\![(\lambda(x_1, x_2).P)(u_1, u_2)]\!]^4 & \stackrel{\text{def}}{=} & \\
& & & (\nu\, s)(s?(x_1).s?(x_2).[\![P]\!]^4 \mid \overline{s}!\langle u_1\rangle.\overline{s}!\langle u_2\rangle.\mathbf{0})
\end{array}
$$

We give the dyadic case; the general polyadic case is as expected. The input cases are defined as the outputs replacing ! by ?. Mappings for the other processes/types/labels are homomorphic.

Figure 8: Encoding of $\mathsf{HO}\vec{\pi}$ into $\mathsf{HO}\pi$.

higher-order mobility without name-passing (like HO). The hierarchies are defined according to the order of types needed in typing. Via type-preserving fully abstract encodings, it is shown that name- and process-passing calculi with equal order of types have the same expressiveness. With respect to these previous results, our approach based on session types has important consequences and allows us to derive new results. Our study stresses the view of "encodings as protocols", namely session protocols which enforce clean linear and shared disciplines for names, a distinction not explored in [25, 29]. In turn, this distinction is central in proper definitions of trigger processes, which are key to encodings (Def. 7.5) and behavioural equivalences (Def. **??** and **??**). More interestingly, we showed that HO suffices to encode the session calculus with name passing (π) but also $\mathsf{HO}\pi$ and its extension with higher-order applications ($\mathsf{HO}\pi^+$). Thus, all these session calculi are equally expressive with fully abstract encodings. To our knowledge, these are the first expressivity results of this kind.

Building upon [32], the work [34] studies the (non)encodability of the π-calculus into a higher-order π-calculus with a powerful name relabelling operator, which is shown to be essential in encoding name-passing. A core higher-order calculus is studied in [18]: it lacks restriction, name passing, output prefix and constructs for infinite behaviour. This calculus has a simple notion of bisimilarity which coincides with contextual equivalence. The absence of restriction plays a key role in the characterisations in [18]; hence, our characterisation of contextual equivalence for HO (which has restriction) cannot be derived from that in [18].

In [17] the core calculus in [18] is extended with restriction, synchronous communication, and polyadicity. It is shown that synchronous communication can encode asynchronous communication, and that process passing polyadicity induces a hierarchy in expressive power. Encodability criteria does not include full abstraction. The paper [35] complements [17] by studying the expressivity of second-order process abstractions. Polyadicity is shown to induce an expressiveness hierarchy; also, by adapting the encoding in [25], process abstractions are encoded into name abstractions. In contrast, we give a fully abstract encoding of $\mathsf{HO}\vec{\pi}^+$ into HO that preserves session types; this improves [17, 35] by enforcing linearity disciplines on process behaviour. Also, the focus of [17, 35] is on the expressiveness of untyped, higher-order processes; they do not address tractable equivalences for processes (such as $\approx^H$ and $\approx^C$) which only require observation of finite values, whose formulations rely on session types.

Session Typed Processes. The works [5, 6] study encodings of binary session calculi into a linearly typed π-calculus. While [6] gives a precise encoding of π into a linear calculus (an extension of [2]), the work [5] gives operational correspondence (without full abstraction, cf. Def. 6.3-4) for the first- and higher-order π-calculi into [12]. They investigate embeddability of two different typing systems; by the result of [6], $\mathsf{HO}\pi^+$ is encodable into the linearly typed π-calculi.

The syntax of $\mathsf{HO}\pi$ is a subset of that in [19, 20]. The work [19] develops a full higher-order session calculus with process abstractions and applications; it admits the type $U = U_1 \to U_2 \ldots U_n \to \diamond$ and its linear type U^1 which corresponds to $\tilde{U}\to\diamond$ and $\tilde{U}\multimap\diamond$ in a super-calculus of $\mathsf{HO}\pi^+$ and $\mathsf{HO}\vec{\pi}$. Our results show that the calculus in [19] is not only expressed but also reasoned in HO (with limited form of arrow types, $C\to\diamond$ and $C\multimap\diamond$), via precise encodings. None of the above works proposes tractable bisimulations for higher-order processes.

Typed Behavioural Equivalences. This work follows the session type behavioural semantics in [15, 16, 24] where a bisimulation is defined on a LTS that assumes a session typed observer. Our theory for higher-order session types differentiates from the work in [15, 16], which considers the first-order binary and multiparty session types, respectively. The work [24] gives a behavioural theory for a logically motivated language of binary sessions without shared names.

Our approach to typed equivalences builds upon techniques by Sangiorgi [25, 27] and Jeffrey and Rathke [11]. The work [25] introduced the first fully-abstract encoding from the higher-order π-calculus into the π-calculus. Sangiorgi's encoding is based on the idea of a replicated input-guarded process (a trigger process). We use a similar replicated triggered process to encode $\mathsf{HO}\pi$ into π (Def. 7.5). Operational correspondence for the triggered encoding is shown using a context bisimulation with first-order labels. To deal with the issue of context bisimilarity, Sangiorgi proposes *normal bisimilarity*, a tractable equivalence without universal quantification. To prove that context and normal bisimilarities coincide, [25] uses triggered processes. Triggered bisimulation is also defined on first-order labels where the context bisimulation is restricted to arbitrary trigger substitution. This characterisation of context bisimilarity was refined in [11] for calculi with recursive types, not addressed in [25, 27] and quite relevant in our work (cf. Def. 7.3). The bisimulation in [11] is based on an LTS which is extended with trigger meta-notation. As in [25, 27], the LTS in [11] observes first-order triggered values instead of higher-order values, offering a more direct characterisation of contextual equivalence and lifting the restriction to finite types. We briefly contrast the approach in [11] and ours based on higher-order ($\approx^H$) and characteristic ($\approx^C$) bisimilarities:

- The LTS in [11] is enriched with extra labels for triggers; an output action transition emits a trigger and introduces a parallel replicated trigger. Our approach retains usual labels/transitions; in case of output, $\approx^H$ and $\approx^C$ introduce a parallel non-replicated trigger.
- Higher-order input in [11] involves the input of a trigger which reduces after substitution. Rather than a trigger name, $\approx^H$ and $\approx^C$ decree the input of a triggered value $\lambda z.t?(x).xz$.
- Unlike [11], $\approx^C$ treats first- and higher-order values uniformly. As the typed LTS distinguishes linear and shared values, replicated closures are used only for shared values.
- In [11] a matching construct is crucial to prove completeness of bisimilarity, while our calculi lack matching. In contrast, we use the characteristic process interaction with the environment, exploiting session type structures, i.e., instead of matching a name is embedded into a process and then observe its behaviour.

The *environmental bisimulations* given in [30] use a higher-order LTS to define a bisimulation that stores the observer's knowledge; hence, observed actions are based on this knowledge at any given

time. This approach is enhanced in [13, 14] where a mapping from constants to higher-order values is introduced. This allows to observe first-order values instead of higher-order values. It differs from [11, 27] in that the mapping between higher- and first-order values is no longer implicit.

References

[1] Full version of this paper. Technical report, Imperial College / Univ. of Groningen, 2015. `http://www.jorgeaperez.net/publications/coresessions`.

[2] M. Berger, K. Honda, and N. Yoshida. Sequentiality and the π-calculus. In *Proc. TLCA'01*, volume 2044 of *LNCS*, pages 29–45, 2001.

[3] G. Bernardi, O. Dardha, S. J. Gay, and D. Kouzapas. On duality relations for session types. In *TGC*, LNCS, 2014. To appear.

[4] V. Bono and L. Padovani. Typing copyless message passing. *LMCS*, 8(1), 2012.

[5] O. Dardha, E. Giachino, and D. Sangiorgi. Session types revisited. PPDP, pages 139–150. ACM, 2012.

[6] R. Demangeon and K. Honda. Full abstraction in a subtyped pi-calculus with linear types. In *CONCUR*, volume 6901 of *LNCS*, pages 280–296. Springer, 2011.

[7] Y. Fu and H. Lu. On the expressiveness of interaction. *Theor. Comput. Sci.*, 411(11-13):1387–1451, 2010.

[8] S. J. Gay and V. T. Vasconcelos. Linear type theory for asynchronous session types. *J. Funct. Program.*, 20(1):19–50, 2010.

[9] D. Gorla. Towards a unified approach to encodability and separation results for process calculi. *Inf. Comput.*, 208(9):1031–1053, 2010.

[10] K. Honda, V. T. Vasconcelos, and M. Kubo. Language primitives and type disciplines for structured communication-based programming. In *ESOP'98*, volume 1381 of *LNCS*, pages 22–138. Springer, 1998.

[11] A. Jeffrey and J. Rathke. Contextual equivalence for higher-order pi-calculus revisited. *LMCS*, 1(1), 2005.

[12] N. Kobayashi, B. C. Pierce, and D. N. Turner. Linearity and the Pi-Calculus. *TOPLAS*, 21(5):914–947, Sept. 1999.

[13] V. Koutavas and M. Hennessy. A testing theory for a higher-order cryptographic language. In *ESOP*, volume 6602 of *LNCS*, pages 358–377, 2011.

[14] V. Koutavas and M. Hennessy. First-order reasoning for higher-order concurrency. *Computer Languages, Systems & Structures*, 38(3):242–277, 2012.

[15] D. Kouzapas and N. Yoshida. Globally governed session semantics. *LMCS*, 10(4), 2014.

[16] D. Kouzapas, N. Yoshida, R. Hu, and K. Honda. On asynchronous eventful session semantics. *MSCS*, 2015.

[17] I. Lanese, J. A. Pérez, D. Sangiorgi, and A. Schmitt. On the expressiveness of polyadic and synchronous communication in higher-order process calculi. In *ICALP*, volume 6199, pages 442–453, 2010.

[18] I. Lanese, J. A. Pérez, D. Sangiorgi, and A. Schmitt. On the expressiveness and decidability of higher-order process calculi. *Inf. Comput.*, 209(2):198–226, 2011.

[19] D. Mostrous and N. Yoshida. Two session typing systems for higher-order mobile processes. In *TLCA*, volume 4583 of *LNCS*, pages 321–335. Springer, 2007.

[20] D. Mostrous and N. Yoshida. Session Typing and Asynchronous Subtying for Higher-Order π-Calculus. *Info.& Comp.*, 2015. To appear.

[21] U. Nestmann. What is a "good" encoding of guarded choice? *Inf. Comput.*, 156(1-2):287–319, 2000.

[22] C. Palamidessi. Comparing the expressive power of the synchronous and asynchronous pi-calculi. *MSCS*, 13(5):685–719, 2003.

[23] C. Palamidessi, V. A. Saraswat, F. D. Valencia, and B. Victor. On the expressiveness of linearity vs persistence in the asychronous pi-calculus. In *Proc. of LICS 2006*, pages 59–68, 2006.

[24] J. A. Pérez, L. Caires, F. Pfenning, and B. Toninho. Linear logical relations and observational equivalences for session-based concurrency. *Inf. Comput.*, 239:254–302, 2014.

[25] D. Sangiorgi. *Expressing Mobility in Process Algebras: First-Order and Higher Order Paradigms*. PhD thesis, University of Edinburgh, 1992.

[26] D. Sangiorgi. The lazy lambda calculus in a concurrency scenario. In *7th LICS Conf.*, pages 102–109. IEEE Computer Society Press, 1992.

[27] D. Sangiorgi. Bisimulation for Higher-Order Process Calculi. *Inf. & Comp.*, 131(2):141–178, 1996.

[28] D. Sangiorgi. π-calculus, internal mobility and agent-passing calculi. *TCS*, 167(2):235–274, 1996.

[29] D. Sangiorgi. Asynchronous process calculi: the first- and higher-order paradigms. *Theor. Comput. Sci.*, 253(2):311–350, 2001.

[30] D. Sangiorgi, N. Kobayashi, and E. Sumii. Environmental bisimulations for higher-order languages. In *LICS*, pages 293–302. IEEE, 2007.

[31] D. Sangiorgi and D. Walker. *The π-calculus: a Theory of Mobile Processes*. Cambridge University Press, 2001.

[32] B. Thomsen. Plain CHOCS: A Second Generation Calculus for Higher Order Processes. *Acta Informatica*, 30(1):1–59, 1993.

[33] R. J. van Glabbeek. Musings on encodings and expressiveness. In *Proc. of EXPRESS/SOS 2012*, volume 89 of *EPTCS*, pages 81–98, 2012.

[34] X. Xu. Distinguishing and relating higher-order and first-order processes by expressiveness. *Acta Informatica*, 49(7-8):445–484, 2012.

[35] X. Xu, Q. Yin, and H. Long. On the expressiveness of parameterization in process-passing. In *WS-FM*, volume 8379, pages 147–167, 2014.

[36] N. Yoshida. Graph types for monadic mobile processes. In *FSTTCS*, volume 1180 of *LNCS*, pages 371–386. Springer, 1996.

A. Appendix: the Typing System of HOπ

In this appendix we formally define *type equivalence* and *duality*. We also present and describe our typing rules, given in Fig. 9.

Definition A.1 (Type Equivalence). *Let* ST *a set of closed session types. Two types S and S' are said to be* isomorphic *if a pair (S, S') is in the largest fixed point of the monotone function $F : \mathcal{P}(\mathsf{ST}\times\mathsf{ST}) \to \mathcal{P}(\mathsf{ST}\times\mathsf{ST})$ defined by:*

$$
\begin{array}{rcl}
F(\mathfrak{R}) & = & \{(\mathsf{end},\mathsf{end})\} \\
& \cup & \{(!\langle U_1\rangle;S_1, !\langle U_2\rangle;S_2) \mid (S_1,S_2),(U_1,U_2)\in\mathfrak{R}\} \\
& \cup & \{(?(U_1);S_1, ?(U_2);S_2) \mid (S_1,S_2),(U_1,U_2)\in\mathfrak{R}\} \\
& \cup & \{(\&\{l_i : S_i\}_{i\in I}, \&\{l_i : S'_i\}_{i\in I}) \mid \forall i\in I.(S_i,S'_i)\in\mathfrak{R}\} \\
& \cup & \{(\oplus\{l_i : S_i\}_{i\in I}, \oplus\{l_i : S'_i\}_{i\in I}) \mid \forall i\in I.(S_i,S'_i)\in\mathfrak{R}\} \\
& \cup & \{(\mu \mathrm{t}.S, S') \mid (S\{^{\mu \mathrm{t}.S}/_{\mathrm{t}}\}, S')\in\mathfrak{R}\} \\
& \cup & \{(S, \mu \mathrm{t}.S') \mid (S, S'\{^{\mu \mathrm{t}.S'}/_{\mathrm{t}}\})\in\mathfrak{R}\}
\end{array}
$$

Standard arguments ensure that F is monotone, thus the greatest fixed point of F exists. We write $S_1 \sim S_2$ if $(S_1, S_2)\in\mathfrak{R}$.

Definition A.2 (Duality). *Let* ST *a set of closed session types. Two types S and S' are said to be* dual *if a pair (S, S') is in the largest fixed point of the monotone function $F : \mathcal{P}(\mathsf{ST}\times\mathsf{ST}) \to \mathcal{P}(\mathsf{ST}\times\mathsf{ST})$ defined by:*

$$
\begin{array}{rcl}
F(\mathfrak{R}) & = & \{(\mathsf{end},\mathsf{end})\} \\
& \cup & \{(!\langle U_1\rangle;S_1, ?(U_2);S_2) \mid (S_1,S_2)\in\mathfrak{R},\ U_1\sim U_2\} \\
& \cup & \{(?(U_1);S_1, !\langle U_2\rangle;S_2) \mid (S_1,S_2)\in\mathfrak{R},\ U_1\sim U_2\} \\
& \cup & \{(\oplus\{l_i : S_i\}_{i\in I}, \&\{l_i : S'_i\}_{i\in I}) \mid \forall i\in I.(S_i,S'_i)\in\mathfrak{R}\} \\
& \cup & \{(\&\{l_i : S_i\}_{i\in I}, \oplus\{l_i : S'_i\}_{i\in I}) \mid \forall i\in I.(S_i,S'_i)\in\mathfrak{R}\} \\
& \cup & \{(\mu \mathrm{t}.S, S') \mid (S\{^{\mu \mathrm{t}.S}/_{\mathrm{t}}\}, S')\in\mathfrak{R}\} \\
& \cup & \{(S, \mu \mathrm{t}.S') \mid (S, S'\{^{\mu \mathrm{t}.S'}/_{\mathrm{t}}\})\in\mathfrak{R}\}
\end{array}
$$

Standard arguments ensure that F is monotone, thus the greatest fixed point of F exists. We write S_1 dual S_2 if $(S_1, S_2)\in\mathfrak{R}$.

Typing System of HOπ The typing system is defined in Fig. 9. Rules [Sess, Sh, LVar] are name and variable introduction rules. The type $C\to\diamond$ is derived using rule [Prom], where we require a value with linear type to be typed without a linear environment to be typed as a shared type. Rule [EProm] allows to freely use a linear type variable as shared.

Abstraction values are typed with rule [Abs]. The dual of abstraction typing is application typing governed by rule [App], where we expect the type C of an application name u to match the type $C\multimap\diamond$ or $C\to\diamond$ of the application variable x.

In [Send], the type U of a send value V should appear as a prefix on the session type $!\langle U\rangle;S$ of u. [Rcv] is its dual. We use a similar approach with session prefixes to type interaction between shared names as defined in rules [Req] and [Acc], where the type of the sent/received object (S and L, respectively) should match the type of the sent/received subject ($\langle S\rangle$ and $\langle L\rangle$, respectively). Rules for selection and branching, [Sel] and [Bra], are standard.

A shared name creation a creates and restricts a in environment Γ as defined in rule [Res]. Creation of a session name s creates and restricts two endpoints with dual types in rule [ResS]. Rule [Par], combines the environments Λ and Δ of the parallel components of a parallel process. The disjointness of environments Λ and Δ is implied. Rule [End] adds the names with type end in Δ. The recursion requires that the body process matches the type of the recursive variable as in rule [Rec]. The recursive variable is typed directly from the shared environment Γ as in rule [RVar]. The inactive process $\mathbf{0}$ is typed with no linear environments as in [Nil].

[Sess] $\Gamma;\emptyset;\{u : S\}\vdash u\triangleright S$ [Sh] $\Gamma\cdot u : U;\emptyset;\emptyset\vdash u\triangleright U$

[LVar] $\Gamma;\{x : C\multimap\diamond\};\emptyset\vdash x\triangleright C\multimap\diamond$

[Prom] $\dfrac{\Gamma;\emptyset;\emptyset\vdash V\triangleright C\multimap\diamond}{\Gamma;\emptyset;\emptyset\vdash V\triangleright C\to\diamond}$ [EProm] $\dfrac{\Gamma;\Lambda\cdot x : C\multimap\diamond;\Delta\vdash P\triangleright\diamond}{\Gamma\cdot x : C\to\diamond;\Lambda;\Delta\vdash P\triangleright\diamond}$

[Abs] $\dfrac{\Gamma;\Lambda;\Delta_1\vdash P\triangleright\diamond \quad \Gamma;\emptyset;\Delta_2\vdash x\triangleright C}{\Gamma\backslash x;\Lambda;\Delta_1\backslash\Delta_2\vdash\lambda x.P\triangleright C\multimap\diamond}$

[App] $\dfrac{U = C\multimap\diamond\vee C\to\diamond \quad \Gamma;\Lambda;\Delta_1\vdash V\triangleright U \quad \Gamma;\emptyset;\Delta_2\vdash u\triangleright C}{\Gamma;\Lambda;\Delta_1\cdot\Delta_2\vdash V\,u\triangleright\diamond}$

[Send] $\dfrac{\Gamma;\Lambda_1;\Delta_1\vdash P\triangleright\diamond \quad \Gamma;\Lambda_2;\Delta_2\vdash V\triangleright U \quad u : S\in\Delta_1\cdot\Delta_2}{\Gamma;\Lambda_1\cdot\Lambda_2;((\Delta_1\cdot\Delta_2)\setminus u : S)\cdot u :!\langle U\rangle;S\vdash u!\langle V\rangle.P\triangleright\diamond}$

[Rcv] $\dfrac{\Gamma;\Lambda_1;\Delta_1\cdot u : S\vdash P\triangleright\diamond \quad \Gamma;\Lambda_2;\Delta_2\vdash x\triangleright U}{\Gamma\backslash x;\Lambda_1\cdot\Lambda_2;\Delta_1\backslash\Delta_2\cdot u :?(U);S\vdash u?(x).P\triangleright\diamond}$

[Req] $\dfrac{\Gamma;\emptyset;\emptyset\vdash u\triangleright U_1 \quad \Gamma;\Lambda;\Delta_1\vdash P\triangleright\diamond \quad \Gamma;\emptyset;\Delta_2\vdash V\triangleright U_2 \quad (U_1 = \langle S\rangle\wedge U_2 = S)\vee(U_1 = \langle L\rangle\wedge U_2 = L)}{\Gamma;\Lambda;\Delta_1\cdot\Delta_2\vdash u!\langle V\rangle.P\triangleright\diamond}$

[Acc] $\dfrac{\Gamma;\emptyset;\emptyset\vdash u\triangleright U_1 \quad \Gamma;\Lambda_1;\Delta_1\vdash P\triangleright\diamond \quad \Gamma;\Lambda_2;\Delta_2\vdash x\triangleright U_2 \quad (U_1 = \langle S\rangle\wedge U_2 = S)\vee(U_1 = \langle L\rangle\wedge U_2 = L)}{\Gamma\backslash x;\Lambda_1\backslash\Lambda_2;\Delta_1\backslash\Delta_2\vdash u?(x).P\triangleright\diamond}$

[Bra] $\dfrac{\forall i\in I \quad \Gamma;\Lambda;\Delta\cdot u : S_i\vdash P_i\triangleright\diamond}{\Gamma;\Lambda;\Delta\cdot u : \&\{l_i : S_i\}_{i\in I}\vdash u\triangleright\{l_i : P_i\}_{i\in I}\triangleright\diamond}$

[Sel] $\dfrac{\Gamma;\Lambda;\Delta\cdot u : S_j\vdash P\triangleright\diamond \quad j\in I}{\Gamma;\Lambda;\Delta\cdot u : \oplus\{l_i : S_i\}_{i\in I}\vdash u\triangleleft l_j.P\triangleright\diamond}$

[ResS] $\dfrac{\Gamma;\Lambda;\Delta\cdot s : S_1\cdot\overline{s} : S_2\vdash P\triangleright\diamond \quad S_1\ \mathsf{dual}\ S_2}{\Gamma;\Lambda;\Delta\vdash(\nu\, s)P\triangleright\diamond}$

[Res] $\dfrac{\Gamma\cdot a : \langle S\rangle;\Lambda;\Delta\vdash P\triangleright\diamond}{\Gamma;\Lambda;\Delta\vdash(\nu a)P\triangleright\diamond}$ [Par] $\dfrac{\Gamma;\Lambda_i;\Delta_i\vdash P_i\triangleright\diamond \quad i = 1,2}{\Gamma;\Lambda_1\cdot\Lambda_2;\Delta_1\cdot\Delta_2\vdash P_1\mid P_2\triangleright\diamond}$

[End] $\dfrac{\Gamma;\Lambda;\Delta\vdash P\triangleright T \quad u\notin\mathsf{dom}(\Gamma,\Lambda,\Delta)}{\Gamma;\Lambda;\Delta\cdot u : \mathsf{end}\vdash P\triangleright\diamond}$ [Rec] $\dfrac{\Gamma\cdot X : \Delta;\emptyset;\Delta\vdash P\triangleright\diamond}{\Gamma;\emptyset;\Delta\vdash\mu X.P\triangleright\diamond}$

[RVar] $\Gamma\cdot X : \Delta;\emptyset;\Delta\vdash X\triangleright\diamond$ [Nil] $\Gamma;\emptyset;\emptyset\vdash\mathbf{0}\triangleright\diamond$

Figure 9: Typing Rules for HOπ.

B. Expressiveness Results

In this section we give the proofs for the expressiveness results stated in Section 7 an in Section 8.

For the purpose of proving Operational Correspondence in this section we prove a stronger result than the result suggested in Def. 6.4(2). We state the requirements below.

Definition B.1 (Operational Correspondence). Consider typed calculi $\mathcal{L}_1 = \langle \mathsf{C}_1, \mathcal{T}_1, \overset{\ell_1}{\longmapsto}_1, \approx_1, \vdash_1 \rangle$ and $\mathcal{L}_2 = \langle \mathsf{C}_2, \mathcal{T}_2, \overset{\ell_2}{\longmapsto}_2, \approx_2, \vdash_2 \rangle$. Let $\mathcal{A}_i$ be the set of labels from relation $\overset{\ell_i}{\longmapsto}$ $(i = 1,2)$ and let mapping $\{\!|\cdot|\!\} : \mathcal{A}_1 \to \mathcal{A}_2$.

We say that $\langle [\![\cdot]\!], (\!|\cdot|\!) \rangle : \mathcal{L}_1 \to \mathcal{L}_2$ is a *operational corresponce* if it satisfies the property: If $\Gamma; \emptyset; \Delta \vdash_1 P \triangleright \diamond$ then

1. Completeness: If $\Gamma; \Delta \vdash_1 P \overset{\ell_1}{\longmapsto}_1 \Delta' \vdash_1 P'$ then
   $\exists Q, \Delta''$ s.t. (i) $(\!|\Gamma|\!); (\!|\Delta|\!) \vdash_2 [\![P]\!] \overset{\ell_2}{\Longmapsto}_2 (\!|\Delta''|\!) \vdash_2 Q$ (ii) $\ell_2 = \{\!|\ell_1|\!\}$, and
   (ii) $(\!|\Gamma|\!); (\!|\Delta''|\!) \vdash_2 Q \approx_2 (\!|\Delta'|\!) \vdash_2 [\![P']\!]$.
2. Soundness: If $(\!|\Gamma|\!); (\!|\Delta|\!) \vdash_2 [\![P]\!] \overset{\ell_2}{\Longmapsto}_2 (\!|\Delta'|\!) \vdash_2 Q$ then
   $\exists P', \Delta''$ s.t. (i) $\Gamma; \Delta \vdash_1 P \overset{\ell_1}{\longmapsto}_1 \Delta'' \vdash_1 P'$ (ii) $\ell_2 = \{\!|\ell_1|\!\}$, and
   (ii) $(\!|\Gamma|\!); (\!|\Delta''|\!) \vdash_2 [\![P']\!] \approx_2 (\!|\Delta'|\!) \vdash_2 Q$.

Proposition B.2. *Consider typed calculi $\mathcal{L}_1 = \langle \mathsf{C}_1, \mathcal{T}_1, \overset{\ell_1}{\longmapsto}_1, \approx_1, \vdash_1 \rangle$ and $\mathcal{L}_2 = \langle \mathsf{C}_2, \mathcal{T}_2, \overset{\ell_2}{\longmapsto}_2, \approx_2, \vdash_2 \rangle$. Let $\mathcal{A}_i$ be the set of labels from relation $\overset{\ell_i}{\longmapsto}$ $(i = 1,2)$ and let mapping $\{\!|\cdot|\!\} : \mathcal{A}_1 \to \mathcal{A}_2$.*

If $\{\!|\tau|\!\} = \tau$ then the requirements of Def. B.1 imply the requirements of Def. 6.4(2).

Proof. The proof is trivial by substituting the τ action on labels ℓ_1 and ℓ_2 in Def. B.1 to get Def. 6.4(2).

B.1 Properties for encoding $\mathcal{L}_{\mathsf{HO}\pi}$ into $\mathcal{L}_{\mathsf{HO}}$

In this section we prove Thm. 7.4, in Page 4 that requires that encoding $\mathcal{L}_{\mathsf{HO}\pi}$ into $\mathcal{L}_{\mathsf{HO}}$ is precise. A precise encoding requires to prove three independent results:

- Type preservation, stated in Prop. B.3.
- Operational Correspondence, stated in Prop. B.5. Note that we prove a stronger operational correspondence condition, as in Def. B.1, than the condition suggested in Def. 6.4(2).
- Full Abstraction, stated in Prop. B.6.

Proposition B.3 (Type Preservation, $\mathsf{HO}\pi$ into HO). *Let P be a $\mathsf{HO}\pi$ process. If $\Gamma; \emptyset; \Delta \vdash P \triangleright \diamond$ then $(\!|\Gamma|\!)^1; \emptyset; (\!|\Delta|\!)^1 \vdash [\![P]\!]^1_f \triangleright \diamond$.*

Proof. By induction on the inference of $\Gamma; \emptyset; \Delta \vdash P \triangleright \diamond$.

1. Case $P = k!\langle n \rangle.P'$. There are two sub-cases. In the first sub-case $n = k'$ (output of a linear channel). Then we have the following typing in the source language:
$$\frac{\Gamma; \emptyset; \Delta \cdot k : S \vdash P' \triangleright \diamond \quad \Gamma; \emptyset; \{k' : S_1\} \vdash k' \triangleright S_1}{\Gamma; \emptyset; \Delta \cdot k' : S_1 \cdot k : !\langle S_1 \rangle; S \vdash k!\langle k' \rangle.P' \triangleright \diamond}$$
Thus, by IH we have
$$(\!|\Gamma|\!)^1; \emptyset; (\!|\Delta|\!)^1 \cdot k : (\!|S|\!)^1 \vdash [\![P']\!]^1 \triangleright \diamond$$
Let us write U_1 to stand for $?((\!|S_1|\!)^1 \multimap \diamond); \mathsf{end} \multimap \diamond$. The corresponding typing in the target language is as follows:
$$\dfrac{\dfrac{\dfrac{\dfrac{(\!|\Gamma|\!)^1; \{x : (\!|S_1|\!)^1 \multimap \diamond\}; \emptyset \vdash x \triangleright (\!|S_1|\!)^1 \multimap \diamond \qquad (\!|\Gamma|\!)^1; \emptyset; \{k' : (\!|S_1|\!)^1\} \vdash k' \triangleright (\!|S_1|\!)^1}{(\!|\Gamma|\!)^1; \{x : (\!|S_1|\!)^1 \multimap \diamond\}; k' : (\!|S_1|\!)^1 \vdash xk' \triangleright \diamond}}{(\!|\Gamma|\!)^1; \{x : (\!|S_1|\!)^1 \multimap \diamond\}; k' : (\!|S_1|\!)^1 \cdot z : \mathsf{end} \vdash xk' \triangleright \diamond}}{(\!|\Gamma|\!)^1; \emptyset; k' : (\!|S_1|\!)^1 \cdot z : ?((\!|S_1|\!)^1 \multimap \diamond); \mathsf{end} \vdash z?(x).(xk') \triangleright \diamond}}{(\!|\Gamma|\!)^1; \emptyset; k' : (\!|S_1|\!)^1 \vdash \lambda z.z?(x).(xk') \triangleright U_1} \tag{1}$$
$$\frac{(\!|\Gamma|\!)^1; \emptyset; (\!|\Delta|\!)^1 \cdot k : (\!|S|\!)^1 \vdash [\![P']\!]^1 \triangleright \diamond \qquad (\!|\Gamma|\!)^1; \emptyset; k' : (\!|S_1|\!)^1 \vdash \lambda z.z?(x).(xk') \triangleright U_1 \ (1)}{(\!|\Gamma|\!)^1; \emptyset; (\!|\Delta|\!)^1 \cdot k' : (\!|S_1|\!)^1 \cdot k : !\langle U_1 \rangle; (\!|S|\!)^1 \vdash k!\langle \lambda z.z?(x).(xk') \rangle.[\![P']\!]^1 \triangleright \diamond}$$
In the second sub-case, we have $n = a$ (output of a shared name). Then we have the following typing in the source language:
$$\frac{\Gamma \cdot a : \langle S_1 \rangle; \emptyset; \Delta \cdot k : S \vdash P' \triangleright \diamond \quad \Gamma \cdot a : \langle S_1 \rangle; \emptyset; \emptyset \vdash a \triangleright S_1}{\Gamma \cdot a : \langle S_1 \rangle; \emptyset; \Delta \cdot k : !\langle \langle S_1 \rangle \rangle; S \vdash k!\langle a \rangle.P' \triangleright \diamond}$$
The typing in the target language is derived similarly as in the first sub-case.

2. Case $P = k?(x).Q$. We have two sub-cases, depending on the type of x. In the first case, x stands for a linear channel. Then we have the following typing in the source language:
$$\frac{\Gamma; \emptyset; \Delta \cdot k : S \cdot x : S_1 \vdash Q \triangleright \diamond}{\Gamma; \emptyset; \Delta \cdot k : ?(S_1); S \vdash k?(x).Q \triangleright \diamond}$$

Thus, by IH we have

$$(\!|\Gamma|\!)^1;\emptyset;(\!|\Delta|\!)^1\cdot k:(\!|S|\!)^1\cdot x:(\!|S_1|\!)^1 \vdash [\![Q]\!]^1 \triangleright \diamond$$

Let us write U_1 to stand for $?((\!|S_1|\!)^1 \multimap \diamond);\mathsf{end}\multimap\diamond$. The corresponding typing in the target language is as follows:

$$\frac{(\!|\Gamma|\!)^1;\{X:U_1\};\emptyset \vdash X \triangleright U_1 \qquad (\!|\Gamma|\!)^1;\emptyset;\cdot s:?((\!|S_1|\!)^1\multimap\diamond);\mathsf{end} \vdash s \triangleright ?((\!|S_1|\!)^1\multimap\diamond);\mathsf{end}}{(\!|\Gamma|\!)^1;\{X:U_1\};\cdot s:?((\!|S_1|\!)^1\multimap\diamond);\mathsf{end} \vdash x\,s \triangleright \diamond} \tag{2}$$

$$\frac{\dfrac{(\!|\Gamma|\!)^1;\emptyset;\emptyset\vdash \mathbf{0}\triangleright\diamond}{(\!|\Gamma|\!)^1;\emptyset;\overline{s}:\mathsf{end}\vdash\mathbf{0}\triangleright\diamond} \qquad \dfrac{(\!|\Gamma|\!)^1;\emptyset;(\!|\Delta|\!)^1\cdot k:(\!|S|\!)^1 x:(\!|S_1|\!)^1 \vdash [\![Q]\!]^1\triangleright\diamond}{(\!|\Gamma|\!)^1;\emptyset;(\!|\Delta|\!)^1\cdot k:(\!|S|\!)^1\vdash \lambda x.[\![Q]\!]^1\triangleright(\!|S_1|\!)^1\multimap\diamond}}{(\!|\Gamma|\!)^1;\emptyset;(\!|\Delta|\!)^1\cdot k:(\!|S|\!)^1\cdot\overline{s}:!\langle(\!|S_1|\!)^1\multimap\diamond\rangle;\mathsf{end}\vdash\overline{s}!\langle\lambda x.[\![Q]\!]^1\rangle.\mathbf{0}\triangleright\diamond} \tag{3}$$

$$\frac{\begin{array}{c}(\!|\Gamma|\!)^1;\{X:U_1\};\cdot s:?((\!|S_1|\!)^1\multimap\diamond);\mathsf{end}\vdash x\,s\triangleright\diamond \quad (2)\\ (\!|\Gamma|\!)^1;\emptyset;(\!|\Delta|\!)^1\cdot k:(\!|S|\!)^1\cdot\overline{s}:!\langle(\!|S_1|\!)^1\multimap\diamond\rangle;\mathsf{end}\vdash\overline{s}!\langle\lambda x.[\![Q]\!]^1\rangle.\mathbf{0}\triangleright\diamond \quad (3)\end{array}}{(\!|\Gamma|\!)^1;\{X:U_1\};(\!|\Delta|\!)^1\cdot k:(\!|S|\!)^1\cdot s:?((\!|S_1|\!)^1\multimap\diamond);\mathsf{end}\cdot\overline{s}:!\langle(\!|S_1|\!)^1\multimap\diamond\rangle;\mathsf{end}\vdash x\,s\mid\overline{s}!\langle\lambda x.[\![Q]\!]^1\rangle.\mathbf{0}\triangleright\diamond} \tag{4}$$

$$\frac{\dfrac{(\!|\Gamma|\!)^1;\{X:U_1\};(\!|\Delta|\!)^1\cdot k:(\!|S|\!)^1\cdot s:?((\!|S_1|\!)^1\multimap\diamond);\mathsf{end}\cdot\overline{s}:!\langle(\!|S_1|\!)^1\multimap\diamond\rangle;\mathsf{end}\vdash x\,s\mid\overline{s}!\langle\lambda x.[\![Q]\!]^1\rangle.\mathbf{0}\triangleright\diamond \quad (4)}{(\!|\Gamma|\!)^1;\{X:U_1\};(\!|\Delta|\!)^1\cdot k:(\!|S|\!)^1\vdash(\nu\,s)(x\,s\mid\overline{s}!\langle\lambda x.[\![Q]\!]^1\rangle.\mathbf{0})\triangleright\diamond}}{(\!|\Gamma|\!)^1;\emptyset;(\!|\Delta|\!)^1\cdot k:?(U_1);(\!|S|\!)^1\vdash k?(x).(\nu\,s)(x\,s\mid\overline{s}!\langle\lambda x.[\![Q]\!]^1\rangle.\mathbf{0})\triangleright\diamond}$$

In the second sub-case, x stands for a shared name. Then we have the following typing in the source language:

$$\frac{\Gamma\cdot x:\langle S_1\rangle;\emptyset;\Delta\cdot k:S\vdash Q\triangleright\diamond}{\Gamma;\emptyset;\Delta\cdot k:?(\langle S_1\rangle);S\vdash k?(x).Q\triangleright\diamond}$$

The typing in the target language is derived similarly as in the first sub-case.

3. Case $P_0 = X$. Then we have the following typing in the source language:

$$\Gamma\cdot X:\Delta;\emptyset;\emptyset\vdash X\triangleright\diamond$$

Then the typing of $[\![X]\!]^1_f$ is as follows, assuming $f(X)=\tilde{n}$ and $\tilde{x}=\|\tilde{n}\|$. Also, we write $\Delta_{\tilde{n}}$ and $\Delta_{\tilde{x}}$ to stand for $n_1:S_1,\ldots,n_m:S_m$ and $x_1:S_1,\ldots,x_m:S_m$, respectively. Below, we assume that $\Gamma=\Gamma'\cdot X:\tilde{T}\to\diamond$, where

$$\tilde{T}=(\tilde{S},S^*) \qquad S^*=?(A);\mathsf{end} \qquad A=\mu\mathsf{t}.(\tilde{S},?(\mathsf{t});\mathsf{end})$$

$$\frac{\dfrac{}{\Gamma;\emptyset;\emptyset\vdash z_X\triangleright\tilde{T}\to\diamond} \qquad \begin{array}{c}\Gamma;\emptyset;\{n_i:S_i\}\vdash n_i\triangleright S_i\\ \Gamma;\emptyset;\{s:S^*\}\vdash s\triangleright S^*\end{array}}{\Gamma;\emptyset;\Delta_{\tilde{n}},s:?(\tilde{T}\to\diamond);\mathsf{end}\vdash z_X(\tilde{n},s)\triangleright\diamond} \tag{5}$$

$$\frac{\dfrac{\Gamma;\emptyset;\emptyset\vdash\mathbf{0}\triangleright\diamond}{\Gamma;\emptyset;\overline{s}:\mathsf{end}\vdash\mathbf{0}\triangleright\diamond} \qquad \dfrac{\begin{array}{c}\Gamma;\emptyset;\{x_i:S_i\}\vdash x_i\triangleright S_i\\ \Gamma;\emptyset;\{z:S^*\}\vdash z\triangleright S^*\\ \Gamma;\emptyset;\emptyset\vdash z_X\triangleright\tilde{T}\to\diamond\end{array}}{\Gamma;\emptyset;\Delta_{\tilde{x}},z:S^*\vdash z_X(\tilde{x},z)\triangleright\diamond}}{\Gamma;\emptyset;\emptyset\vdash\lambda(\tilde{x},z).\ z_X(\tilde{x},z)\triangleright\tilde{T}\to\diamond}}{\Gamma;\emptyset;\overline{s}:!\langle\tilde{T}\to\diamond\rangle;\mathsf{end}\vdash\overline{s}!\langle\lambda(\tilde{x},z).\ z_X(\tilde{x},z)\rangle.\mathbf{0}\triangleright\diamond} \tag{6}$$

$$\frac{\begin{array}{c}\Gamma;\emptyset;\Delta_{\tilde{n}},s:?(\tilde{T}\to\diamond);\mathsf{end}\vdash z_X(\tilde{n},s)\triangleright\diamond \quad (5)\\ \Gamma;\emptyset;\overline{s}:!\langle\tilde{T}\to\diamond\rangle;\mathsf{end}\vdash\overline{s}!\langle\lambda(\tilde{x},z).\ z_X(\tilde{x},z)\rangle.\mathbf{0}\triangleright\diamond \quad (6)\end{array}}{\dfrac{\Gamma;\emptyset;\Delta_{\tilde{n}},s:?(\tilde{T}\to\diamond);\mathsf{end},\overline{s}:!\langle\tilde{T}\to\diamond\rangle;\mathsf{end}\vdash z_X(\tilde{n},s)\mid\overline{s}!\langle\lambda(\tilde{x},z).\ x(\tilde{x},z)\rangle.\mathbf{0}\triangleright\diamond}{\Gamma;\emptyset;\Delta_{\tilde{n}}\vdash(\nu\,s)(z_X(\tilde{n},s)\mid\overline{s}!\langle\lambda(\tilde{x},z).\ z_X(\tilde{x},z)\rangle.\mathbf{0})\triangleright\diamond}}$$

4. Case $P_0=\mu X.P$. Then we have the following typing in the source language:

$$\frac{\Gamma\cdot X:\Delta;\emptyset;\Delta\vdash P\triangleright\diamond}{\Gamma;\emptyset;\Delta\vdash\mu X.P\triangleright\diamond}$$

Then we have the following typing in the target language —we write R to stand for $[\![P]\!]^1_{f,\{X\to\tilde{n}\}}$ and $\tilde{x}$ to stand for $\|\mathsf{ofn}(P)\|$.

$$\frac{\dfrac{(\!|\Gamma|\!)^1\cdot z_X:\tilde{T}\to\diamond;\emptyset;(\!|\Delta_{\tilde{n}}|\!)^1\vdash R\triangleright\diamond}{(\!|\Gamma|\!)^1\cdot z_X:\tilde{T}\to\diamond;\emptyset;(\!|\Delta_{\tilde{n}}|\!)^1,s:\mathsf{end}\vdash R\triangleright\diamond}}{(\!|\Gamma|\!)^1;\emptyset;(\!|\Delta_{\tilde{n}}|\!)^1,s:?(\tilde{T}\to\diamond);\mathsf{end}\vdash s?(z_X).R\triangleright\diamond} \tag{7}$$

$$\frac{\dfrac{(\!|\Gamma|\!)^1;\emptyset;\emptyset\vdash\mathbf{0}\triangleright\diamond}{(\!|\Gamma|\!)^1;\emptyset;\overline{s}:\mathsf{end}\vdash\mathbf{0}\triangleright\diamond} \qquad \dfrac{\dfrac{\dfrac{(\!|\Gamma|\!)^1\cdot z_X:\tilde{T}\to\diamond;\emptyset;(\!|\Delta_{\tilde{x}}|\!)^1\vdash\{\{R\}\}^{\emptyset}\triangleright\diamond}{(\!|\Gamma|\!)^1\cdot z_X:\tilde{T}\to\diamond;\emptyset;(\!|\Delta_{\tilde{x}}|\!)^1,y:\mathsf{end}\vdash\{\{R\}\}^{\emptyset}\triangleright\diamond}}{(\!|\Gamma|\!)^1;\emptyset;(\!|\Delta_{\tilde{x}}|\!)^1,y:?(A);\mathsf{end}\vdash y?(z_X).\{\{R\}\}^{\emptyset}\triangleright\diamond}}{(\!|\Gamma|\!)^1;\emptyset;\emptyset\vdash\lambda(\tilde{x},y).\ y?(z_X).\{\{R\}\}^{\emptyset}\triangleright\tilde{T}\to\diamond}}{(\!|\Gamma|\!)^1;\emptyset;\overline{s}:!\langle\tilde{T}\to\diamond\rangle;\mathsf{end}\vdash\overline{s}!\langle\lambda(\tilde{x},y).\ y?(z_X).\{\{R\}\}^{\emptyset}\rangle.\mathbf{0}\triangleright\diamond} \tag{8}$$

$$\frac{\dfrac{(\!|\Gamma|\!)^1;\emptyset;(\!|\Delta_{\tilde{n}}|\!)^1, s:?(\tilde{T}\to\diamond);\mathsf{end}\vdash s?(z_X).R\triangleright\diamond \quad (7) \qquad (\!|\Gamma|\!)^1;\emptyset;\overline{s}:!\langle\tilde{T}\to\diamond\rangle;\mathsf{end}\vdash \overline{s}!\langle\lambda(\tilde{x},y).\, y?(z_X).\{\!\{R\}\!\}^{\emptyset}\rangle.\mathbf{0}\triangleright\diamond \quad (8)}{(\!|\Gamma|\!)^1;\emptyset;(\!|\Delta_{\tilde{n}}|\!)^1, s:?(\tilde{T}\to\diamond);\mathsf{end},\overline{s}:!\langle\tilde{T}\to\diamond\rangle;\mathsf{end}\vdash s?(z_X).R\mid\overline{s}!\langle\lambda(\tilde{x},y).\, y?(z_X).\{\!\{R\}\!\}^{\emptyset}\rangle.\mathbf{0}\triangleright\diamond}}{(\!|\Gamma|\!)^1;\emptyset;(\!|\Delta_{\tilde{n}}|\!)^1\vdash(\nu\, s)(s?(z_X).R\mid\overline{s}!\langle\lambda(\tilde{x},y).\, y?(z_X).\{\!\{R\}\!\}^{\emptyset}\rangle.\mathbf{0})\triangleright\diamond}$$

□

Before we prove operational correspondence we define mapping from $\{\!|\cdot|\!\}^1 : \mathcal{A}\to\mathcal{A}$ where $\mathcal{A}$ is the set of labels of the relation $\overset{\ell}{\longmapsto}$:

Definition B.4 ($\{\!|\cdot|\!\}^1 : \mathcal{A}\to\mathcal{A}$). Let $\mathcal{A}$ is the set of labels of the relation $\overset{\ell}{\longmapsto}$ then we define:

$$\{\!|(\nu\tilde{m})n!\langle m\rangle|\!\}^1 \stackrel{\mathsf{def}}{=} (\nu\tilde{m})n!\langle\lambda z.\, z?(x).(xm)\rangle \qquad \{\!|n?\langle m\rangle|\!\}^1 \stackrel{\mathsf{def}}{=} n?\langle\lambda z.\, z?(x).(xm)\rangle \qquad \{\!|(\nu\tilde{m})n!\langle\lambda x.P\rangle|\!\}^1 \stackrel{\mathsf{def}}{=} (\nu\tilde{m})n!\langle\lambda x.[\![P]\!]^1_{\emptyset}\rangle$$
$$\{\!|n?\langle\lambda x.P\rangle|\!\}^1 \stackrel{\mathsf{def}}{=} n?\langle\lambda x.[\![P]\!]^1_{\emptyset}\rangle \qquad \{\!|n\oplus l|\!\}^1 \stackrel{\mathsf{def}}{=} n\oplus l \qquad \{\!|n\&l|\!\}^1 \stackrel{\mathsf{def}}{=} n\&l \qquad \{\!|\tau|\!\}^1 \stackrel{\mathsf{def}}{=} \tau$$

We now state and prove a detailed version of the operational corresponce in Def. B.1.

Proposition B.5 (Operational Correspondence, HOπ into HO). Let P be a HOπ process. If $\Gamma;\emptyset;\Delta\vdash P\triangleright\diamond$ then:

1. Suppose $\Gamma;\Delta\vdash P\overset{\ell_1}{\longmapsto}\Delta'\vdash P'$. Then we have:
   a) If $\ell_1\in\{(\nu\tilde{m})n!\langle m\rangle, (\nu\tilde{m})n!\langle\lambda x.Q\rangle, s\oplus l, s\&l\}$ then $\exists\ell_2$ s.t.
   $(\!|\Gamma|\!)^1;(\!|\Delta|\!)^1\vdash[\![P]\!]^1_f\overset{\ell_2}{\longmapsto}(\!|\Delta'|\!)^1\vdash[\![P']\!]^1_f$ and $\ell_2=\{\!|\ell_1|\!\}^1$.
   b) If $\ell_1=n?\langle\lambda y.Q\rangle$ and $P'=P_0\{^{\lambda y.Q}/_x\}$ then $\exists\ell_2$ s.t.
   $(\!|\Gamma|\!)^1;(\!|\Delta|\!)^1\vdash[\![P]\!]^1_f\overset{\ell_2}{\longmapsto}(\!|\Delta'|\!)^1\vdash[\![P_0]\!]^1_f\{^{\lambda y.[\![Q]\!]^1_{\emptyset}}/_x\}$ and $\ell_2=\{\!|\ell_1|\!\}^1$.
   c) If $\ell_1=n?\langle m\rangle$ and $P'=P_0\{^m/_x\}$ then $\exists\ell_2, R$ s.t.
   $(\!|\Gamma|\!)^1;(\!|\Delta|\!)^1\vdash[\![P]\!]^1_f\overset{\ell_2}{\longmapsto}(\!|\Delta'|\!)^1\vdash R$, with $\ell_2=\{\!|\ell_1|\!\}^1$,
   and $(\!|\Gamma|\!)^1;(\!|\Delta'|\!)^1\vdash R\overset{\tau_\beta}{\longmapsto}\overset{\tau_s}{\longmapsto}\overset{\tau_\beta}{\longmapsto}(\!|\Delta'|\!)^1\vdash[\![P_0]\!]^1_f\{^m/_x\}$.
   d) If $\ell_1=\tau$ and $P'\equiv(\nu\tilde{m})(P_1\mid P_2\{^m/_x\})$ then $\exists R$ s.t.
   $(\!|\Gamma|\!)^1;(\!|\Delta|\!)^1\vdash[\![P]\!]^1_f\overset{\tau}{\longmapsto}(\!|\Delta|\!)^1\vdash(\nu\tilde{m})([\![P_1]\!]^1_f\mid R)$, and
   $(\!|\Gamma|\!)^1;(\!|\Delta|\!)^1\vdash(\nu\tilde{m})([\![P_1]\!]^1_f\mid R)\overset{\tau_\beta}{\longmapsto}\overset{\tau_s}{\longmapsto}\overset{\tau_\beta}{\longmapsto}(\!|\Delta|\!)^1\vdash(\nu\tilde{m})([\![P_1]\!]^1_f\mid[\![P_2]\!]^1_f\{^m/_x\})$.
   e) If $\ell_1=\tau$ and $P'\equiv(\nu\tilde{m})(P_1\mid P_2\{^{\lambda y.Q}/_x\})$ then
   $(\!|\Gamma|\!)^1;(\!|\Delta|\!)^1\vdash[\![P]\!]^1_f\overset{\tau}{\longmapsto}(\!|\Delta_1|\!)^1\vdash(\nu\tilde{m})([\![P_1]\!]^1_f\mid[\![P_2]\!]^1_f\{^{\lambda y.[\![Q]\!]^1_{\emptyset}}/_x\})$.
   f) If $\ell_1=\tau$ and $P'\not\equiv(\nu\tilde{m})(P_1\mid P_2\{^m/_x\})\wedge P'\not\equiv(\nu\tilde{m})(P_1\mid P_2\{^{\lambda y.Q}/_x\})$ then
   $(\!|\Gamma|\!)^1;(\!|\Delta|\!)^1\vdash[\![P]\!]^1_f\overset{\tau}{\longmapsto}(\!|\Delta'_1|\!)^1\vdash[\![P']\!]^1_f$.
2. Suppose $(\!|\Gamma|\!)^1;(\!|\Delta|\!)^1\vdash[\![P]\!]^1_f\overset{\ell_2}{\longmapsto}(\!|\Delta'|\!)^1\vdash Q$. Then we have:
   a) If $\ell_2\in\{(\nu\tilde{m})n!\langle\lambda z.\, z?(x).(xm)\rangle, (\nu\tilde{m})n!\langle\lambda x.R\rangle, s\oplus l, s\&l\}$ then $\exists\ell_1, P'$ s.t.
   $\Gamma;\Delta\vdash P\overset{\ell_1}{\longmapsto}\Delta'\vdash P'$, $\ell_1=\{\!|\ell_2|\!\}^1$, and $Q=[\![P']\!]^1_f$.
   b) If $\ell_2=n?\langle\lambda y.R\rangle$ then either:
      (i) $\exists\ell_1, x, P', P''$ s.t.
      $\Gamma;\Delta\vdash P\overset{\ell_1}{\longmapsto}\Delta'\vdash P'\{^{\lambda y.P''}/_x\}$, $\ell_1=\{\!|\ell_2|\!\}^1$, $[\![P'']\!]^1_{\emptyset}=R$, and $Q=[\![P']\!]^1_f$.
      (ii) $R\equiv y?(x).(xm)$ and $\exists\ell_1, z, P'$ s.t.
      $\Gamma;\Delta\vdash P\overset{\ell_1}{\longmapsto}\Delta'\vdash P'\{^m/_z\}$, $\ell_1=\{\!|\ell_2|\!\}^1$, and
      $(\!|\Gamma|\!)^1;(\!|\Delta'|\!)^1\vdash Q\overset{\tau_\beta}{\longmapsto}\overset{\tau_s}{\longmapsto}\overset{\tau_\beta}{\longmapsto}(\!|\Delta''|\!)^1\vdash[\![P'\{^m/_z\}]\!]^1_f$
   c) If $\ell_2=\tau$ then $\Delta'=\Delta$ and either
      (i) $\exists P'$ s.t. $\Gamma;\Delta\vdash P\overset{\tau}{\longmapsto}\Delta\vdash P'$, and $Q=[\![P']\!]^1_f$.
      (ii) $\exists P_1, P_2, x, m, Q'$ s.t. $\Gamma;\Delta\vdash P\overset{\tau}{\longmapsto}\Delta\vdash(\nu\tilde{m})(P_1\mid P_2\{^m/_x\})$, and
      $(\!|\Gamma|\!)^1;(\!|\Delta|\!)^1\vdash Q\overset{\tau_\beta}{\longmapsto}\overset{\tau_s}{\longmapsto}\overset{\tau_\beta}{\longmapsto}(\!|\Delta|\!)^1\vdash[\![P_1]\!]^1_f\mid[\![P_2\{^m/_x\}]\!]^1_f$

Proof.

By transition induction. We consider parts (1) and (2) separately:

Part (1) - Completeness. We consider two representative cases, the rest is similar or simpler:

1. Subcase (a): $P=s!\langle n\rangle.P'$ and $\ell_1=s!\langle n\rangle$ (the case $\ell_1=(\nu n)s!\langle n\rangle$ is similar). By assumption, P is well-typed. We may have:

$$\frac{\Gamma;\emptyset;\Delta_0\cdot s:S_1\vdash P'\triangleright\diamond \quad \Gamma;\emptyset;\{n{:}S\}\vdash n\triangleright S}{\Gamma;\emptyset;\Delta_0\cdot n{:}S\cdot s:!\langle S\rangle;S_1\vdash s!\langle n\rangle.P'\triangleright\diamond}$$

for some S, S_1, Δ_0. We may then have the following transition:

$$\Gamma; \Delta_0 \cdot n{:}S \cdot s : !\langle S\rangle; S_1 \vdash s!\langle n\rangle.P' \overset{\ell_1}{\longmapsto} \Delta_0 \cdot s{:}S_1 \vdash P'$$

The encoding of the source judgment for P is as follows:

$$(\!|\Gamma|\!)^1; \emptyset; (\!|\Delta_0 \cdot n{:}S \cdot s : !\langle S\rangle; S_1|\!)^1 \vdash [\![s!\langle n\rangle.P']\!]^1 \triangleright \diamond$$

which, using Def. 7.3 can be expressed as

$$(\!|\Gamma|\!)^{\mathsf{p}}; \emptyset; (\!|\Delta_0|\!) \cdot n{:}(\!|S|\!)^1 \cdot s : !\langle ?((\!|S|\!)^1 \multimap \diamond); \mathsf{end} \multimap \diamond\rangle; (\!|S_1|\!)^1 \vdash s!\langle \lambda z.\, z?(x).(xn)\rangle.[\![P']\!]^1 \triangleright \diamond$$

Now, $\{\!|\ell_1|\!\}^1 = s!\langle \lambda z.\, z?(x).xn\rangle$. We may infer the following transition for $[\![P]\!]^1$:

$$\begin{array}{rl} & (\!|\Gamma|\!)^1; \emptyset; (\!|\Delta|\!)^1 \vdash s!\langle \lambda z.\, z?(x).(xn)\rangle.[\![P']\!]^1 \triangleright \diamond \\ \overset{\{\!|\ell_1|\!\}^1}{\longmapsto} & (\!|\Gamma|\!)^1; \emptyset; (\!|\Delta_0|\!)^1 \cdot s : (\!|S_1|\!)^1 \vdash [\![P']\!]^1 \triangleright \diamond \\ = & (\!|\Gamma|\!)^1; \emptyset; (\!|\Delta_0 \cdot s : S_1|\!)^1 \vdash [\![P']\!]^1 \triangleright \diamond \end{array}$$

from which the thesis follows easily.

2. Subcase (c): $P = n?(x).P'$ and $\ell_1 = n?\langle m\rangle$. By assumption P is well-typed. We may have:

$$\frac{\Gamma; \emptyset; \Delta_0 \cdot x : S \cdot n : S_1 \vdash P' \triangleright \diamond \quad \Gamma; \emptyset; \{x : S\} \vdash x \triangleright S}{\Gamma; \emptyset; \Delta_0 \cdot n : ?(S); S_1 \vdash n?(x).P' \triangleright \diamond}$$

for some S, S_1, Δ_0. We may infer the following typed transition:

$$\Gamma; \emptyset; \Delta_0 \cdot n : ?(S); S_1 \vdash n?(x).P' \triangleright \diamond \overset{n?\langle m\rangle}{\longmapsto} \Gamma; \emptyset; \Delta_0 \cdot n : S_1 \cdot m : S \vdash P'\{m/x\} \triangleright \diamond$$

The encoding of the source judgment for P is as follows:

$$\begin{array}{rl} & (\!|\Gamma|\!)^1; \emptyset; (\!|\Delta_0 \cdot n : ?(S); S_1|\!)^1 \vdash [\![P]\!]^1 \triangleright \diamond \\ = & (\!|\Gamma|\!)^1; \emptyset; (\!|\Delta_0|\!)^1 \cdot n : ?(?((\!|S|\!)^1 \multimap \diamond); \mathsf{end} \multimap \diamond); (\!|S_1|\!)^1 \vdash n?(x).(\nu\, s)((x\, s) \mid \overline{s}!\langle \lambda x.[\![P']\!]^1\rangle.\mathbf{0}) \triangleright \diamond \end{array}$$

Now, $\{\!|\ell_1|\!\}^1 = n?\langle \lambda z.\, z?(x).(xm)\rangle$ and it is immediate to infer the following transition for $[\![P]\!]^1$:

$$\begin{array}{rl} & (\!|\Gamma|\!)^1; \emptyset; (\!|\Delta_0|\!)^1 \cdot n : ?(?((\!|S|\!)^1 \multimap \diamond); \mathsf{end} \multimap \diamond); (\!|S_1|\!)^1 \vdash n?(x).(\nu\, s)((x\, s) \mid \overline{s}!\langle \lambda x.[\![P']\!]^1\rangle.\mathbf{0}) \triangleright \diamond \\ \overset{\{\!|\ell_1|\!\}^1}{\longmapsto} & (\!|\Gamma|\!)^1; \emptyset; (\!|\Delta_0|\!)^1 \cdot n : (\!|S_1|\!)^1 \cdot m : (\!|S|\!)^1 \vdash (\nu\, s)((x\, s) \mid \overline{s}!\langle \lambda x.[\![P']\!]^1\rangle.\mathbf{0})\{^{\lambda z.\, z?(x).(xm)}/_x\} \triangleright \diamond \end{array}$$

Let us write R to stand for process $(\nu\, s)((x\, s) \mid \overline{s}!\langle \lambda x.[\![P']\!]^1\rangle.\mathbf{0})\{^{\lambda z.\, z?(x).(xm)}/_x\}$. We then have:

$$\begin{array}{rcl} R & \overset{\tau}{\longrightarrow} & (\nu\, s)(s?(x).(xm) \mid \overline{s}!\langle \lambda x.[\![P']\!]^1\rangle.\mathbf{0}) \\ & \overset{\tau}{\longrightarrow} & (\lambda x.[\![P']\!]^1)\, m \mid \mathbf{0} \\ & \overset{\tau}{\longrightarrow} & [\![P']\!]^1\{^m/_x\} \end{array}$$

and so the thesis follows.

Part (2) - Soundness. We consider two representative cases, the rest is similar or simpler:

1. Subcase (a): $P = n!\langle m\rangle.P'$ and $\ell_2 = n!\langle \lambda z.\, z?(x).(xm)\rangle$ (the case $\ell_2 = (\nu m)n!\langle \lambda z.\, z?(x).(xm)\rangle$ is similar). Then we have:

$$(\!|\Gamma|\!)^1; \emptyset; (\!|\Delta_0|\!)^1 \cdot n : !\langle ?((\!|S|\!)^1 \multimap \diamond); \mathsf{end} \multimap \diamond\rangle; (\!|S_1|\!)^1 \vdash n!\langle \lambda z.\, z?(x).(xm)\rangle.[\![P']\!]^1 \triangleright \diamond$$

for some S, S_1, and Δ_0. We may infer the following typed transition for $[\![P]\!]^1$:

$$\begin{array}{rl} & (\!|\Gamma|\!)^1; (\!|\Delta_0|\!)^1 \cdot n : !\langle ?((\!|S|\!)^1 \multimap \diamond); \mathsf{end} \multimap \diamond\rangle; (\!|S_1|\!)^1 \vdash n!\langle \lambda z.\, z?(x).(xm)\rangle.[\![P']\!]^1 \\ \overset{\ell_2}{\longmapsto} & (\!|\Gamma|\!)^1; (\!|\Delta_0|\!)^1 \cdot n : (\!|S_1|\!)^1 \vdash [\![P']\!]^1 \end{array}$$

Now, in the source term P we can infer the following transition

$$\Gamma; \Delta_0 \cdot n : !\langle S\rangle; S_1 \vdash n!\langle m\rangle.P' \overset{n!\langle m\rangle}{\longmapsto} \Gamma; \Delta_0 \cdot n : S_1 \vdash P'$$

and thus the thesis follows easily by noticing that $\{\!|n!\langle m\rangle|\!\}^1 = n!\langle \lambda z.\, z?(x).(xm)\rangle$.

2. Subcase (c): $P = n?(x).P'$ and $\ell_2 = n?\langle \lambda y.\, y?(x).(xm)\rangle$. Then we have

$$(\!|\Gamma|\!)^1; \emptyset; (\!|\Delta_0|\!)^1 \cdot n : ?(?((\!|S|\!)^1 \multimap \diamond); \mathsf{end} \multimap \diamond); (\!|S_1|\!)^1 \vdash n?(x).(\nu\, s)((x\, s) \mid \overline{s}!\langle \lambda x.[\![P']\!]^1\rangle.\mathbf{0}) \triangleright \diamond$$

for some S, S_1, Δ_0. We may infer the following typed transitions for $[\![P]\!]^1$:

$$\begin{array}{rl} & (\!|\Gamma|\!)^1; (\!|\Delta_0|\!)^1 \cdot n : ?(?((\!|S|\!)^1 \multimap \diamond); \mathsf{end} \multimap \diamond); (\!|S_1|\!)^1 \vdash n?(x).(\nu s)((x\,s) \mid \overline{s}!\langle \lambda x.[\![P']\!]^1\rangle.\mathbf{0}) \\ \xrightarrow{\ell_2} & (\!|\Gamma|\!)^1; (\!|\Delta_0|\!)^1 \cdot n : (\!|S_1|\!)^1 \cdot m : (\!|S_1|\!)^1 \vdash (\nu s)((x\,s) \mid \overline{s}!\langle \lambda x.[\![P']\!]^1\rangle.\mathbf{0})\{^{\lambda z.z?(x).xm}/_{x}\} \\ = & (\!|\Gamma|\!)^1; (\!|\Delta_0|\!)^1 \cdot n : (\!|S_1|\!)^1 \cdot m : (\!|S|\!)^1 \vdash (\nu s)(s?(x).(xm) \mid \overline{s}!\langle \lambda x.[\![P']\!]^1\rangle.\mathbf{0}) \\ \xrightarrow{\tau} & (\!|\Gamma|\!)^1; (\!|\Delta_0|\!)^1 \cdot n : (\!|S_1|\!)^1 \cdot m : (\!|S|\!)^1 \vdash (\lambda x.[\![P']\!]^1)m \\ \xrightarrow{\tau} & (\!|\Gamma|\!)^1; (\!|\Delta_0|\!)^1 \cdot n : (\!|S_1|\!)^1 \cdot m : (\!|S|\!)^1 \vdash [\![P']\!]^1\{^{m}/_{x}\} \end{array}$$

Now, in the source term P we can infer the following transition

$$\Gamma; \Delta_0 \cdot n : ?(S); S_1 \vdash n?(x).P' \overset{n?\langle m\rangle}{\longmapsto} \Gamma; \Delta_0 \cdot n : S_1 \cdot m : S \vdash P'\{^{m}/_{x}\}$$

and the thesis follows.

□

Proposition B.6 (Full Abstraction, HOπ into HO). $\Gamma; \Delta_1 \vdash P_1 \approx \Delta_2 \vdash Q_1$ if and only if $(\!|\Gamma|\!)^1; (\!|\Delta_1|\!)^1 \vdash [\![P_1]\!]^1_f \approx (\!|\Delta_2|\!)^1 \vdash [\![Q_2]\!]^1_f$.

Proof. **Proof of Soundness Direction.**
Let

$$\mathfrak{R} = \{\Gamma; \Delta_1 \vdash P_1 \approx \Delta_2 \vdash Q_1 \;\mid\; (\!|\Gamma|\!)^1; (\!|\Delta_1|\!)^1 \vdash [\![P_1]\!]^1_f \approx (\!|\Delta_2|\!)^1 \vdash [\![Q_1]\!]^1_f\}$$

The proof considers a case analysis on the transition $\overset{\ell}{\longmapsto}$ and uses the soundness direction of operational correspondence (cf. Prop. B.5). We give an interesting case. The others are similar of easier.
- Case: $\ell = (\nu \tilde{m_1}')n!\langle m_1\rangle$.
Prop. B.5 implies that

$$\Gamma; \Delta_1 \vdash P_1 \overset{(\nu \tilde{m_1}')n!\langle m_1\rangle}{\longmapsto} \Delta_1' \vdash P_2$$

implies

$$(\!|\Gamma|\!)^1; (\!|\Delta_1|\!)^1 \vdash [\![P_1]\!]^1_f \overset{(\nu \tilde{m_1}')n!\langle \lambda z.z?(x).(xm_1)\rangle}{\longmapsto} (\!|\Delta_1'|\!)^1 \vdash [\![P_2]\!]^1_f$$

that in combination with the definition of $\mathfrak{R}$ we get

$$(\!|\Gamma|\!)^1; (\!|\Delta_2|\!)^1 \vdash [\![Q_1]\!]^1_f \overset{(\nu \tilde{m_2}')n!\langle \lambda z.z?(x).(xm_2)\rangle}{\Longmapsto} (\!|\Delta_2'|\!)^1 \vdash [\![Q_2]\!]^1_f \tag{9}$$

and

$$\begin{array}{rcll} (\!|\Gamma|\!)^1; \emptyset; (\!|\Delta_1'|\!)^1 & \approx & (\!|\Delta_2'|\!)^1 \vdash & (\nu \tilde{m_1}')([\![P_2]\!]^1_f \mid t?(x).(\nu s)(x\,s \mid \overline{s}!\langle x\rangle.\mathbf{0})s\lambda z.z?(x).(xm_1)) \\ & \approx & & (\nu \tilde{m_2}')([\![Q_2]\!]^1_f \mid t?(x).(\nu s)(x\,s \mid \overline{s}!\langle x\rangle.\mathbf{0})s\lambda z.z?(x).(xm_2)) \end{array}$$

We rewrite the last result as

$$\begin{array}{rcll} (\!|\Gamma|\!)^1; \emptyset; (\!|\Delta_1'|\!)^1 & \approx & (\!|\Delta_2'|\!)^1 \vdash & [\![(\nu \tilde{m_1}')(P_2 \mid t?(x).(\nu s)(x\,s \mid \overline{s}!\langle x\rangle.\mathbf{0})sm_1)]\!]^1_f \\ & \approx & & [\![(\nu \tilde{m_2}')(Q_2 \mid t?(x).(\nu s)(x\,s \mid \overline{s}!\langle x\rangle.\mathbf{0})sm_2)]\!]^1_f \end{array}$$

to conclude that

$$\begin{array}{rcll} \Gamma; \emptyset; \Delta_1' & \mathfrak{R} & \Delta_2' \vdash & (\nu \tilde{m_1}')(P_2 \mid t?(x).(\nu s)(x\,s \mid \overline{s}!\langle x\rangle.\mathbf{0})sm_1) \\ & \mathfrak{R} & & (\nu \tilde{m_2}')(Q_2 \mid t?(x).(\nu s)(x\,s \mid \overline{s}!\langle x\rangle.\mathbf{0})sm_2) \end{array}$$

as required
Proof of Completeness Direction.
Let

$$\mathfrak{R} = \{(\!|\Gamma|\!)^1; (\!|\Delta_1|\!)^1 \vdash [\![P_1]\!]^1_f, (\!|\Delta_2|\!)^1 \vdash [\![Q_1]\!]^1_f \;\mid\; \Gamma; \Delta_1 \vdash P_1 \approx \Delta_2 \vdash Q_1\}$$

We show that $\mathfrak{R} \subset \approx$ by a case analysis on the action ℓ
- Case: $\ell \notin \{(\nu \tilde{m})n!\langle \lambda x.P\rangle, n?\langle \lambda x.P\rangle\}$.
The proof of Prop. B.5 implies that

$$(\!|\Gamma|\!)^1; (\!|\Delta_1|\!)^1 \vdash [\![P_1]\!]^1_f \overset{\ell}{\longmapsto} (\!|\Delta_1'|\!)^1 \vdash [\![P_2]\!]^1_f$$

implies

$$\Gamma; \Delta_1 \vdash P_1 \overset{\ell}{\longmapsto} \Delta_1' \vdash P_2$$

From the latter transition and the definition of $\mathfrak{R}$ we imply

$$\Gamma; \Delta_2 \vdash Q_1 \overset{\ell}{\Longmapsto} \Delta_2' \vdash Q_2 \tag{10}$$

$$\Gamma; \Delta_1' \vdash P_2 \approx \Delta_2' \vdash Q_2 \tag{11}$$

From 10 and Prop. B.5 we get

$$(\!|\Gamma|\!)^1; (\!|\Delta_2|\!)^1 \vdash [\![Q_1]\!]^1_f \overset{\ell}{\Longmapsto} (\!|\Delta_2'|\!)^1 \vdash [\![Q_2]\!]^1_f$$

Furthermore, from 11 and the definition of $\mathfrak{R}$ we get

$$(\!|\Gamma|\!)^1;(\!|\Delta_1'|\!)^1 \vdash [\![P_2]\!]_f^1 \;\mathfrak{R}\; (\!|\Delta_2'|\!)^1 \vdash [\![Q_2]\!]_f^1$$

as required.
- Case: $\ell = (\nu\tilde{m})n!\langle\lambda x.P\rangle$
There are two subcases:
-Subcase:
The proof of Prop. B.5 implies that

$$(\!|\Gamma|\!)^1;(\!|\Delta_1|\!)^1 \vdash [\![P_1]\!]_f^1 \stackrel{\ell}{\longmapsto} (\!|\Delta_1'|\!)^1 \vdash [\![P_2]\!]_f^1$$

implies

$$\Gamma;\Delta_1 \vdash P_1 \stackrel{\ell}{\longmapsto} \Delta_1' \vdash P_2$$

where the proof is similar with the previous case.
- Subcase:
The proof of Prop. B.5 implies that

$$(\!|\Gamma|\!)^1;(\!|\Delta_1|\!)^1 \vdash [\![P_1]\!]_f^1 \stackrel{(\nu\tilde{m_1}')n!\langle\lambda z.z?(x).(x m_1)\rangle}{\longmapsto} (\!|\Delta_1'|\!)^1 \vdash [\![P_2]\!]_f^1$$

implies

$$\Gamma;\Delta_1 \vdash P_1 \stackrel{(\nu\tilde{m_1}')n!\langle m_1\rangle}{\longmapsto} \Delta_1' \vdash P_2$$

From the latter transition and the definition of $\mathfrak{R}$ we imply

$$\Gamma;\Delta_2 \vdash Q_1 \stackrel{(\nu\tilde{m_2}')n!\langle m_2\rangle}{\Longmapsto} \Delta_2' \vdash Q_2 \tag{12}$$

and

$$\begin{array}{rl} \Gamma;\emptyset;\Delta_1' & \vdash (\nu\tilde{m_1}')(P_2 \mid t?(x).(\nu s)(x\,s \mid \overline{s}!\langle x\rangle.\mathbf{0})sm_1) \\ \approx & \Delta_2' \vdash (\nu\tilde{m_2}')(Q_2 \mid t?(x).(\nu s)(x\,s \mid \overline{s}!\langle x\rangle.\mathbf{0})sm_2) \end{array} \tag{13}$$

From (12) and Prop. B.5 we get

$$(\!|\Gamma|\!)^1;(\!|\Delta_2|\!)^1 \vdash [\![Q_1]\!]_f^1 \stackrel{(\nu\tilde{m_2}')n!\langle\lambda z.z?(x).(x m_2)\rangle}{\Longmapsto} (\!|\Delta_2'|\!)^1 \vdash [\![Q_2]\!]_f^1$$

Furthermore, from (13) and the definition of $\mathfrak{R}$ we get

$$\begin{array}{lcll} (\!|\Gamma|\!)^1;\emptyset;(\!|\Delta_1'|\!)^1 & \mathfrak{R} & (\!|\Delta_2'|\!)^1 \vdash & [\![(\nu\tilde{m_1}')(P_2 \mid t?(x).(\nu s)(x\,s \mid \overline{s}!\langle x\rangle.\mathbf{0})sm_1)]\!]_f^1 \\ & \mathfrak{R} & & [\![(\nu\tilde{m_2}')(Q_2 \mid t?(x).(\nu s)(x\,s \mid \overline{s}!\langle x\rangle.\mathbf{0})sm_2)]\!]_f^1 \end{array}$$

as required.
- Case: $\ell = n?\langle\lambda x.P\rangle$
We have two subcases.
- Subcase: Similar with the first subcase of the previous case.
- Subcase: The proof of Prop. B.5 implies that

$$(\!|\Gamma|\!)^1;(\!|\Delta_1|\!)^1 \vdash [\![P_1]\!]_f^1 \stackrel{n?\langle\lambda z.z?(x).(x s)\rangle}{\longmapsto} (\!|\Delta_1''|\!)^1 \vdash R$$

implies

$$\Gamma;\Delta_1 \vdash P_1 \stackrel{n?\langle m_1\rangle}{\longmapsto} \Delta_1' \vdash P_2 \tag{14}$$

and

$$(\!|\Gamma|\!)^1;(\!|\Delta_1''|\!)^1 \vdash R \stackrel{\tau_s}{\longmapsto} (\!|\Delta_1'|\!)^1 \vdash [\![P_2]\!]_f^1 \tag{15}$$

From the transition (14) and the definition of $\mathfrak{R}$ we imply

$$\Gamma;\Delta_2 \vdash Q_1 \stackrel{n?\langle m_2\rangle}{\Longmapsto} \Delta_2' \vdash Q_2 \tag{16}$$

$$\Gamma;\Delta_1' \vdash P_2 \approx \Delta_2' \vdash Q_2 \tag{17}$$

From (16) and Prop. B.5 we get

$$(\!|\Gamma|\!)^1;(\!|\Delta_2|\!)^1 \vdash [\![Q_1]\!]_f^1 \stackrel{n?\langle\lambda z.z?(x).(x s)\rangle}{\Longmapsto} (\!|\Delta_2'|\!)^1 \vdash [\![Q_2]\!]_f^1$$

Furthermore, from 17 and the definition of $\mathfrak{R}$ we get

$$(\!|\Gamma|\!)^1;(\!|\Delta_1'|\!)^1 \vdash [\![P_2]\!]_f^1 \;\mathfrak{R}\; (\!|\Delta_2'|\!)^1 \vdash [\![Q_2]\!]_f^1$$

If we consider result (15) we get:

$$(\!|\Gamma|\!)^1;(\!|\Delta_1''|\!)^1 \vdash R \stackrel{\tau_s}{\longmapsto} \mathfrak{R}\; (\!|\Delta_2'|\!)^1 \vdash [\![Q_2]\!]_f^1$$

where following Lem. **??** we show that R is a bisimulation an up to $\stackrel{\tau_s}{\Longmapsto}$. □

B.2 Properties for encoding $\mathcal{L}_{\mathsf{HO}\pi}$ into $\mathcal{L}_\pi$

In this section we prove Thm. 7.6, in Page 4 that requires that encoding $\mathcal{L}_{\mathsf{HO}\pi}$ into $\mathcal{L}_\pi$ is precise. A precise encoding requires to prove three independent results:

- Type preservation, stated in Prop. B.7.
- Operational Correspondence, stated in Prop. B.9. Note that we prove a stronger operational correspondence condition, as in Def. B.1, than the condition suggested in Def. 6.4(2).
- Full Abstraction, stated in Prop. B.10.

Proposition B.7 (Type Preservation, $\mathsf{HO}\pi$ into π). Let P be a $\mathsf{HO}\pi$ process.
If $\Gamma;\emptyset;\Delta \vdash P \triangleright \diamond$ then $(\!|\Gamma|\!)^2;\emptyset;(\!|\Delta|\!)^2 \vdash [\![P]\!]^2 \triangleright \diamond$.

Proof. By induction on the inference $\Gamma;\emptyset;\Delta \vdash P \triangleright \diamond$.

1. Case $P = k!\langle \lambda x.\, Q\rangle.P$. Then we have two possibilities, depending on the typing for $\lambda x.\, Q$. The first case concerns a linear typing, and we have the following typing in the source language:

$$\frac{\Gamma;\emptyset;\Delta_1 \cdot k : S \vdash P \triangleright \diamond \quad \dfrac{\Gamma;\emptyset;\Delta_2 \cdot x : S_1 \vdash Q \triangleright \diamond}{\Gamma;\emptyset;\Delta_2 \vdash \lambda x.\, Q \triangleright S_1 \multimap \diamond}}{\Gamma;\emptyset;\Delta_1 \cdot \Delta_2 \cdot k : !\langle S_1 \multimap \diamond\rangle; S \vdash k!\langle \lambda x.\, Q\rangle.P \triangleright \diamond}$$

   This way, by IH we have

$$(\!|\Gamma|\!)^2;\emptyset;(\!|\Delta_2|\!)^2, x : (\!|S_1|\!)^2 \vdash [\![Q]\!]^2 \triangleright \diamond$$

   Let us write U_1 to stand for $\langle ?((\!|S_1|\!)^2); \mathsf{end}\rangle$. The corresponding typing in the target language is as follows:

$$\begin{array}{rcl}(\!|\Gamma_1|\!)^2 & = & (\!|\Gamma|\!)^2 \cup a : \langle ?((\!|S_1|\!)^2); \mathsf{end}\rangle \\ (\!|\Gamma_2|\!)^2 & = & (\!|\Gamma_1|\!)^2 \cup X : (\!|\Delta_2|\!)^2\end{array}$$

   Also $(*)$ stands for $(\!|\Gamma_1|\!)^2;\emptyset;\emptyset \vdash a \triangleright U_1$; $(**)$ stands for $(\!|\Gamma_2|\!)^2;\emptyset;\emptyset \vdash a \triangleright U_1$; and $(***)$ stands for $(\!|\Gamma_2|\!)^2;\emptyset;\emptyset \vdash X \triangleright \diamond$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{(\!|\Gamma_2|\!)^2;\emptyset;(\!|\Delta_2|\!)^2, x : (\!|S_1|\!)^2 \vdash [\![Q]\!]^2 \triangleright \diamond}}{(\!|\Gamma_2|\!)^2;\emptyset;(\!|\Delta_2|\!)^2, y : \mathsf{end}, x : (\!|S_1|\!)^2 \vdash [\![Q]\!]^2 \triangleright \diamond}}{(\!|\Gamma_2|\!)^2;\emptyset;(\!|\Delta_2|\!)^2, y : ?((\!|S_1|\!)^2); \mathsf{end} \vdash y?(x).[\![Q]\!]^2 \triangleright \diamond} \quad \dfrac{}{(**)}}{(***) \qquad (\!|\Gamma_2|\!)^2;\emptyset;(\!|\Delta_2|\!)^2 \vdash a?(y).y?(x).[\![Q]\!]^2 \triangleright \diamond}}{(\!|\Gamma_2|\!)^2;\emptyset;(\!|\Delta_2|\!)^2 \vdash a?(y).y?(x).[\![Q]\!]^2 \mid X \triangleright \diamond}}{(\!|\Gamma_1|\!)^2;\emptyset;(\!|\Delta_2|\!)^2 \vdash \mu X.(a?(y).y?(x).[\![Q]\!]^2 \mid X) \triangleright \diamond} \tag{18}$$

$$\dfrac{\begin{array}{c}(\!|\Gamma_1|\!)^2;\emptyset;(\!|\Delta_1|\!)^2, k : (\!|S|\!)^2 \vdash [\![P]\!]^2 \triangleright \diamond \\ (\!|\Gamma_1|\!)^2;\emptyset;(\!|\Delta_2|\!)^2 \vdash \mu X.(a?(y).y?(x).[\![Q]\!]^2 \mid X) \triangleright \diamond \quad (18)\end{array}}{(\!|\Gamma_1|\!)^2;\emptyset;(\!|\Delta_1, \Delta_2|\!)^2, k : (\!|S|\!)^2 \vdash [\![P]\!]^2 \mid \mu X.(a?(y).y?(x).[\![Q]\!]^2 \mid X) \triangleright \diamond} \tag{19}$$

$$\dfrac{\begin{array}{c}(\!|\Gamma_1|\!)^2;\emptyset;\emptyset \vdash a \triangleright U_1 \\ (\!|\Gamma_1|\!)^2;\emptyset;(\!|\Delta_1, \Delta_2|\!)^2, k : (\!|S|\!)^2 \vdash [\![P]\!]^2 \mid \mu X.(a?(y).y?(x).[\![Q]\!]^2 \mid X) \triangleright \diamond \quad (19)\end{array}}{\dfrac{(\!|\Gamma_1|\!)^2;\emptyset;(\!|\Delta_1, \Delta_2|\!)^2, k : !\langle U_1\rangle; (\!|S|\!)^2 \vdash k!\langle a\rangle.([\![P]\!]^2 \mid \mu X.(a?(y).y?(x).[\![Q]\!]^2 \mid X)) \triangleright \diamond}{(\!|\Gamma|\!)^2;\emptyset;(\!|\Delta_1, \Delta_2|\!)^2, k : !\langle U_1\rangle; (\!|S|\!)^2 \vdash (\nu a)(k!\langle a\rangle.([\![P]\!]^2 \mid \mu X.(a?(y).y?(x).[\![Q]\!]^2 \mid X))) \triangleright \diamond}$$

   In the second case, $\lambda x.\, Q$ has a shared type. We have the following typing in the source language:

$$\frac{\Gamma;\emptyset;\Delta \cdot k : S \vdash P \triangleright \diamond \quad \dfrac{\dfrac{\Gamma;\emptyset;\cdot x : S_1 \vdash Q \triangleright \diamond}{\Gamma;\emptyset;\emptyset \vdash \lambda x.\, Q \triangleright S_1 \multimap \diamond}}{\Gamma;\emptyset;\emptyset \vdash \lambda x.\, Q \triangleright S_1 \to \diamond}}{\Gamma;\emptyset;\Delta \cdot k : !\langle S_1 \to \diamond\rangle; S \vdash k!\langle \lambda x.\, Q\rangle.P \triangleright \diamond}$$

   The corresponding typing in the target language can be derived similarly as in the first case.

2. Case $P = k?(x).P$. Then there are two cases, depending on the type of X. In the first case, we have the following typing in the source language:

$$\frac{\Gamma \cdot x : S_1 \to \diamond;\emptyset;\Delta \cdot k : S \vdash P \triangleright \diamond}{\Gamma;\emptyset;\Delta \cdot k : ?(S_1 \to \diamond); S \vdash k?(x).P \triangleright \diamond}$$

   The corresponding typing in the target language is as follows:

$$\frac{\dfrac{}{(\!|\Gamma|\!)^2 \cdot x : \langle ?((\!|S_1|\!)^2); \mathsf{end}\rangle;\emptyset;\Delta \cdot k : (\!|S|\!)^2 \vdash (\!|P|\!)^2 \triangleright \diamond}}{(\!|\Gamma|\!)^2;\emptyset;(\!|\Delta|\!)^2 \cdot k : ?(\langle ?((\!|S_1|\!)^2); \mathsf{end}\rangle); (\!|S|\!)^2 \vdash k?(x).[\![P]\!]^2 \triangleright \diamond}$$

   In the second case, we have the following typing in the source language:

$$\frac{\Gamma;\{x : S_1 \multimap \diamond\};\emptyset;\Delta \cdot k : S \vdash P \triangleright \diamond}{\Gamma;\emptyset;\Delta \cdot k : ?(S_1 \multimap \diamond); S \vdash k?(x).P \triangleright \diamond}$$

The corresponding typing in the target language is as follows:

$$\frac{(\!|\Gamma|\!)^2 \cdot x : \langle ?((\!|S_1|\!)^2); \mathsf{end}\rangle; \emptyset; \Delta \cdot k : (\!|S|\!)^2 \vdash (\!|P|\!)^2 \triangleright \diamond}{(\!|\Gamma|\!)^2; \emptyset; (\!|\Delta|\!)^2 \cdot k : ?(\langle ?((\!|S_1|\!)^2); \mathsf{end}\rangle); (\!|S|\!)^2 \vdash k?(x).[\![P]\!]^2 \triangleright \diamond}$$

3. Case $P = xk$. Also here we have two cases, depending on whether X has linear or shared type. In the first case, x is linear and we have the following typing in the source language:

$$\frac{\Gamma; \{x : S_1 \multimap \diamond\}; \emptyset \vdash X \triangleright S_1 \multimap \diamond \quad \Gamma; \emptyset; \{k : S_1\} \vdash k \triangleright S_1}{\Gamma; \{x : S_1 \multimap \diamond\}; k : S_1 \vdash xk \triangleright \diamond}$$

Let us write $(\!|\Gamma_1|\!)^2$ to stand for $(\!|\Gamma|\!)^2 \cdot x : \langle !\langle (\!|S_1|\!)^2\rangle; \mathsf{end}\rangle$. The corresponding typing in the target language is as follows:

$$\frac{\dfrac{(\!|\Gamma_1|\!)^2; \emptyset; \emptyset \vdash \mathbf{0} \triangleright \diamond}{(\!|\Gamma_1|\!)^2; \emptyset; \overline{s} : \mathsf{end} \vdash \mathbf{0} \triangleright \diamond} \quad (\!|\Gamma_1|\!)^2; \emptyset; \{k : (\!|S_1|\!)^2\} \vdash k \triangleright (\!|S_1|\!)^2}{(\!|\Gamma_1|\!)^2; \emptyset;\ k : (\!|S_1|\!)^2, \overline{s} : !\langle (\!|S_1|\!)^2\rangle; \mathsf{end} \vdash \overline{s}!\langle k\rangle.\mathbf{0} \triangleright \diamond} \tag{20}$$

$$\frac{\dfrac{(\!|\Gamma_1|\!)^2; \emptyset;\ k : (\!|S_1|\!)^2, \overline{s} : !\langle (\!|S_1|\!)^2\rangle; \mathsf{end} \vdash \overline{s}!\langle k\rangle.\mathbf{0} \triangleright \diamond \quad (20) \qquad (\!|\Gamma_1|\!)^2; \emptyset; \emptyset \vdash x \triangleright \langle !\langle (\!|S_1|\!)^2\rangle; \mathsf{end}\rangle}{(\!|\Gamma_1|\!)^2; \emptyset; k : (\!|S_1|\!)^2, s : ?((\!|S_1|\!)^2); \mathsf{end}, \overline{s} : !\langle (\!|S_1|\!)^2\rangle; \mathsf{end} \vdash x!\langle s\rangle.\overline{s}!\langle k\rangle.\mathbf{0} \triangleright \diamond}}{(\!|\Gamma_1|\!)^2; \emptyset; k : (\!|S_1|\!)^2 \vdash (\nu\, s)(x!\langle s\rangle.\overline{s}!\langle k\rangle.\mathbf{0}) \triangleright \diamond}$$

In the second case, x is shared, and we have the following typing in the source language:

$$\frac{\Gamma \cdot x : S_1 \multimap \diamond; \emptyset; \emptyset \vdash x \triangleright S_1 \to \diamond \quad \Gamma; \emptyset; k : S_1 \vdash k \triangleright S_1}{\Gamma \cdot x : S_1 \to \diamond; \emptyset; k : S_1 \vdash xk \triangleright \diamond}$$

The associated typing in the target language is obtained similarly as in the first case. □

Before we prove operational correspondence we define mapping from $\{\!|\cdot|\!\}^2 : \mathcal{A} \to \mathcal{A}$ where $\mathcal{A}$ is the set of labels of the relation $\overset{\ell}{\longmapsto}$:

Definition B.8 ($\{\!|\cdot|\!\}^2 : \mathcal{A} \to \mathcal{A}$). Let $\mathcal{A}$ is the set of labels of the relation $\overset{\ell}{\longmapsto}$ then we define:

$$\{\!|(\nu\tilde{m})n!\langle\lambda x.P\rangle|\!\}^2 \stackrel{\mathsf{def}}{=} (\nu m)n!\langle m\rangle \qquad\qquad \{\!|n?\langle\lambda x.P\rangle|\!\}^2 \stackrel{\mathsf{def}}{=} n?\langle m\rangle \qquad m \text{ fresh}$$

and homomorphic for all other cases of $\ell \in \mathcal{A}$.

We now state and prove a detailed and extended version of the operational corresponce in Def. B.1.

Proposition B.9 (Operational Correspondence, HOπ into π). *Let P be an* HOπ *process such that* $\Gamma; \emptyset; \Delta \vdash P \triangleright \diamond$.

1. *Suppose* $\Gamma; \Delta \vdash P \overset{\ell_1}{\longmapsto} \Delta' \vdash P'$. *Then we have:*
   a) *If* $\ell_1 = (\nu\tilde{m})n!\langle\lambda x.Q\rangle$, *then* $\exists \Gamma', \Delta'', R$ *where either:*
      - $(\!|\Gamma|\!)^2; (\!|\Delta|\!)^2 \vdash [\![P]\!]^2 \overset{\{\!|\ell_1|\!\}^2}{\longmapsto} \Gamma' \cdot (\!|\Gamma|\!)^2; (\!|\Delta'|\!)^2 \vdash [\![P']\!]^2 \mid * a?(y).y?(x).[\![Q]\!]^2$
      - $(\!|\Gamma|\!)^2; (\!|\Delta|\!)^2 \vdash [\![P]\!]^2 \overset{\{\!|\ell_1|\!\}^2}{\longmapsto} (\!|\Gamma|\!)^2; \Delta'' \vdash [\![P']\!]^2 \mid s?(y).y?(x).[\![Q]\!]^2$

   b) *If* $\ell_1 = n?\langle\lambda y.Q\rangle$ *then* $\exists R$ *where either*
      - $(\!|\Gamma|\!)^2; (\!|\Delta|\!)^2 \vdash [\![P]\!]^2 \overset{\{\!|\ell_1|\!\}^2}{\longmapsto} \Gamma'; (\!|\Delta''|\!)^2 \vdash R$, *for some* Γ' *and* $(\!|\Gamma|\!)^2; (\!|\Delta'|\!)^2 \vdash [\![P']\!]^2 \approx (\!|\Delta''|\!)^2 \vdash (\nu\, a)(R \mid * a?(y).y?(x).[\![Q]\!]^2)$
      - $(\!|\Gamma|\!)^2; (\!|\Delta|\!)^2 \vdash [\![P]\!]^2 \overset{\{\!|\ell_1|\!\}^2}{\longmapsto} (\!|\Gamma|\!)^2; (\!|\Delta''|\!)^2 \vdash R$, *and* $(\!|\Gamma|\!)^2; (\!|\Delta'|\!)^2 \vdash [\![P']\!]^2 \approx (\!|\Delta''|\!)^2 \vdash (\nu\, s)(R \mid s?(y).y?(x).[\![Q]\!]^2)$

   c) *If* $\ell_1 = \tau$ *then either:*
      - $\exists R$ *such that*

$$\begin{array}{lllll} (\!|\Gamma|\!)^2; \emptyset; (\!|\Delta|\!)^2 & \overset{\tau}{\longmapsto} & (\!|\Delta'|\!)^2 \vdash & [\![P]\!]^2 \\ & \overset{\tau}{\longmapsto} & & (\nu\tilde{m})([\![P_1]\!]^2 \mid (\nu\, a)([\![P_2]\!]^2\{^a/_x\} \mid * a?(y).y?(x).[\![Q]\!]^2)) \end{array}$$

      - $\exists R$ *such that*

$$\begin{array}{lllll} (\!|\Gamma|\!)^2; \emptyset; (\!|\Delta|\!)^2 & \overset{\tau}{\longmapsto} & (\!|\Delta'|\!)^2 \vdash & [\![P]\!]^2 \\ & \overset{\tau}{\longmapsto} & & (\nu\tilde{m})([\![P_1]\!]^2 \mid (\nu\, s)([\![P_2]\!]^2\{^{\overline{s}}/_x\} \mid s?(y).y?(x).[\![Q]\!]^2)) \end{array}$$

      - $(\!|\Gamma|\!)^2; (\!|\Delta|\!)^2 \vdash [\![P]\!]^2 \overset{\tau}{\longmapsto} (\!|\Gamma|\!)^2; (\!|\Delta'|\!)^2 \vdash [\![P']\!]^2$
      - $\ell_1 = \tau_\beta$ *and* $(\!|\Gamma|\!)^2; (\!|\Delta|\!)^2 \vdash [\![P]\!]^2 \overset{\tau_s}{\longmapsto} (\!|\Gamma|\!)^2; (\!|\Delta'|\!)^2 \vdash [\![P']\!]^2$

   d) *If* $\ell_1 \in \{n \oplus l, n\&l\}$ *then*

      $\exists \ell_2 = \{\!|\ell_1|\!\}^2$ *such that* $(\!|\Gamma|\!)^2; (\!|\Delta|\!)^2 \vdash [\![P]\!]^2 \overset{\ell_2}{\longmapsto} (\!|\Gamma|\!)^2; (\!|\Delta'|\!)^2 \vdash [\![P']\!]^2$.

2. *Suppose* $(\!|\Gamma|\!)^2; (\!|\Delta|\!)^2 \vdash [\![P]\!]^2 \overset{\ell_2}{\longmapsto} (\!|\Delta'|\!)^2 \vdash R$.

a) If $\ell_2 = (\nu m)n!\langle m\rangle$ *then either*

- $\exists P'$ *such that* $P \overset{(\nu m)n!\langle m\rangle}{\longmapsto} P'$ *and* $R = [\![P']\!]^2$.
- $\exists Q, P'$ *such that* $P \overset{n!\langle\lambda x.\, Q\rangle}{\longmapsto} P'$ *and* $R = [\![P']\!]^2 \mid * \, a?(y).y?(x).[\![Q]\!]^2$
- $\exists Q, P'$ *such that* $P \overset{n!\langle\lambda x.\, Q\rangle}{\longmapsto} P'$ *and* $R = [\![P']\!]^2 \mid s?(y).y?(x).[\![Q]\!]^2$

b) If $\ell_2 = n?\langle m\rangle$ *then either*

- $\exists P'$ *such that* $P \overset{n?\langle m\rangle}{\longmapsto} P'$ *and* $R = [\![P']\!]^2$.
- $\exists Q, P'$ *such that* $P \overset{n?\langle\lambda x.\, Q\rangle}{\longmapsto} P'$
  and $(\!|\Gamma|\!)^2; (\!|\Delta'|\!)^2 \vdash [\![P']\!]^2 \approx (\!|\Delta'|\!)^2 \vdash (\nu a)(R \mid * \, a?(y).y?(x).[\![Q]\!]^2)$
- $\exists Q, P'$ *such that* $P \overset{n?\langle\lambda x.\, Q\rangle}{\longmapsto} P'$
  and $(\!|\Gamma|\!)^2; (\!|\Delta'|\!)^2 \vdash [\![P']\!]^2 \approx (\!|\Delta'|\!)^2 \vdash (\nu s)(R \mid s?(y).y?(x).[\![Q]\!]^2)$

c) If $\ell_2 = \tau$ *then* $\exists P'$ *such that* $P \overset{\tau}{\longmapsto} P'$ *and* $(\!|\Gamma|\!)^2; (\!|\Delta'|\!)^2 \vdash [\![P']\!]^2 \approx (\!|\Delta'|\!)^2 \vdash R$.

d) If $\ell_2 \notin \{n!\langle m\rangle, n \oplus l, n\&l\}$ *then* $\exists \ell_1$ *such that* $\ell_1 = \{\!|\ell_2|\!\}^2$ *and*
$\Gamma; \Delta \vdash P \overset{\ell_1}{\longmapsto} \Gamma; \Delta \vdash P'$.

Proof. The proof is done by transition induction. We conside the two parts separately.
- Part 1
- Basic Step:
- Subcase: $P = n!\langle\lambda x.\, Q\rangle.P'$ and also from Def. 7.5 we have that
$[\![P]\!]^2 = (\nu a)(n!\langle a\rangle.[\![P']\!]^2 \mid * \, a?(y).y?(x).[\![Q]\!]^2)$
Then

$$\begin{array}{rcl} \Gamma; \emptyset; \Delta \vdash P & \overset{n!\langle\lambda x.\, Q\rangle}{\longmapsto} & \Delta' \vdash P' \\ (\!|\Gamma|\!)^2; \emptyset; (\!|\Delta|\!)^2 \vdash [\![P]\!]^2 & \overset{(\nu a)n!\langle a\rangle}{\longmapsto} & (\!|\Delta|\!)^2 \vdash [\![P']\!]^2 \mid * \, a?(y).y?(x).[\![Q]\!]^2 \end{array}$$

and from Def. 7.5

$$\{\!|n!\langle\lambda x.\, Q\rangle|\!\} = (\nu a)n!\langle a\rangle$$

as required.
- Subcase: $P = n!\langle\lambda x.\, Q\rangle.P'$ and also from Def. 7.5 we have that
$[\![P]\!]^2 = (\nu s)(n!\langle \overline{s}\rangle.[\![P']\!]^2 \mid s?(y).y?(x).[\![Q]\!]^2)$ is similar as above.
- Subcase $P = n?(x).P'$.
- From Def. 7.5 we have that $[\![P]\!]^2 = n?(x).[\![P']\!]^2$
Then

$$\begin{array}{rcl} \Gamma; \emptyset; \Delta \vdash P & \overset{n?\langle\lambda x.\, Q\rangle}{\longmapsto} & \Delta' \vdash P'\{^{\lambda x.\, Q}/_x\} \\ (\!|\Gamma|\!)^2; \emptyset; (\!|\Delta|\!)^2 \vdash [\![P]\!]^2 & \overset{n?\langle a\rangle}{\longrightarrow} & (\!|\Delta''|\!)^2 \vdash R\{^a/_x\} \end{array}$$

with

$$\{\!|n?\langle\lambda x.\, Q\rangle|\!\}^2 \quad = \quad n?\langle a\rangle$$

It remains to show that

$$(\!|\Gamma|\!)^2; \emptyset; (\!|\Delta'|\!)^2 \vdash [\![P'\{^{\lambda x.\, Q}/_x\}]\!]^2 \approx (\!|\Delta''|\!)^2 \vdash (\nu a)(R\{^a/_x\} \mid * \, a?(y).y?(x).[\![Q]\!]^2)$$

The proof is an induction on the syntax structure of P'. Suppose $P' = xm$, then:

$$\begin{array}{rcl} [\![xm\{^{\lambda x.\, Q}/_x\}]\!]^2 & = & [\![Q\{^m/_x\}]\!]^2 \\ (\nu a)(R\{^a/_x\} \mid * \, a?(y).y?(x).[\![Q]\!]^2) & = & (\nu a)((\nu s)(x!\langle s\rangle.\overline{s}!\langle m\rangle.\mathbf{0})\{^a/_x\} \mid * \, a?(y).y?(x).[\![Q]\!]^2) \end{array}$$

The second term can be deterministically reduced as:

$$\begin{array}{rcll} (\!|\Gamma|\!)^2; \emptyset; (\!|\Delta''|\!)^2 & \overset{\tau}{\longmapsto}\overset{\tau_s}{\longmapsto} & (\!|\Delta''|\!)^2 \vdash & (\nu a)((\nu s)(x!\langle s\rangle.\overline{s}!\langle m\rangle.\mathbf{0})\{^a/_x\} \mid * \, a?(y).y?(x).[\![Q]\!]^2) \\ & \overset{\tau}{\longmapsto}\overset{\tau_s}{\longmapsto} & & (\nu a)([\![Q\{^m/_x\}]\!]^2 \mid * \, a?(y).y?(x).[\![Q]\!]^2) \end{array}$$

which is bisimilar with:

$$[\![Q\{^m/_x\}]\!]^2$$

because a is fresh and cannot interact anymore.
An interesting inductive step case is parallel composition. Suppose $P' = P_1 \mid P_2$. We need to show that:

$$(\!|\Gamma|\!)^2; \emptyset; (\!|\Delta'|\!)^2 \vdash [\![(P_1 \mid P_2)\{^{\lambda x.\, Q}/_x\}]\!]^2 \approx (\!|\Delta''|\!)^2 \vdash (\nu a)([\![P_1 \mid P_2]\!]^2\{^a/_x\} \mid * \, a?(y).y?(x).[\![Q]\!]^2)$$

We know that

$$\begin{array}{rcl}(\!|\Gamma|\!)^2;(\!|\Delta_1|\!)^2 \vdash [\![P_1\{^{\lambda x.\,Q}/_x\}]\!]^2 & \approx & (\!|\Delta_1''|\!)^2 \vdash (\nu a)([\![P_1]\!]^2\{^a/_x\} \mid * a?(y).y?(x).[\![Q]\!]^2)\\ (\!|\Gamma|\!)^2;(\!|\Delta_2|\!)^2 \vdash [\![P_2\{^{\lambda x.\,Q}/_x\}]\!]^2 & \approx & (\!|\Delta_1''|\!)^2 \vdash (\nu a)([\![P_2]\!]^2\{^a/_x\} \mid * a?(y).y?(x).[\![Q]\!]^2)\end{array}$$

We conclude from the congruence of $\approx$.
- The rest of the cases for Part 1 are easy to follow using Def. 7.5.
- Part 2.
The proof for Part 2 is straightforward following Def. 7.5. We give some distinctive cases:
- Case $P = n!\langle\lambda x.\,Q\rangle.P'$

$$\begin{array}{rcl}\Gamma;\Delta \vdash P & \overset{n!\langle\lambda x.\,Q\rangle}{\longmapsto} & \Delta' \vdash P'\\ (\!|\Gamma|\!)^2;(\!|\Delta|\!)^2 \vdash [\![P]\!]^2 & \overset{(\nu a)n!\langle a\rangle}{\longmapsto} & (\!|\Delta'|\!)^2 \vdash [\![P']\!]^2 \mid * a?(y).y?(s).[\![Q]\!]^2\end{array}$$

as required.
- Case $P = n?(x).P'$

$$\begin{array}{rcl}\Gamma;\Delta \vdash P & \overset{n?\langle\lambda x.\,Q\rangle}{\longmapsto} & \Delta' \vdash P'\{^{\lambda x.}/_Q\}x\\ (\!|\Gamma|\!)^2;(\!|\Delta|\!)^2 \vdash [\![P]\!]^2 & \overset{n?\langle a\rangle}{\longmapsto} & (\!|\Delta''|\!)^2 \vdash [\![P']\!]^2\{^a/_x\}\end{array}$$

We now use a similar argumentation as the input case in Part 1 to prove that:

$$\Gamma;\Delta' \vdash P'\{^{\lambda x.\,Q}/_x\} \approx (\!|\Delta''|\!)^2 \vdash (\nu a)([\![P']\!]^2\{^a/_x\} \mid * a?(y).y?(x).[\![Q]\!]^2)$$

□

Proposition B.10 (Full Abstraction, From HOπ to π). *Let P_1, Q_1 be HOπ processes.* $\Gamma;\Delta_1 \vdash P_1 \approx^H \Delta_2 \vdash Q_1$ *if and only if* $(\!|\Gamma|\!)^2;(\!|\Delta_1|\!)^2 \vdash [\![P_1]\!]^2 \approx^C (\!|\Delta_2|\!)^2 \vdash [\![Q_1]\!]^2$.

Proof.
Proof follows directly from Prop. B.9. The cases of Prop. B.9 are used to create a bisimulation closure to prove the the soundness direction and a bisimulation up to determinate transition (Lem. **??**) to prove the completeness direction. □

B.3 Properties for encoding $\mathcal{L}_{\mathsf{HO}\pi^+}$ into $\mathcal{L}_{\mathsf{HO}\pi}$

In this section we prove Thm. 8.1, in Page 5 that requires that encoding $\mathcal{L}_{\mathsf{HO}\pi^+}$ into $\mathcal{L}_{\mathsf{HO}\pi}$ is precise. A precise encoding requires to prove three independent results:

- Type preservation, stated in Prop. B.11.
- Operational Correspondence, stated in Prop. B.13. Note that we prove a stronger operational correspondence condition, as in Def. B.1, than the condition suggested in Def. 6.4(2).
- Full Abstraction, stated in Prop. B.14.

Proposition B.11 (Type Preservation. From HOπ^+ to HOπ). *Let P be a HOπ^+ process. If* $\Gamma;\emptyset;\Delta \vdash P \triangleright \diamond$ *then* $(\!|\Gamma|\!)^3;\emptyset;(\!|\Delta|\!)^3 \vdash [\![P]\!]^3 \triangleright \diamond$.

Proof. By induction on the inference of $\Gamma;\emptyset;\Delta \vdash P \triangleright \diamond$. We detail some representative cases:

1. Case $P = u!\langle\lambda\underline{x}.\,Q\rangle.P'$. Then we may have the following typing in HOπ^+:

$$\dfrac{\dfrac{}{\Gamma;\Lambda_1;\Delta_1 \cdot u : S \vdash P' \triangleright \diamond} \quad \dfrac{\overline{\Gamma \cdot \underline{x} : L;\Lambda_2;\Delta_2 \vdash Q \triangleright \diamond} \quad \overline{\Gamma \cdot \underline{x} : L;\emptyset;\emptyset \vdash \underline{x} \triangleright L}}{\Gamma;\Lambda_2;\Delta_2 \vdash \lambda\underline{x} : L.\,Q \triangleright L \multimap \diamond}}{\Gamma;\Lambda_1 \cdot \Lambda_2;\Delta_1 \cdot \Delta_2 \cdot u :!\langle L \multimap \diamond\rangle;S \vdash u!\langle\lambda\underline{x}.\,Q\rangle.P' \triangleright \diamond}$$

Thus, by IH we have:

$$\begin{array}{rcll}(\!|\Gamma|\!)^3;(\!|\Lambda_1|\!)^3;(\!|\Delta_1|\!)^3 \cdot u : (\!|S|\!)^3 & \vdash & [\![P']\!]^3 \triangleright \diamond & (21)\\ (\!|\Gamma|\!)^3 \cdot \underline{x} : (\!|L|\!)^3;(\!|\Lambda_2|\!)^3;(\!|\Delta_2|\!)^3 & \vdash & [\![Q]\!]^3 \triangleright \diamond & (22)\\ (\!|\Gamma|\!)^3 \cdot \underline{x} : (\!|L|\!)^3;\emptyset;\emptyset & \vdash & \underline{x} \triangleright (\!|L|\!)^3 & (23)\end{array}$$

The corresponding typing in HOπ is as follows:

$$\dfrac{\dfrac{\dfrac{\overline{(22)}}{(\!|\Gamma|\!)^3 \cdot x : (\!|L|\!)^3;(\!|\Lambda_2|\!)^3;(\!|\Delta_2|\!)^3 \cdot z : \mathsf{end} \vdash [\![Q]\!]^3 \triangleright \diamond} \quad \overline{(23)}}{(\!|\Gamma|\!)^3;(\!|\Lambda_2|\!)^3;(\!|\Delta_2|\!)^3 \cdot z :?((\!|L|\!)^3);\mathsf{end} \vdash z?(\underline{x}).[\![Q]\!]^3 \triangleright \diamond} \quad (24)$$

$$\dfrac{\overline{(21)} \quad \dfrac{(24) \quad \overline{(\!|\Gamma|\!)^3;\emptyset;z :?((\!|L|\!)^3);\mathsf{end} \vdash z \triangleright ?((\!|L|\!)^3);\mathsf{end}}}{(\!|\Gamma|\!)^3;(\!|\Lambda_2|\!)^3;(\!|\Delta_2|\!)^3 \vdash \lambda z.\,z?(\underline{x}).[\![Q]\!]^3 \triangleright (?((\!|L|\!)^3);\mathsf{end}) \multimap \diamond}}{(\!|\Gamma|\!)^3;(\!|\Lambda_1|\!)^3 \cdot (\!|\Lambda_2|\!)^3;(\!|\Delta_1|\!)^3 \cdot (\!|\Delta_2|\!)^3 \cdot u :!\langle?((\!|L|\!)^3);\mathsf{end} \multimap \diamond\rangle;(\!|S|\!)^3 \vdash u!\langle\lambda z.\,z?(\underline{x}).[\![Q]\!]^3\rangle.[\![P']\!]^3 \triangleright \diamond}$$

2. Case $P = (\lambda x.P)(\lambda y.Q)$. We may have different possibilities for the types of each abstraction. We consider only one of them, as the rest are similar:

$$\dfrac{\dfrac{\overline{\Gamma\cdot x : C{\to}\diamond;\Lambda;\Delta_1 \vdash P \triangleright \diamond}}{\Gamma;\Lambda;\Delta_1 \vdash \lambda x.P \triangleright (C{\multimap}\diamond){\multimap}\diamond} \quad \dfrac{\overline{\Gamma;\emptyset;\Delta_2, y : C \vdash Q \triangleright \diamond}}{\Gamma;\emptyset;\Delta_2 \vdash \lambda y.Q \triangleright C{\multimap}\diamond}}{\Gamma;\Lambda;\Delta_1\cdot\Delta_2 \vdash (\lambda x.P)(\lambda y.Q) \triangleright \diamond}$$

Thus, by IH we have:

$$(\!|\Gamma|\!)^3 \cdot x : (\!|C{\to}\diamond|\!)^3; (\!|\Lambda|\!)^3; (\!|\Delta_1|\!)^3 \quad \vdash \quad [\![P]\!]^3 \triangleright \diamond \tag{25}$$

$$(\!|\Gamma|\!)^3; \emptyset; (\!|\Delta_1|\!)^3, y : (\!|C|\!)^3 \quad \vdash \quad [\![Q]\!]^3 \triangleright \diamond \tag{26}$$

The corresponding typing in HOπ is as follows — recall that $(\!|C{\multimap}\diamond|\!)^3 = (\!|C|\!)^3{\multimap}\diamond$.

$$\dfrac{\dfrac{\overline{(25)}}{(\!|\Gamma|\!)^3 \cdot x : (\!|C{\to}\diamond|\!)^3; (\!|\Lambda|\!)^3; (\!|\Delta_1|\!)^3 \cdot s : \mathsf{end} \vdash [\![P]\!]^3 \triangleright \diamond}}{(\!|\Gamma|\!)^3; (\!|\Lambda|\!)^3; (\!|\Delta_1|\!)^3 \cdot s : ?((\!|C{\multimap}\diamond|\!)^3); \mathsf{end} \vdash s?(x).[\![P]\!]^3 \triangleright \diamond} \tag{27}$$

(27)
$$\dfrac{\dfrac{\dfrac{\dfrac{\overline{(26)}}{(\!|\Gamma|\!)^3; \emptyset; (\!|\Delta_2|\!)^3 \cdot y : (\!|C|\!)^3 \vdash [\![Q]\!]^3 \triangleright \diamond}}{(\!|\Gamma|\!)^3; \emptyset; (\!|\Delta_2|\!)^3 \vdash \lambda y.[\![Q]\!]^3 \triangleright (\!|C{\multimap}\diamond|\!)^3}}{(\!|\Gamma|\!)^3; \emptyset; (\!|\Delta_2|\!)^3 \cdot \overline{s} : \mathsf{end} \vdash \lambda y.[\![Q]\!]^3 \triangleright (\!|C{\multimap}\diamond|\!)^3}}{(\!|\Gamma|\!)^3; \emptyset; (\!|\Delta_2|\!)^3 \cdot \overline{s} : !\langle (\!|C{\multimap}\diamond|\!)^3\rangle; \mathsf{end} \vdash \overline{s}!\langle \lambda y.[\![Q]\!]^3\rangle.\mathbf{0} \triangleright \diamond}}{\dfrac{(\!|\Gamma|\!)^3; (\!|\Lambda|\!)^3; (\!|\Delta_1|\!)^3 \cdot (\!|\Delta_2|\!)^3 \cdot s : ?((\!|C{\multimap}\diamond|\!)^3); \mathsf{end} \cdot \overline{s} : !\langle (\!|C{\multimap}\diamond|\!)^3\rangle; \mathsf{end} \vdash s?(x).[\![P]\!]^3 \mid \overline{s}!\langle \lambda y.[\![Q]\!]^3\rangle.\mathbf{0} \triangleright \diamond}{(\!|\Gamma|\!)^3; (\!|\Lambda|\!)^3; (\!|\Delta_1|\!)^3 \cdot (\!|\Delta_2|\!)^3 \vdash (\nu\, s)(s?(x).[\![P]\!]^3 \mid \overline{s}!\langle \lambda y.[\![Q]\!]^3\rangle.\mathbf{0}) \triangleright \diamond}$$

□

Before we prove operational correspondence we define mapping from $\{\!|\cdot|\!\}^3 : \mathcal{A} \to \mathcal{A}$ where $\mathcal{A}$ is the set of labels of the relation $\overset{\ell}{\longmapsto}$:

Definition B.12 ($\{\!|\cdot|\!\}^3 : \mathcal{A} \to \mathcal{A}$). Let $\mathcal{A}$ is the set of labels of the relation $\overset{\ell}{\longmapsto}$ then we define:

$$\{\!|(\nu\,\tilde{m})n!\langle\lambda x{:}L.P\rangle|\!\}^3 \stackrel{\mathtt{def}}{=} (\nu\,\tilde{m})n!\langle\lambda z.z?(x).[\![P]\!]^3\rangle \qquad\qquad \{\!|n?\langle\lambda x{:}L.P\rangle|\!\}^3 \stackrel{\mathtt{def}}{=} n?\langle\lambda z.z?(x).[\![P]\!]^3\rangle$$

and homomorphic for all other cases of $\ell \in \mathcal{A}$.

We now state and prove a detailed version of the operational corresponce in Def. B.1.

Proposition B.13 (Operational Correspondence. From HOπ^+ to HOπ). *1. Let* $\Gamma;\emptyset;\Delta \vdash P$. $\Gamma;\Delta \vdash P \overset{\ell}{\longmapsto} \Delta' \vdash P'$ *implies*

a) If $\ell \in \{(\nu\,\tilde{m})n!\langle\lambda x.Q\rangle, n?\langle\lambda x.Q\rangle\}$ *then* $(\!|\Gamma|\!)^3;(\!|\Delta|\!)^3 \vdash [\![P]\!]^3 \overset{\ell'}{\longmapsto} (\!|\Delta'|\!)^3 \vdash [\![P']\!]^3$ *with* $\{\!|\ell|\!\}^3 = \ell'$.

b) If $\ell \notin \{(\nu\,\tilde{m})n!\langle\lambda x.Q\rangle, n?\langle\lambda x.Q\rangle, \tau\}$ *then* $(\!|\Gamma|\!)^3;(\!|\Delta|\!)^3 \vdash [\![P]\!]^3 \overset{\ell}{\longmapsto} (\!|\Delta'|\!)^3 \vdash [\![P']\!]^3$.

c) If $\ell = \tau_\beta$ *then* $(\!|\Gamma|\!)^3;(\!|\Delta|\!)^3 \vdash [\![P]\!]^3 \overset{\tau}{\longmapsto} \Delta'' \vdash R$ *and* $(\!|\Gamma|\!)^3(\!|\Delta'|\!)^3[\![P']\!]^3 \approx \Delta'' R$.

d) If $\ell = \tau$ *and* $\ell \neq \tau_\beta$ *then* $(\!|\Gamma|\!)^3;(\!|\Delta|\!)^3 \vdash [\![P]\!]^3 \overset{\tau}{\longmapsto} (\!|\Delta'|\!)^3 \vdash [\![P']\!]^3$.

2. Let $\Gamma;\emptyset;\Delta \vdash P$. $(\!|\Gamma|\!)^3;(\!|\Delta|\!)^3 \vdash [\![P]\!]^3 \overset{\ell}{\longmapsto} (\!|\Delta''|\!)^3 \vdash Q$ *implies*

a) If $\ell \in \{(\nu\,\tilde{m})n!\langle\lambda x.Q\rangle, n?\langle\lambda x.Q\rangle, \tau\}$ *then* $\Gamma;\Delta \vdash P \overset{\ell'}{\longmapsto} \Delta' \vdash P'$ *with* $\{\!|\ell'|\!\}^3 = \ell$ *and* $Q \equiv [\![P']\!]^3$.

b) If $\ell \notin \{(\nu\,\tilde{m})n!\langle\lambda x.R\rangle, n?\langle\lambda x.R\rangle, \tau\}$ *then* $\Gamma;\Delta \vdash P \overset{\ell}{\longmapsto} \Delta' \vdash P'$ *and* $Q \equiv [\![P']\!]^3$.

c) If $\ell = \tau$ *then either* $\Gamma;\Delta \vdash \Delta \overset{\tau}{\longmapsto} \Delta' \vdash P'$ *with* $Q \equiv [\![P']\!]^3$
or $\Gamma;\Delta \vdash \Delta \overset{\tau_\beta}{\longmapsto} \Delta' \vdash P'$ *and* $(\!|\Gamma|\!)^3;(\!|\Delta''|\!)^3 \vdash Q \overset{\tau_\beta}{\longmapsto} (\!|\Delta''|\!)^3 \vdash [\![P']\!]^3$.

Proof.

1. The proof of Part 1 does a transition induction and considers the mapping as defined in Fig. 7. We give the most interesting cases.

- Case: $P = (\lambda x.Q_1)\,\lambda x.Q_2$.

  $\Gamma;\Delta \vdash (\lambda x.Q_1)\,\lambda x.Q_2 \overset{\tau_\beta}{\longmapsto} \Delta \vdash Q_1\{^{\lambda x.Q_2}/_x\}$ implies

  $$(\!|\Gamma|\!)^3;(\!|\Delta|\!)^3 \vdash (\nu\, s)(s?(x).[\![Q_1]\!]^3 \mid \overline{s}!\langle\lambda x.[\![Q_2]\!]^3\rangle.\mathbf{0}) \overset{\tau_s}{\longmapsto} (\!|\Delta'|\!)^3 \vdash [\![Q_1]\!]^3\{^{\lambda x.[\![Q_2]\!]^3}/_x\}$$

- Case: $P = n!\langle\lambda \underline{x}.Q\rangle.P$

  $\Gamma;\Delta \vdash n!\langle\lambda \underline{x}.Q\rangle.P \overset{n!\langle\lambda x.Q\rangle}{\longmapsto} \Delta \vdash P$ implies

  $(\!|\Gamma|\!)^3;(\!|\Delta|\!)^3 \vdash n!\langle\lambda z.z?(x).[\![Q]\!]^3\rangle.[\![P]\!]^3 \overset{n!\langle\lambda z.z?(x).[\![Q]\!]^3\rangle}{\longmapsto} \Delta \vdash [\![P]\!]^3$

- Other cases are similar.

2. The proof of Part 2 also does a transition induction and considers the mapping as defined in Fig. 7. We give the most interesting cases.
   - Case: $P = (\lambda x.\, Q_1)\,\lambda x.\, Q_2$.

$$\begin{array}{lll} (\!|\Gamma|\!)^3;\emptyset;(\!|\Delta|\!)^3 & \stackrel{\tau_\beta}{\longmapsto} & (\!|\Delta'|\!)^3 \vdash \quad (\nu\, s)((\lambda z.\, z?(x).[\![Q]\!]^3)\, s \mid \overline{s}!\langle \lambda x.\, Q_2\rangle.\mathbf{0}) \\ & \stackrel{\tau_\beta}{\longmapsto} & \qquad\qquad (\nu\, s)(s?(x).[\![Q]\!]^3 \mid \overline{s}!\langle \lambda x.\, Q_2\rangle.\mathbf{0}) \end{array}$$

     implies $\Gamma;\Delta \vdash (\lambda x.\, Q_1)\,\lambda x.\, Q_2 \stackrel{\tau_\beta}{\longmapsto} \Delta \vdash Q_1\{^{\lambda x.\, Q_2}/_x\}$ and

$$\begin{array}{lll} (\!|\Gamma|\!)^3;\emptyset;(\!|\Delta|\!)^3 & \stackrel{\tau_s}{\longmapsto} & (\!|\Delta'|\!)^3 \vdash \quad (\nu\, s)(s?(x).[\![Q]\!]^3 \mid \overline{s}!\langle \lambda x.\, Q_2\rangle.\mathbf{0}) \\ & \stackrel{\tau_s}{\longmapsto} & \qquad\qquad [\![Q_1]\!]^3\{^{\lambda x.\, [\![Q_2]\!]^3}/_x\} \end{array}$$

   - Case: $P = n!\langle \lambda \underline{x}.\, Q\rangle.P$

     $(\!|\Gamma|\!)^3;(\!|\Delta|\!)^3 \vdash n!\langle \lambda z.\, z?(x).[\![Q]\!]^3\rangle.[\![P]\!]^3 \stackrel{n!\langle \lambda z.\, z?(x).[\![Q]\!]^3\rangle}{\longmapsto} \Delta \vdash [\![P]\!]^3$ and

     $\Gamma;\Delta \vdash n!\langle \lambda \underline{x}.\, Q\rangle.P \stackrel{n!\langle \lambda \underline{x}.\, Q\rangle}{\longmapsto} \Delta \vdash P$
   - Other cases are similar.

□

Proposition B.14 (Full Abstraction. From $\mathsf{HO}\pi^+$ to $\mathsf{HO}\pi$). *Let P, Q $\mathsf{HO}\pi^+$ processes with $\Gamma;\emptyset;\Delta_1 \vdash P \triangleright \diamond$ and $\Gamma;\emptyset;\Delta_2 \vdash Q \triangleright \diamond$. Then $\Gamma;\Delta_1 \vdash P \approx \Delta_2 \vdash Q$ if and only if $(\!|\Gamma|\!)^3;(\!|\Delta_1|\!)^3 \vdash [\![P]\!]^3 \approx (\!|\Delta_2|\!)^3 \vdash [\![Q]\!]^3$*

Proof. **Soundness Direction.**
We create the closure

$$\mathfrak{R} = \{\Gamma;\Delta_1 \vdash P\ ,\ \Delta_2 \vdash Q \ \mid\ (\!|\Gamma|\!)^3;(\!|\Delta_1|\!)^3 \vdash [\![P]\!]^3 \approx (\!|\Delta_2|\!)^3 \vdash [\![Q]\!]^3\}$$

It is straightforward to show that $\mathfrak{R}$ is a bisimulation if we follow Part 2 of Prop. B.13 for subcases a and b. In subcase c we make use of Prop. **??**.
Completeness Direction.
We create the closure

$$\mathfrak{R} = \{(\!|\Gamma|\!)^3;(\!|\Delta_1|\!)^3 \vdash [\![P]\!]^3\ ,\ (\!|\Delta_2|\!)^3 \vdash [\![Q]\!]^3 \ \mid\ \Gamma;\Delta_1 \vdash P \approx \Delta_2 \vdash Q\}$$

We show that $\mathfrak{R}$ is a bisimulation up to deterministic transitions by following Part 1 of Prop. B.13. The proof is straightforward for subcases a), b) and d). In subcase c) we make use of Lem. **??**. □

B.4 Properties for encoding $\mathcal{L}_{\mathsf{HO}\vec{\pi}}$ into $\mathcal{L}_{\mathsf{HO}\pi}$

In this section we prove Thm. 8.2, in Page 5 that requires that encoding $\mathcal{L}_{\mathsf{HO}\vec{\pi}}$ into $\mathcal{L}_{\mathsf{HO}\pi}$ is precise. A precise encoding requires to prove three independent results:

- Type preservation, stated in Prop. B.15.
- Operational Correspondence, stated in Prop. B.17. Note that we prove a stronger operational correspondence condition, as in Def. B.1, than the condition suggested in Def. 6.4(2).
- Full Abstraction, stated in Prop. B.18.

Proposition B.15 (Type Preservation. From $\mathsf{HO}\vec{\pi}$ to $\mathsf{HO}\pi$). Let P be a $\mathsf{HO}\vec{\pi}$ process. If $\Gamma;\emptyset;\Delta \vdash P \triangleright \diamond$ then $(\!|\Gamma|\!)^4;\emptyset;(\!|\Delta|\!)^4 \vdash [\![P]\!]^4 \triangleright \diamond$.

Proof. By induction on the inference $\Gamma;\emptyset;\Delta \vdash P \triangleright \diamond$. We examine two representative cases, using biadic communications.

1. Case $P = n!\langle V\rangle.P'$ and $\Gamma;\emptyset;\Delta_1 \cdot \Delta_2 \cdot n : !\langle (C_1,C_2) \multimap \diamond\diamond\rangle; S \vdash n!\langle V\rangle.P' \triangleright \diamond$. Then either $V = y$ or $V = \lambda(x_1,x_2).\, Q$, for some Q. The case $V = y$ is immediate; we give details for the case $V = \lambda(x_1,x_2).\, Q$, for which we have the following typing:

$$\dfrac{\dfrac{}{\Gamma;\emptyset;\Delta_1 \cdot n : S \vdash P' \triangleright \diamond} \quad \dfrac{\Gamma;\emptyset;\Delta_2 \cdot x_1 : C_1 \cdot x_2 : C_2 \vdash Q \triangleright \diamond}{\Gamma;\emptyset;\Delta_2 \vdash \lambda(x_1,x_2).\, Q \triangleright (C_1,C_2) \multimap \diamond\diamond}}{\Gamma;\emptyset;\Delta_1 \cdot \Delta_2 \cdot n : !\langle (C_1,C_2) \multimap \diamond\diamond\rangle; S \vdash k!\langle \lambda(x_1,x_2).\, Q\rangle.P \triangleright \diamond}$$

   We now show the typing for $[\![P]\!]^4$. By IH we have both:

$$(\!|\Gamma|\!)^4;\emptyset;(\!|\Delta_1|\!)^4 \cdot n : (\!|S|\!)^4 \vdash [\![P']\!]^4 \triangleright \diamond \qquad (\!|\Gamma|\!)^4;\emptyset;(\!|\Delta_2|\!)^4 \cdot x_1 : (\!|C_1|\!)^4 \cdot x_2 : (\!|C_2|\!)^4 \vdash [\![Q]\!]^4 \triangleright \diamond$$

   Let $L = (C_1,C_2) \multimap \diamond\diamond$. By Fig. 8 we have $(\!|L|\!)^4 = (?((\!|C_1|\!)^4); ?((\!|C_2|\!)^4); \mathsf{end}) \multimap \diamond\diamond$ and $[\![P]\!]^4 = n!\langle \lambda z.\, z?(x_1).z?(x_2).[\![Q]\!]^4\rangle.[\![P']\!]^4$. We can now infer the following typing derivation:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\overline{(\!|\Gamma|\!)^4;\emptyset;(\!|\Delta_2|\!)^4 \cdot x_1 : (\!|C_1|\!)^4 \cdot x_2 : (\!|C_2|\!)^4 \vdash [\![Q]\!]^4 \triangleright \diamond}}{(\!|\Gamma|\!)^4;\emptyset;(\!|\Delta_2|\!)^4 \cdot x_1 : (\!|C_1|\!)^4 \cdot x_2 : (\!|C_2|\!)^4 \cdot z : \mathsf{end} \vdash [\![Q]\!]^4 \triangleright \diamond}}{(\!|\Gamma|\!)^4;\emptyset;(\!|\Delta_2|\!)^4 \cdot x_1 : (\!|C_1|\!)^4 \cdot z : ?((\!|C_2|\!)^4); \mathsf{end} \vdash z?(x_2).[\![Q]\!]^4 \triangleright \diamond}}{(\!|\Gamma|\!)^4;\emptyset;(\!|\Delta_2|\!)^4 \cdot z : ?((\!|C_1|\!)^4); ?((\!|C_2|\!)^4); \mathsf{end} \vdash z?(x_1).z?(x_2).[\![Q]\!]^4 \triangleright \diamond}}{(\!|\Gamma|\!)^4;\emptyset;(\!|\Delta_2|\!)^4 \vdash \lambda z.\, z?(x_1).z?(x_2).[\![Q]\!]^4 \triangleright ((\!|C_1|\!)^4,(\!|C_2|\!)^4) \multimap \diamond\diamond} \qquad (28)$$

$$\dfrac{\dfrac{}{(\!|\Gamma|\!)^{\mathsf{p}};\emptyset;(\!|\Delta_1|\!)^{\mathsf{p}}\cdot k:(\!|S|\!)^{\mathsf{p}}\vdash [\![P']\!]^{\mathsf{p}}\triangleright\diamond}}{(\!|\Gamma|\!)^4;\emptyset;(\!|\Delta_1|\!)^4\cdot(\!|\Delta_2|\!)^4\cdot n:!\langle(\!|L|\!)^4\rangle;(\!|S|\!)^4\vdash [\![P]\!]^4\triangleright\diamond}\quad(28)$$

2. Case $P = n?(x_1,x_2).P'$ and $\Gamma;\emptyset;\Delta_1\cdot n:?((C_1,C_2));S \vdash n?(x_1,x_2).P' \triangleright \diamond$. We have the following typing derivation:

$$\dfrac{\Gamma;\emptyset;\Delta_1\cdot n:S\cdot x_1:C_1\cdot x_2:C_2\vdash P'\triangleright\diamond \quad \Gamma;\emptyset;\vdash x_1,x_2\triangleright C_1,C_2}{\Gamma;\emptyset;\Delta_1\cdot n:?((C_1,C_2));S\vdash n?(x_1,x_2).P'\triangleright\diamond}$$

By Fig. 8 we have $[\![P]\!]^4 = n?(x_1).k?(x_2).[\![P']\!]^4$. By IH we have

$$(\!|\Gamma|\!)^4;\emptyset;(\!|\Delta_1|\!)^4\cdot n:(\!|S|\!)^4\cdot x_1:(\!|C_1|\!)^4\cdot x_2:(\!|C_2|\!)^4\vdash [\![P']\!]^4\triangleright\diamond$$

and the following type derivation:

$$\dfrac{\dfrac{\dfrac{}{(\!|\Gamma|\!)^4;\emptyset;(\!|\Delta_1|\!)^4\cdot x_1:(\!|C_1|\!)^4\cdot x_2:(\!|C_2|\!)^4\cdot n:(\!|S|\!)^4\vdash [\![P']\!]^4\triangleright\diamond}}{(\!|\Gamma|\!)^4;\emptyset;(\!|\Delta_1|\!)^4\cdot x_1:(\!|C_1|\!)^4\cdot n:?((\!|C_2|\!)^4);(\!|S|\!)^4\vdash n?(x_2).[\![P']\!]^4\triangleright\diamond}}{(\!|\Gamma|\!)^4;\emptyset;(\!|\Delta_1|\!)^4\cdot n:?((\!|C_1|\!)^4);?((\!|C_2|\!)^4);(\!|S|\!)^4\vdash [\![P]\!]^4\triangleright\diamond}$$

□

Before we prove operational correspondence we define mapping from $\{\!|\cdot|\!\}^4 : \mathcal{A}\to\mathcal{A}$ where $\mathcal{A}$ is the set of labels of the relation $\overset{\ell}{\longmapsto}$:

Definition B.16 ($\{\!|\cdot|\!\}^4 : \mathcal{A}\to\mathcal{A}$). Let $\mathcal{A}$ is the set of labels of the relation $\overset{\ell}{\longmapsto}$ then we define:

$$\{\!|(\nu\tilde{m})n!\langle m_1,m_2\rangle|\!\}^4 \stackrel{\mathtt{def}}{=} \ell_1,\ell_2 \text{ where } (m_i\in\tilde{m}\Leftrightarrow \ell_i=(\nu m_i)n!\langle m_i\rangle)\ \vee\ (m_i\notin\tilde{m}\Leftrightarrow \ell_i = n!\langle m_i\rangle)$$
$$\{\!|(\nu\tilde{m})n!\langle\lambda(x_1,x_2).P\rangle|\!\}^4 \stackrel{\mathtt{def}}{=} (\nu\tilde{m})n!\langle\lambda z.z?(x_1).z?(x_2).[\![P]\!]^4\rangle$$

and homomorphic for all other cases of $\ell\in\mathcal{A}$.

Proposition B.17 (Operational Correspondence. From $\mathsf{HO}\vec{\pi}$ to $\mathsf{HO}\pi$). *1. Let* $\Gamma;\emptyset;\Delta\vdash P$. *Then* $\Gamma;\Delta\vdash P\overset{\ell}{\longmapsto}\Delta'\vdash P'$ *implies*

a) If $\ell=(\nu\tilde{m}')n!\langle\tilde{m}\rangle$ *then* $(\!|\Gamma|\!)^4;(\!|\Delta|\!)^4\vdash[\![P]\!]^4\overset{\ell_1}{\longmapsto}\dots\overset{\ell_n}{\longmapsto}(\!|\Delta'|\!)^4\vdash[\![P]\!]^4$ *with* $\{\!|\ell|\!\}^4=\ell_1\dots\ell_n$.

b) If $\ell=n?\langle\tilde{m}\rangle$ *then* $(\!|\Gamma|\!)^4;(\!|\Delta|\!)^4\vdash[\![P]\!]^4\overset{\ell_1}{\longmapsto}\dots\overset{\ell_n}{\longmapsto}(\!|\Delta'|\!)^4\vdash[\![P]\!]^4$ *with* $\{\!|\ell|\!\}^4=\ell_1\dots\ell_n$.

c) If $\ell\in\{(\nu\tilde{m})n!\langle\lambda\tilde{x}.R\rangle,n?\langle\lambda\tilde{x}.R\rangle\}$ *then* $(\!|\Gamma|\!)^4;(\!|\Delta|\!)^4\vdash[\![P]\!]^4\overset{\ell'}{\longmapsto}(\!|\Delta'|\!)^4\vdash[\![P']\!]^4$ *with* $\{\!|\ell|\!\}^4=\ell'$.

d) If $\ell\in\{n\oplus l,n\&l\}$ *then* $(\!|\Gamma|\!)^4;(\!|\Delta|\!)^4\vdash[\![P]\!]^4\overset{\ell}{\longmapsto}(\!|\Delta'|\!)^4\vdash[\![P']\!]^4$.

e) If $\ell=\tau_\beta$ *then either* $(\!|\Gamma|\!)^4;(\!|\Delta|\!)^4\vdash[\![P]\!]^4\overset{\tau_\beta}{\longmapsto}\overset{\tau_s}{\longmapsto}\dots\overset{\tau_s}{\longmapsto}(\!|\Delta'|\!)^4\vdash[\![P']\!]^4$ *with* $\{\!|\ell|\!\}=\tau_\beta,\tau_s\dots\tau_s$.

f) If $\ell=\tau$ *then* $(\!|\Gamma|\!)^4;(\!|\Delta|\!)^4\vdash[\![P]\!]^4\overset{\tau}{\longmapsto}\dots\overset{\tau}{\longmapsto}(\!|\Delta'|\!)^4\vdash[\![P']\!]^4$ *with* $\{\!|\ell|\!\}^4=\tau\dots\tau$.

2. Let $\Gamma;\emptyset;\Delta\vdash P$. $(\!|\Gamma|\!)^4;(\!|\Delta|\!)^4\vdash[\![P]\!]^4\overset{\ell_1}{\longmapsto}(\!|\Delta_1|\!)^4\vdash P_1$ *implies*

a) If $\ell\in\{n?\langle m\rangle,n!\langle m\rangle,(\nu m)n!\langle m\rangle\}$ *then* $\Gamma;\Delta\vdash P\overset{\ell}{\longmapsto}\Delta'\vdash P'$ *and*
$(\!|\Gamma|\!)^4;(\!|\Delta_1|\!)^4\vdash P_1\overset{\ell_2}{\longmapsto}\dots\overset{\ell_n}{\longmapsto}(\!|\Delta'|\!)^4\vdash(\!|P'|\!)^4$ *with* $\{\!|\ell|\!\}^4=\ell_1\dots\ell_n$.

b) If $\ell\in\{(\nu\tilde{m})n!\langle\lambda x.R\rangle,n?\langle\lambda x.R\rangle\}$ *then* $\Gamma;\Delta\vdash P\overset{\ell'}{\longmapsto}\Delta'\vdash P'$ *with* $\{\!|\ell'|\!\}^4=\ell$ *and* $P_1\equiv[\![P']\!]^4$.

c) If $\ell\in\{n\oplus l,n\&l\}$ *then* $\Gamma;\Delta\vdash P\overset{\ell}{\longmapsto}\Delta'\vdash P'$ *and* $P_1\equiv[\![P']\!]^4$.

d) If $\ell=\tau_\beta$ *then* $\Gamma;\Delta\vdash P\overset{\tau_\beta}{\longmapsto}\Delta'\vdash P'$ *and* $(\!|\Gamma|\!)^4;(\!|\Delta_1|\!)^4\vdash P_1\overset{\tau_s}{\longmapsto}\dots\overset{\tau_s}{\longmapsto}(\!|\Delta'|\!)^4\vdash(\!|P'|\!)^4$ *with* $\{\!|\ell|\!\}^4=\tau_\beta,\tau_s\dots\tau_s$.

e) If $\ell=\tau$ *then* $\Gamma;\Delta\vdash P\overset{\tau}{\longmapsto}\Delta'\vdash P'$ *and* $(\!|\Gamma|\!)^4;(\!|\Delta_1|\!)^4\vdash P_1\overset{\tau}{\longmapsto}\dots\overset{\tau}{\longmapsto}(\!|\Delta'|\!)^4\vdash(\!|P'|\!)^4$ *with* $\{\!|\ell|\!\}^4=\tau\dots\tau$.

Proof. The proof of both parts is by transition induction, following the mapping defined in Fig. 8 . We consider some representative cases, using biadic communication:

- Case (1(a)), with $P = n!\langle m_1,m_2\rangle.P'$ and $\ell_1 = n!\langle m_1,m_2\rangle$. By assumption, P is well-typed. As one particular possibility, we may have:

$$\dfrac{\Gamma;\emptyset;\Delta_0\cdot n:S\vdash P'\triangleright\diamond \quad \Gamma;\emptyset;m_1{:}S_1\cdot m_2{:}S_2\vdash m_1,m_2\triangleright S_1,S_2}{\Gamma;\emptyset;\Delta_0\cdot m_1{:}S_1\cdot m_2{:}S_2\cdot n:!\langle S_1,S_2\rangle;S\vdash n!\langle m_1,m_2\rangle.P'\triangleright\diamond}$$

for some $\Gamma,S,S_1,S_2,\Delta_0$, such that $\Delta=\Delta_0\cdot m_1{:}S_1\cdot m_2{:}S_2\cdot n:!\langle S_1,S_2\rangle;S$. We may then have the following typed transition

$$\Gamma;\Delta_0\cdot m_1{:}S_1\cdot m_2{:}S_2\cdot n:!\langle S_1,S_2\rangle;S\vdash n!\langle m_1,m_2\rangle.P'\overset{\ell_1}{\longmapsto}\Delta_0\cdot n{:}S\vdash P'$$

The encoding of the source judgment for P is as follows:

$$(\!|\Gamma|\!)^4;\emptyset;(\!|\Delta_0\cdot m_1{:}S_1\cdot m_2{:}S_2\cdot n:!\langle S_1,S_2\rangle;S|\!)^4\vdash[\![n!\langle m_1,m_2\rangle.P']\!]^4\triangleright\diamond$$

which, using Fig. 8 , can be expressed as

$$(\!|\Gamma|\!)^4;\emptyset;(\!|\Delta_0\cdot m_1{:}(\!|S_1|\!)^4\cdot m_2{:}(\!|S_2|\!)^4\cdot n:!\langle(\!|S_1|\!)^4\rangle;!\langle(\!|S_2|\!)^4\rangle;(\!|S|\!)^4\vdash n!\langle m_1\rangle.n!\langle m_2\rangle.[\![P']\!]^4\triangleright\diamond$$

Now, $\{\!|\ell_1|\!\}^4 = n!\langle m_1\rangle, n!\langle m_2\rangle$. It is immediate to infer the following typed transitions for $[\![P]\!]^4 = n!\langle m_1\rangle.n!\langle m_2\rangle.[\![P']\!]^4$:

$$\begin{array}{rl} & (\!|\Gamma|\!)^4; (\!|\Delta_0|\!)\cdot m_1{:}(\!|S_1|\!)^4\cdot m_2{:}(\!|S_2|\!)^4\cdot n:!\langle(\!|S_1|\!)^4\rangle; !\langle(\!|S_2|\!)^4\rangle; (\!|S|\!)^4 \vdash n!\langle m_1\rangle.n!\langle m_2\rangle.[\![P']\!]^4 \\ \overset{n!\langle m_1\rangle}{\longmapsto} & (\!|\Gamma|\!)^4; (\!|\Delta_0|\!)\cdot m_2{:}(\!|S_2|\!)^4\cdot n:!\langle(\!|S_2|\!)^4\rangle; (\!|S|\!)^4 \vdash n!\langle m_2\rangle.[\![P']\!]^4 \\ \overset{n!\langle m_2\rangle}{\longmapsto} & (\!|\Gamma|\!)^4; (\!|\Delta_0|\!)\cdot n{:}(\!|S|\!)^4 \vdash [\![P']\!]^4 \\ = & (\!|\Gamma|\!)^4; (\!|\Delta_0\cdot n:S|\!)^4 \vdash [\![P']\!]^4 \end{array}$$

which concludes the proof for this case.

- Case (1(c)) with $P = n!\langle\lambda(x_1,x_2).Q\rangle.P'$ and $\ell_1 = n!\langle\lambda(x_1,x_2).Q\rangle$. By assumption, P is well-typed. We may have:

$$\frac{\Gamma;\emptyset;\Delta_0\cdot n:S\vdash P'\triangleright\diamond \quad \Gamma;\emptyset;\Delta_1\vdash\lambda(x_1,x_2).Q\triangleright(C_1,C_2)\multimap\diamond}{\Gamma;\emptyset;\Delta_0\cdot\Delta_1\cdot n:!\langle(C_1,C_2)\multimap\diamond\rangle;S\vdash n!\langle\lambda(x_1,x_2).Q\rangle.P'\triangleright\diamond}$$

for some Γ, S, C_1, C_2, Δ_0, Δ_1, such that $\Delta = \Delta_0\cdot\Delta_1\cdot n:!\langle(C_1,C_2)\multimap\diamond\rangle;S$. (For simplicity, we consider only the case of a linear function.) We may have the following typed transition:

$$\Gamma;\Delta_0\cdot\Delta_1\cdot n:!\langle(C_1,C_2)\multimap\diamond\rangle;S\vdash n!\langle\lambda(x_1,x_2).Q\rangle.P' \overset{\ell_1}{\longmapsto} \Delta_0\cdot n{:}S\vdash P'$$

The encoding of the source judgment is

$$(\!|\Gamma|\!)^4;\emptyset;(\!|\Delta_0\cdot\Delta_1\cdot n:!\langle(C_1,C_2)\multimap\diamond\rangle;S|\!)^4\vdash[\![n!\langle\lambda(x_1,x_2).Q\rangle.P']\!]^4\triangleright\diamond$$

which, using Fig. 8 , can be equivalently expressed as

$$(\!|\Gamma|\!)^4;\emptyset;(\!|\Delta_0\cdot\Delta_1|\!)\cdot n:!\langle(?(\!|C_1|\!)^4);?(\!|C_2|\!)^4);\mathsf{end})\multimap\diamond\rangle;(\!|S|\!)^4\vdash n!\langle\lambda z.z?(x_1).z?(x_2).[\![Q]\!]^4\rangle.[\![P']\!]^4\triangleright\diamond$$

Now, $\{\!|\ell_1|\!\}^4 = n!\langle\lambda z.z?(x_1).z?(x_2).[\![Q]\!]^4\rangle$. It is immediate to infer the following typed transition for $[\![P]\!]^4 = n!\langle\lambda z.z?(x_1).z?(x_2).[\![Q]\!]^4\rangle.[\![P']\!]^4$:

$$\begin{array}{rl} & (\!|\Gamma|\!)^4; (\!|\Delta_0|\!)\cdot n:!\langle(?(\!|C_1|\!)^4);?(\!|C_2|\!)^4);\mathsf{end})\multimap\diamond\rangle;(\!|S|\!)^4\vdash n!\langle\lambda z.z?(x_1).z?(x_2).[\![Q]\!]^4\rangle.[\![P']\!]^4 \\ \overset{\{\!|\ell_1|\!\}^4}{\longmapsto} & (\!|\Gamma|\!)^4; (\!|\Delta_0|\!)\cdot n:(\!|S|\!)^4,\vdash [\![P']\!]^4 \\ = & (\!|\Gamma|\!)^4; (\!|\Delta_0\cdot n:S|\!)^4\vdash[\![P']\!]^4 \end{array}$$

which concludes the proof for this case.

- Case (2(a)), with $P = n?(x_1,x_2).P'$, $[\![P]\!]^4 = n?(x_1).n?(x_2).[\![P']\!]^4$. We have the following typed transitions for $[\![P]\!]^4$, for some S, S_1, S_2, and Δ:

$$\begin{array}{rl} & (\!|\Gamma|\!)^4; (\!|\Delta|\!)^4\cdot n:?(\!|S_1|\!)^4);?(\!|S_2|\!)^4);(\!|S|\!)^4\cdot\vdash n?(x_1).n?(x_2).[\![P']\!]^4 \\ \overset{n?\langle m_1\rangle}{\longmapsto} & (\!|\Gamma|\!)^4; (\!|\Delta|\!)^4\cdot n:?(\!|S_2|\!)^4);(\!|S|\!)^4\cdot m_1:(\!|S_1|\!)^4\vdash n?(x_2).[\![P']\!]^4\{m_1/x_1\} \\ \overset{n?\langle m_2\rangle}{\longmapsto} & (\!|\Gamma|\!)^4; (\!|\Delta|\!)^4\cdot n:(\!|S|\!)^4\cdot m_1:(\!|S_1|\!)^4\cdot m_2:(\!|S_2|\!)^4\vdash[\![P']\!]^4\{m_1/x_1\}\{m_2/x_2\} = Q \end{array}$$

Observe that the substitution lemma (Lemma **??**(1)) has been used twice. It is then immediate to infer the label for the source transition: $\ell_1 = n?\langle m_1,m_2\rangle$. Indeed, $\{\!|\ell_1|\!\}^4 = n?\langle m_1\rangle, n?\langle m_2\rangle$. Now, in the source term P we can infer the following transition:

$$\Gamma;\Delta\cdot n:?(S_1,S_2);S\vdash n?(x_1,x_2).P' \overset{\ell_1}{\longmapsto} \Delta\cdot n{:}S\cdot m_1:S_1\cdot m_2:S_2\vdash P'\{m_1,m_2/x_1,x_2\}$$

which concludes the proof for this case.

- Case (2(b)), with $P = n!\langle\lambda(x_1,x_2).Q\rangle.P'$, $[\![P]\!]^4 = n!\langle\lambda z.z?(x_1).z?(x_2).[\![Q]\!]^4\rangle.[\![P']\!]^4$. We have the following typed transition, for some S, C_1, C_2, and Δ:

$$\begin{array}{rl} & (\!|\Gamma|\!)^4; (\!|\Delta|\!)^4\cdot n:(\!|!\langle(C_1,C_2)\multimap\diamond\rangle;S|\!)^4\vdash n!\langle\lambda z.z?(x_1).z?(x_2).[\![Q]\!]^4\rangle.[\![P']\!]^4 \\ \overset{\ell_1'}{\longmapsto} & (\!|\Gamma|\!)^4; (\!|\Delta|\!)^4\cdot n:(\!|S|\!)^4\vdash[\![P']\!]^4 = Q \end{array}$$

where $\ell_1' = n!\langle\lambda z.z?(x_1).z?(x_2).[\![Q]\!]^4\rangle$. For simplicity, we consider only the case of linear functions. It is then immediate to infer the label for the source transition: $\ell_1 = n!\langle\lambda(x_1,x_2).Q\rangle$. Now, in the source term P we can infer the following transition:

$$\Gamma;\Delta\cdot n:!\langle(C_1,C_2)\multimap\diamond\rangle;S\vdash n!\langle\lambda x_1,x_2.Q\rangle.P' \overset{\ell_1}{\longmapsto} \Delta\cdot n{:}S\vdash P'$$

which concludes the proof for this case.

□

Proposition B.18 (Full Abstraction. From HOπ^+ to HOπ). *Let P,Q HO$\vec{\pi}$ process with $\Gamma;\emptyset;\Delta_1\vdash P\triangleright\diamond$ and $\Gamma;\emptyset;\Delta_2\vdash Q\triangleright\diamond$. $\Gamma;\Delta_1\vdash P\approx\Delta_2\vdash Q$ if and only if $(\!|\Gamma|\!)^4;(\!|\Delta_1|\!)^4\vdash[\![P]\!]^4\approx(\!|\Delta_2|\!)^4\vdash[\![Q]\!]^4$*

Proof. The proof for both direction is a consequence of Operational Correspondence, Prop. B.17.

Soundness Direction.

We create the closure

$$\mathfrak{R} = \{\Gamma;\Delta_1\vdash P\ ,\ \Delta_2\vdash Q\ \mid\ (\!|\Gamma|\!)^4;(\!|\Delta_1|\!)^4\vdash[\![P]\!]^4\approx(\!|\Delta_2|\!)^4\vdash[\![Q]\!]^4\}$$

It is straightforward to show that $\mathfrak{R}$ is a bisimulation if we follow Part 2 of Prop. B.17.

Completeness Direction.

We create the closure

$$\mathfrak{R} = \{(\!|\Gamma|\!)^4; (\!|\Delta_1|\!)^4 \vdash [\![P]\!]^4 \ , \ (\!|\Delta_2|\!)^4 \vdash [\![Q]\!]^4 \ \mid \ \Gamma; \Delta_1 \vdash P \approx \Delta_2 \vdash Q\}$$

We show that $\mathfrak{R}$ is a bisimulation up to deterministic transitions by following Part 1 of Prop. B.17. □

C. Negative Result

Theorem C.1. There is no encoding $\langle [\![\cdot]\!], (\!|\cdot|\!), \{\!|\cdot|\!\}\rangle : \mathsf{HO}\pi \longrightarrow \mathsf{HO}\pi^{-\mathsf{sh}}$ that enjoys operational correspondence and full abstraction.

Proof. Let $\Gamma_1; \Delta_1 \vdash P_1 \not\approx \Delta_2 \vdash P_2$ with $P = \overline{a}\langle s\rangle.\mathbf{0} \mid a(x).P_1 \mid a(x).P_2$ and let $\Gamma; \emptyset; \Delta \vdash P \triangleright \diamond$. Assume also a encoding $\langle [\![\cdot]\!], (\!|\cdot|\!), \{\!|\cdot|\!\}\rangle : \mathsf{HO}\pi \longrightarrow \mathsf{HO}\pi^{-\mathsf{sh}}$ that enjoys operational correspondence and full abstraction.

From operational correspondence we get that:

$$P \longrightarrow P_1 \mid a(x).P_2 \quad \text{implies} \quad [\![P]\!] \longrightarrow [\![P_1 \mid a(x).P_2]\!]$$
$$P \longrightarrow P_2 \mid a(x).P_1 \quad \text{implies} \quad [\![P]\!] \longrightarrow [\![P_2 \mid a(x).P_1]\!]$$

From the fact that $\Gamma_1; \Delta_1 \vdash P_1 \not\approx \Delta_2 \vdash P_2$ we can derive that

$$\Gamma_1'; \Delta_1' \vdash P_1 \mid a(x).P_2 \not\approx \Delta_2' \vdash P_2 \mid a(x).P_1$$

From Corollary **??** we know that

$$(\!|\Gamma|\!); (\!|\Delta|\!) \vdash [\![P]\!] \approx (\!|\Delta_1'|\!) \vdash [\![P_1 \mid a(x).P_2]\!]$$
$$(\!|\Gamma|\!); (\!|\Delta|\!) \vdash [\![P]\!] \approx (\!|\Delta_2'|\!) \vdash [\![P_2 \mid a(x).P_1]\!]$$

thus

$$(\!|\Gamma|\!); (\!|\Delta_1'|\!) \vdash [\![P_1 \mid a(x).P_2]\!] \approx (\!|\Delta_2'|\!) \vdash [\![P_2 \mid a(x).P_1]\!]$$

From here we conclude that the full abstraction property does not hold, which is a contradiction. □