Logical Methods in Computer Science
Volume 21, Issue 3, 2025, pp. 25:1–25:52
https://lmcs.episciences.org/

Submitted May 07, 2024
Published Sep. 05, 2025

EXTENSIONAL AND NON-EXTENSIONAL FUNCTIONS AS PROCESSES

KEN SAKAYORI [a] AND DAVIDE SANGIORGI [b,c]

[a] The University of Tokyo, Japan

[b] Università di Bologna, Italy

[c] Inria, France

Abstract. Following Milner's seminal paper, the representation of functions as processes has received considerable attention. For pure λ-calculus, the process representations yield (at best) *non-extensional* λ-theories (i.e., β rule holds, whereas η does not).

In the paper, we study how to obtain *extensional* representations, and how to move between extensional and non-extensional representations. Using Internal π, Iπ (a subset of the π-calculus in which all outputs are bound), we develop a refinement of Milner's original encoding of functions as processes that is *parametric* on certain abstract components called *wires*. These are, intuitively, processes whose task is to connect two end-point channels. We show that when a few algebraic properties of wires hold, the encoding yields a λ-theory. Exploiting the symmetries and dualities of Iπ, we isolate three main classes of wires. The first two have a sequential behaviour and are dual of each other; the third has a parallel behaviour and is the dual of itself. We show the adoption of the parallel wires yields an extensional λ-theory; in fact, it yields an equality that coincides with that of Böhm trees with infinite η. In contrast, the other two classes of wires yield non-extensional λ-theories whose equalities are those of the Lévy-Longo and Böhm trees.

1. Introduction

Milner's work [Mil90, Mil92] on the encoding of the pure λ-calculus into the π-calculus is generally considered a landmark paper in the area of semantics and programming languages. The encoding of the λ-calculus is a significant test of expressiveness for the π-calculus. The encoding also gives an interactive semantics to the λ-calculus, which allows one to analyse it using the instruments available in the π-calculus. After Milner's seminal work, a number of encoding variants have been put forward (e.g. [SW01] and references therein) by modifying the target language (often to a subcalculus of the π-calculus) or the encoding itself. The correctness of these encodings is usually supported by the operational correspondence against a certain evaluation strategy of the λ-calculus and by the validity of the β-rule, $(\lambda x.\, M)N = M\{^N/_x\}$. (In this paper, by validity of a λ-calculus rule with respect to a certain process encoding $\{\!|\cdot|\!\}$, we mean that $\{\!|M|\!\} \approx \{\!|N|\!\}$ for all instances $M = N$ of (the congruence closure of) the rule, where $\approx$ is a basic behavioural equivalence for the pure processes, such as ordinary bisimilarity.)

LOGICAL METHODS IN COMPUTER SCIENCE
DOI:10.46298/LMCS-21(3:25)2025

The equality on λ-terms induced by the encoding has also been investigated; in this equality two λ-terms M and N are equal when their images are behaviourally equivalent processes. For Milner's original (call-by-name) encoding, such an equality coincides with the Lévy-Longo tree (LT) equality [San93, San00] (the result is by large independent of the behavioural equivalence adopted for the processes [SX18]). It has also been shown how to recover the Böhm tree (BT) equality [SW01], by modifying Milner's encoding — allowing reductions underneath a λ-abstraction — and selecting divergence-sensitive behavioural equivalences on processes such as must-testing.

Tree structures play a pivotal role in the λ-calculus. For instance, trees allow one to unveil the computational content hidden in a λ-term, with respect to some relevant minimal information. In BTs the information is the head normal forms, whereas in LTs it is the weak head normal forms. BT and LT equalities coincide with the local structures of well-known models of the λ-calculus, such as Plotkin and Scott's P_ω [Plo72, Sco76], and the *free lazy Plotkin-Scott-Engeler models* [Lév76, Eng81, Lon83].

In BTs and LTs, the computational content of a λ-term is unveiled using the β-rule alone. Such structures are sometimes called *non-extensional*, as opposed to the *extensional* structures, in which the β-rule is coupled with the η-rule, $M = \lambda x.\, M\, x$ (for x not free in M). In extensional theories two functions are equated if, whenever applied to the same argument, they yield equal results. A well-known extensional tree-structure are BTs with infinite η, shortly $\mathrm{BT}_{\eta\infty}$s. The equality of $\mathrm{BT}_{\eta\infty}$s, also known as the equational theory $\mathcal{H}^*$, coincides with that of Scott's D_∞ model [Sco76], historically the first model of the untyped λ-calculus. A seminal result by Wadsworth [Wad76] shows that the $\mathrm{BT}_{\eta\infty}$s are intimately related to the head normal forms, as the $\mathrm{BT}_{\eta\infty}$ equality coincides with contextual equivalence in which the head normal forms are the observables.

In representations of functions as processes, extensionality and the η-rule, even in their most basic form, have always appeared out of reach. For instance, in Milner's encoding, x and $\lambda y.\, x\, y$ have quite different behaviours: the former process is a single output particle, whereas the latter has an infinite behaviour and, moreover, the initial action is an input.

The general goal of this paper is to study extensionality in the representation of functions as processes. In particular, we wish to understand if and how one can derive extensional representations, and the difference between extensional and non-extensional representations from a process perspective.

We outline the main technical contributions. We develop a refinement of Milner's original encoding of functions, using Internal π (Iπ), a subcalculus of the π-calculus in which only bound names may be exported. The encoding makes use of certain abstract components called *wires*. These are, intuitively, processes whose task is to connect two end-point channels; and when one of the two end-points is restricted, the wires behave as substitutions. In the encoding, wires are called 'abstract' because their definitions are not made explicit. We show that assuming a few basic algebraic properties of wires (having to do with transitivity of wires and substitution) is sufficient to obtain a λ-theory, i.e. the validity of the β-rule.

We then delve into the impact of the concrete definition of the wires, notably on the equivalence on λ-terms induced by the encoding. In the π-calculus literature, the most common form of wire between two channels a and b is written $!a(u).\,\overline{b}\langle u\rangle$ (or $a(u).\,\overline{b}\langle u\rangle$, if only needed once), and sometimes called a *forwarder* [HY95, Mer01]. In Iπ, free outputs are forbidden and such a wire becomes a recursively-defined process. We call this kind of wires *I-O wires*, because of their 'input before output' behaviour. Exploiting the properties of Iπ, e.g., its symmetries and dualities, we identify two other main kinds of wires: the *O-I wires*,

Table 1: Instances of the abstract encoding

Encoding	Parameter (wires)	Characterises
$\mathcal{A}_{\mathtt{IO}}$	I-O wires	BT
$\mathcal{A}_{\mathtt{P}}$	P wires	$\mathrm{BT}_{\eta\infty}$
$\mathcal{A}_{\mathtt{OI}}$	O-I wires	LT

with an 'output before input' behaviour and which are thus the dual of the I-O wires; and the *P wires*, or *parallel wires*, where input and output can fire concurrently (hence such wires are behaviourally the same as their dual).

We show that moving among these three kinds of wire corresponds to moving among the three above-mentioned tree structures of the λ-calculus, namely BTs, LTs, $\mathrm{BT}_{\eta\infty}$s. Precisely, we obtain BTs when adopting the ordinary I-O wires; LTs when adopting the O-I wires; and $\mathrm{BT}_{\eta\infty}$s when adopting the P wires. This also implies that P wires allow us to validate the η-rule (in fact both η and infinite η). The results are summarised in Table 1, where $\mathcal{A}_{\mathtt{X}}$ is the concrete encoding in which the $\mathtt{X}$ wires are used.

We are not aware of results in the literature that produce an *extensional* λ-theory from a processes model, let alone that derive the $\mathrm{BT}_{\eta\infty}$ equality. We should also stress that the choice of the wire is the *only* modification needed for switching among the three tree structures: the encoding of the λ-calculus is otherwise the same, nor does it change the underlying calculus and its behavioural equivalence (namely, Iπ and bisimilarity).

There are various reasons for using Iπ in our study. The first and most important reason has to do with the symmetries and dualities of Iπ, as hinted above. The second reason is proof techniques: in the paper we use a wealth of proof techniques, ranging from algebraic laws to forms of 'up-to bisimulation' and to unique solutions of equations; not all of them are available in the ordinary π-calculus. The third reason has to do with η-rule. In studies of the expressiveness of Iπ in the literature [Bor98] the encoding of the free-output construct into Iπ resembles an (infinite) η-expansion. The essence of the encoding is the following transformation (which needs to be recursively applied to eliminate all free outputs):

$$\overline{a}\langle p\rangle \mapsto \boldsymbol{\nu} q\ (\overline{a}\langle q\rangle \mid q(\widetilde{y}).\,\overline{p}\langle\widetilde{y}\rangle)\,. \tag{1.1}$$

A free output of p is replaced by a bound output, that is, an output of a freshly created name q (for simplicity, we assume that p is meant to be used only once by the recipient). The transformation requires *localised* calculi [MS04], in which the recipient of a name may only use it in output, and resembles an η-expansion of a variable of the λ-calculus in that, intuitively, direct access to the name p is replaced by access to the function $\lambda\widetilde{y}.\,\overline{p}\langle\widetilde{y}\rangle$.

A possible connection between Iπ and η-expansion may also be found in papers such as [CPT16], where η-expanded proofs (proofs in which the identity rule is only applied to atomic formulas) are related to (session-typed) processes with bound outputs only. Yet, the technical link with our works appears weak because the wires that we use to achieve extensionality (the P wires of Table 1) are behaviourally quite different from the process structures mentioned above.

We derive the encoding into Iπ in two steps. The first step consists, intuitively, in transplanting Milner's encoding into Iπ, by replacing free outputs with bound outputs plus wires, following the idea in (1.1) above. However, (1.1) is only valid in localised calculi, whereas Milner's encoding also requires the *input* capability of names to be transmitted.

Therefore we have to modify the wire in (1.1), essentially inverting the two names p and q. The correctness of the resulting transformation relies on properties about the usage of names that are specific to the representation of functions. The second step adopted to derive the encoding consists of allowing reductions underneath a λ abstraction; that is, implementing a *strong* reduction strategy. This transformation is necessary in order to mimic the computation required to obtain head normal forms.

Encodings of strong reduction strategies have appeared in the literature; they rely on the possibility of encoding non-blocking prefixes (sometimes called *delayed* in the literature) [Abr94, BS94, Fu97, PV98, Mer01, MS04, SW01], i.e., prefixes $\mu : P$ in which actions from P may fire before μ, as long as μ does not bind names of the action. The encodings of non-blocking prefixes in the literature require the names bound in μ to be localised. Here again, the difficulty was to adapt the schema to non-localised names. Similar issues arise within wires, as their definition also requires certain prefixes to be non-blocking.

Structure of the paper Section 2 recalls background material on λ-calculus and Iπ. Section 3 introduces wires and permeable prefixes. In Section 4, we present the abstract encoding, using the abstract wires, and the assumptions we make on wires; we then verify that such assumptions are sufficient to obtain a λ-theory. Section 5 defines an optimised abstract encoding, which will be useful for later proofs. In Section 6, we introduce the three classes of concrete wires, and show that they satisfy the required assumptions for wires. In Section 7, we pick the I-O wires and O-I wires, and prove full abstraction for LTs and BTs. In Section 8, we do the same for the P wires and $\mathrm{BT}_{\eta\infty}$s. Section 9 discusses further related work and possible future developments. For readability, some proofs are only given in the Appendices.

2. Background

A tilde represents a tuple. The i-th element of a tuple $\widetilde{P}$ is referred to as P_i. All notations are extended to tuples componentwise.

2.1. **The λ-calculus.** We let x and y range over the set of λ-calculus variables. The set Λ of λ-terms is defined by the grammar

$$M ::= \; x \;\mid\; \lambda x.\, M \;\mid\; M_1\, M_2\,.$$

Free variables, closed terms, substitution, α-conversion etc. are defined as usual [Bar84]; the set of free variables of M is $\mathsf{fv}(M)$. Here and in the rest of the paper (including when reasoning about processes), we adopt the usual 'Barendregt convention'. This will allow us to assume freshness of bound variables and names whenever needed. We group brackets on the left; therefore MNL is $(MN)L$. We abbreviate $\lambda x_1.\cdots.\lambda x_n.\, M$ as $\lambda x_1 \cdots x_n.\, M$, or $\lambda\widetilde{x}.\, M$.

A number of reduction relations are mentioned in this paper. The (standard) *β-reduction* relation $M \to N$ is the relation on λ-terms induced by the following rules:

$$[\beta]\ \frac{}{(\lambda x.\, M)\, N \to M\{^{N}/_{x}\}} \qquad [\mu]\ \frac{N \to N'}{M\, N \to M\, N'}$$

$$[\nu]\ \frac{M \to M'}{M\, N \to M'\, N} \qquad [\xi]\ \frac{M \to M'}{\lambda x.\, M \to \lambda x.\, M'}$$

The (weak) *call-by-name reduction* relation uses only the β and ν rules, whereas *strong call-by-name*, written $\to_{\mathtt{sn}}$, also has ξ; the *head reduction*, written $\to_{\mathtt{h}}$, is a deterministic variant of $\to_{\mathtt{sn}}$ in which the redex contracted is the head one, i.e., $(\lambda y.\, M_0)\, M_1$ of $\lambda\widetilde{x}.\, (\lambda y.\, M_0)\, M_1 \cdots M_n$.

Head normal forms are of the form $\lambda\widetilde{x}.\, y\, \widetilde{M}$. As usual, we use a double arrow to indicate the reflexive and transitive closure of a reduction relation, as in $\Longrightarrow$ and $\Longrightarrow_{\mathtt{h}}$. A term M *has a head normal form* N if $M \Longrightarrow_{\mathtt{h}} N$ and N is the (unique) head normal form. Terms that do not have a head normal form are called *unsolvable*. An unsolvable M has an *order of unsolvability* n, if n is the largest natural number such that $M \Longrightarrow_{\mathtt{h}} \lambda x_1 \dots x_n.\, M'$, for $n \geq 0$, and some $x_1, \dots, x_n, M'$. If there is no such largest number, then M is *of order* ω. For instance, if Δ is $\lambda x.\, x\, x$, and Ω is $\Delta\, \Delta$, and Ξ is $(\lambda xy.\, x\, x)\, (\lambda xy.\, x\, x)$, then we have $\Omega \rightarrow_{\mathtt{h}} \Omega$, therefore Ω is an unsolvable of order 0, and $\lambda x.\, \Omega$ is of order 1; whereas $\Xi \rightarrow_{\mathtt{h}} \lambda y.\, \Xi$, indeed for any n we have $\Xi \Longrightarrow_{\mathtt{h}} \lambda y_1 \dots y_n.\, \Xi$, therefore Ξ is an unsolvable of order ω.

We recall the (informal) definitions of Lévy-Longo trees and Böhm trees, and of Böhm trees up-to infinite η-expansion. The *Lévy-Longo tree* of M is the labelled tree, $\mathrm{LT}(M)$, defined coinductively as follows:

(1) $\mathrm{LT}(M) = \top$ if M is an unsolvable of order ω;
(2) $\mathrm{LT}(M) = \lambda x_1 \dots x_n.\, \bot$ if M is an unsolvable of order $n < \omega$;
(3) $\mathrm{LT}(M)$ is the tree with $\lambda\widetilde{x}.\, y$ as the root and $\mathrm{LT}(M_1) \dots \mathrm{LT}(M_n)$ as the children, if M has head normal form $\lambda\widetilde{x}.\, y\, M_1 \cdots M_n$ with $n \geq 0$.

The definition of Böhm trees (BTs) is obtained from that of LTs using BT in place of LT, and demanding that $\mathrm{BT}(M) = \bot$ whenever M is unsolvable (in place of clauses (1) and (2)).

An η-expansion of a BT, whose root is $\lambda\widetilde{x}.\, y$ and children are $\mathrm{BT}(M_1), \dots \mathrm{BT}(M_n)$, is given by a tree whose root is $\lambda\widetilde{x}z.\, y$ and children are $\mathrm{BT}(M_1), \dots \mathrm{BT}(M_n), z$. Intuitively, an infinite η-expansion of a BT is obtained by allowing this expansion at each step of the clause (3). Thus, an (informal) coinductive definition of the *Böhm trees up-to infinite η-expansion* of M, $\mathrm{BT}_{\eta\infty}(M)$, is given as follows:

(1) $\mathrm{BT}_{\eta\infty}(M) = \bot$ if M is unsolvable, and
(2) if $M \Longrightarrow_{\mathtt{h}} \lambda x_1 \dots x_m.\, y\, M_1 \cdots M_n$ for $m \geq 0$ and $n \geq 0$, then
   (a) $\mathrm{BT}_{\eta\infty}(M) = \mathrm{BT}_{\eta\infty}(\lambda x_1 \dots x_{m-1}.\, y\, M_1 \cdots M_{n-1})$, if $x_m \notin \mathsf{fv}(y\, M_1 \cdots M_{n-1})$ and $\mathrm{BT}_{\eta\infty}(M_n) = \mathrm{BT}_{\eta\infty}(x_m)$; and otherwise
   (b) $\mathrm{BT}_{\eta\infty}(M) =$

$$\begin{array}{c} \lambda x_1 \dots x_m.\, y \\ \diagup \quad \diagdown \\ \mathrm{BT}_{\eta\infty}(M_1) \cdots \mathrm{BT}_{\eta\infty}(M_n) \end{array}$$

In the equality induced by the above trees, two terms are related if their trees are the same (as usual modulo α-conversion). These equalities may be defined coinductively as forms of bisimilarity on λ-terms ([San93]), in the expected manner. We only present *$BT_{\eta\infty}$-bisimilarity*, because it is the most delicate one and also because it will be used in a few important proofs. In such a bisimilarity, first introduced by Lassen [Las99], a (finite) η-expansion is allowed at each step of the bisimulation game.

Definition 2.1 [Las99]**.** A relation $\mathcal{R}$ on λ-terms is a *$BT_{\eta\infty}$-bisimulation* if, whenever $M\, \mathcal{R}\, N$, either one of the following holds:

(1) M and N are unsolvable
(2) $M \Longrightarrow_{\mathtt{h}} \lambda x_1 \dots x_{l+m}.\, y\, M_1 \cdots M_{n+m}$ and $N \Longrightarrow_{\mathtt{h}} \lambda x_1 \dots x_l.\, y\, N_1 \cdots N_n$, where the variables $x_{l+1}, \dots, x_{l+m}$ are not free in $y\, N_1 \cdots N_n$, and $M_i\, \mathcal{R}\, N_i$ for $1 \leq i \leq n$, and also $M_{n+j}\, \mathcal{R}\, x_{l+j}$ for $1 \leq j \leq m$
(3) the symmetric case, where N reduces to a head normal form with more leading λs.

The largest $\mathrm{BT}_{\eta\infty}$-bisimulation is called *$BT_{\eta\infty}$-bisimilarity*. We also write $\mathrm{BT}_{\eta\infty}(M) = \mathrm{BT}_{\eta\infty}(N)$ when M and N are $\mathrm{BT}_{\eta\infty}$-bisimilar.

Two terms are indeed $\mathrm{BT}_{\eta\infty}$-bisimilar precisely when they have the same Böhm tree up-to infinite η-expansion.

Example 2.2. We provide some examples of Böhm trees and Lévy-Longo trees, using the terms Δ, Ω, and Ξ introduced earlier.

$$\mathrm{BT}(\Delta) = \begin{array}{c}\lambda x.\, x\\ |\\ x\end{array} \qquad \mathrm{BT}(\Omega) = \bot \quad \mathrm{BT}(\lambda x.\, \Omega) = \bot \qquad \mathrm{BT}(\Xi) = \bot$$

$$\mathrm{LT}(\Delta) = \begin{array}{c}\lambda x.\, x\\ |\\ x\end{array} \qquad \mathrm{LT}(\Omega) = \bot \quad \mathrm{LT}(\lambda x.\, \Omega) = \lambda x.\, \bot \quad \mathrm{LT}(\Xi) = \top$$

For all the terms M used in this example, we have $\mathrm{BT}(M) = \mathrm{BT}_{\eta\infty}(M)$.

Example 2.3. Let J be a term such that $J\, z \Longrightarrow_{\mathtt{h}} \lambda y.\, z\, (J y)$; for instance, we can take J to be $Y\, (\lambda f x y.\, x\, (f\, y))$ using the fixpoint combinator $Y = \lambda f.\, (\lambda x.\, f\, (x\, x))\, (\lambda x.\, f\, (x\, x))$. Intuitively, the term $J\, z$ can be considered as the 'limit of the sequence of η-expansions'

$$z \to_\eta \lambda z_1.\, z\, z_1 \to_\eta \lambda z_1.\, z\, (\lambda z_2.\, z_1\, z_2) \to_\eta \cdots$$

In other words, $J\, z$ is a term that represents an 'infinite η-expansion' of z. The terms z and $J\, z$ have different Böhm trees:

$$\mathrm{BT}(J\, z) = \begin{array}{c}\lambda z_1.\, z\\ |\\ \lambda z_2.\, z_1\\ |\\ \lambda z_3.\, z_2\\ |\\ \vdots\end{array} \qquad \mathrm{BT}(z) = z.$$

However, $\mathrm{BT}_{\eta\infty}(J\, z) = \mathrm{BT}_{\eta\infty}(z)$ as the two terms can be equated using an infinite form of η-expansion. To show that they are $\mathrm{BT}_{\eta\infty}$-bisimilar, let $\mathcal{R}$ be the set of pairs of the form $(x, J\, x)$ and $\eqsim_{B\eta}$ be $\mathrm{BT}_{\eta\infty}$-bisimilarity. Then $\mathcal{R} \cup \eqsim_{B\eta}$ is a $\mathrm{BT}_{\eta\infty}$-bisimulation. To see this, assume that $M (\mathcal{R} \cup \eqsim_{B\eta}) N$. We only consider the case where M is a variable, say x, and N is $J\, x$. We have $x \Longrightarrow_{\mathtt{h}} x$ and $J x \Longrightarrow_{\mathtt{h}} \lambda y.\, x\, (J\, y)$. This closes the case because $x \eqsim_{B\eta} x$ and $J\, y\ \mathcal{R}\ y$.

2.2. **Internal π-calculus.** In all encodings we consider, the encoding of a λ-term is parametric on a name, i.e., it is a function from names to π-calculus processes. We also need parametric processes (over one or several names) for writing recursive process definitions and equations. We call such parametric processes *abstractions*. The actual instantiation of the parameters of an abstraction F is done via the *application* construct $F\langle \widetilde{a} \rangle$. We use P, Q for processes, F for abstractions. Processes and abstractions form the set of π-*agents* (or simply *agents*), ranged over by A. Small letters $a, b, \ldots, x, y, \ldots$ range over the infinite set of names. The grammar of the internal π-calculus (Iπ) is thus:

$$\begin{array}{lcll} A & ::= & P \mid F & \text{(agents)} \\ P & ::= & \mathbf{0} \mid a(\widetilde{b}).\, P \mid \overline{a}(\widetilde{b}).\, P \mid \boldsymbol{\nu} a\, P & \text{(processes)} \\ & & \quad \mid P_1 \mid P_2 \mid\ !a(\widetilde{b}).\, P \mid F\langle \widetilde{a} \rangle & \\ F & ::= & (\widetilde{a})\ P \mid \mathtt{K} & \text{(abstractions)} \end{array}$$

The operators used have the usual meanings. Here, $\overline{a}(\widetilde{b})$ is a *bound output*, which may be thought of as an abbreviation $\boldsymbol{\nu}\widetilde{b}\,\overline{a}\langle\widetilde{b}\rangle$ using the syntax of the standard π-caclulus. Having only bound output constructs is a key characteristic of Iπ, which poses a symmetry between input and output constructs. In prefixes $a(\widetilde{b})$ and $\overline{a}(\widetilde{b})$, we call a the *subject*. We often abbreviate $\boldsymbol{\nu}a\,\boldsymbol{\nu}b\,P$ as $(\boldsymbol{\nu}a, b)\,P$. Prefixes, restriction, and abstraction are binders and give rise in the expected way to the definition of *free names*, *bound names*, and *names* of an agent, respectively indicated with $\mathsf{fn}(-)$, $\mathsf{bn}(-)$, and $\mathsf{n}(-)$, as well as that of α-conversion. An agent is *name-closed* if it does not contain free names. In the grammar, $\mathtt{K}$ is a *constant*, used to write recursive definitions. Each constant $\mathtt{K}$ has a defining equation of the form $\mathtt{K} \stackrel{\text{def}}{=} (\widetilde{x})\ P$, where $(\widetilde{x})\ P$ is name-closed; $\widetilde{x}$ are the formal parameters of the constant (replaced by the actual parameters whenever the constant is used). Replication could be avoided in the syntax since it can be encoded with recursion. However its semantics is simple, and it is a useful construct for encodings; thus we chose to include it in the grammar.

For convenience, we set some conventions. An application redex $((\widetilde{x})P)\langle\widetilde{a}\rangle$ can be normalised as $P\{\widetilde{x}/\widetilde{a}\}$. An agent is *normalised* if all such application redexes have been contracted. When reasoning on behaviours it is useful to assume that all expressions are normalised, in the above sense. Thus in the remainder of the paper *we identify an agent with its normalised expression.* As in the λ-calculus, following the usual Barendregt convention we identify processes or actions which only differ on the choice of the bound names. The symbol $=$ indicates syntactic identity; in the case of terms or objects of the calculi, it will mean 'syntactic identity modulo α-conversion'.

Since the calculus is polyadic, we assume a *sorting system* [Mil93] to avoid disagreements in the arities of the tuples of names carried by a given name and in applications of abstractions. In Milner's encoding (written in the polyadic π-calculus) as well as in all encodings in the paper, there are only two sorts of names: *location names*, and *variable names*. Location names carry a pair of a variable name and a location name; variable names carry a single location name. Using $p, q, r, \ldots$ for location names, and $x, y, z, \ldots$ for variable names, the forms of the possible prefixes are:

$$p(x, q).\,P \ \mid\ x(p).\,P \ \mid\ \overline{p}(x, q).\,P \ \mid\ \overline{x}(p).\,P$$

This sorting will be maintained throughout the paper. Hence process transformations and algebraic laws will be given with reference to such a sorting.

The operational semantics of Iπ is standard [SW01, Section 7.5], and reported in Figure 1. Transitions are of the form $P \xrightarrow{\mu} P'$, where the set of *actions* is given by

$$\mu ::= a(\widetilde{b}) \ \mid\ \overline{a}(\widetilde{b}) \ \mid\ \tau$$

and bound names of μ are *fresh*, i.e., they do not appear free in P. The meaning of actions is the usual one: $a(\widetilde{b})$ and $\overline{a}(\widetilde{b})$ are bound input and outputs, respectively, and τ is the internal action. The co-action $\bar{\mu}$ of μ used in the rule $\mathtt{com}$ is defined by $\overline{a(\widetilde{b})} \stackrel{\text{def}}{=} \overline{a}(\widetilde{b})$ and $\overline{\overline{a}(\widetilde{b})} \stackrel{\text{def}}{=} a(\widetilde{b})$. There is only one rule for communication and one rule for restriction, which makes the LTS simpler than that of the π-calculus. This is because we do not need to care about the difference between free and bound outputs as all the outputs in Iπ are bound. Since we are using the Barendregt convention, in $\mathtt{par}$, we have an implicit side condition $\mathsf{bn}(\mu) \cap \mathsf{fn}(Q) = \emptyset$. The $(\widetilde{y})\,Q = (\widetilde{x})\,P$ in the premise of rule $\mathtt{com}$ merely means that the two

$$[\texttt{pre}]\ \frac{}{\mu.\,P \xrightarrow{\mu} P} \qquad [\texttt{par}]\ \frac{P \xrightarrow{\mu} P'}{P \mid Q \xrightarrow{\mu} P' \mid Q}$$

$$[\texttt{res}]\ \frac{P \xrightarrow{\mu} P' \qquad x \notin \mathsf{n}(\mu)}{\boldsymbol{\nu} x\, P \xrightarrow{\mu} \boldsymbol{\nu} x\, P'} \qquad [\texttt{com}]\ \frac{P \xrightarrow{\mu} P' \qquad Q \xrightarrow{\overline{\mu}} Q' \qquad \mu \neq \tau,\ \widetilde{x} = \mathsf{bn}(\mu)}{P \mid Q \xrightarrow{\tau} \boldsymbol{\nu}\widetilde{x}\,(P' \mid Q')}$$

$$[\texttt{rep}]\ \frac{}{!\mu.\,P \xrightarrow{\mu} P' \mid\, !\mu.\,P} \qquad [\texttt{con}]\ \frac{P \xrightarrow{\mu} P' \qquad \mathtt{K} \stackrel{\text{def}}{=} (\widetilde{y})\,Q \qquad (\widetilde{y})\,Q = (\widetilde{x})\,P}{\mathtt{K}\langle\widetilde{x}\rangle \xrightarrow{\mu} P'}$$

Figure 1: The standard LTS for Iπ.

agents are equal up-to α-conversion. As usual, we write $\Rightarrow$ for the reflexive transitive closure of $\xrightarrow{\tau}$, $\stackrel{\mu}{\Longrightarrow}$ for $\Rightarrow\xrightarrow{\mu}\Rightarrow$, and $\stackrel{\hat{\mu}}{\Longrightarrow}$ is $\stackrel{\mu}{\Longrightarrow}$ for $\mu \neq \tau$ and $\Rightarrow$ otherwise.

The reference behavioural equivalence for Iπ is (weak) bisimilarity. It is known that, in Iπ, bisimilarity coincides with barbed congruence (for processes that are image-finite up to weak bisimilarity; all the agents obtained as encodings of λ-terms will be image-finite up to weak bisimilarity).

Definition 2.4 (Bisimilarity)**.** A symmetric relation $\mathcal{R}$ over processes is a *(weak) bisimulation*, if, whenever $P\ \mathcal{R}\ Q$,

- $P \xrightarrow{\mu} P'$ implies $Q \stackrel{\hat{\mu}}{\Longrightarrow} Q'$ for some Q' such that $P'\ \mathcal{R}\ Q'$.

Processes P and Q are *(weakly) bisimilar*, written $P \approx Q$ if there exists a bisimulation $\mathcal{R}$ such that $P\ \mathcal{R}\ Q$.

In a few places we will also use *strong bisimilarity*, written $\sim$; this is defined analogous to $\approx$, but Q must respond with a (strong) transition $Q \xrightarrow{\mu} Q'$.

We also use the *expansion* preorder, written $\lesssim$, an asymmetric variant of $\approx$ in which, intuitively, $P \lesssim Q$ holds if $P \approx Q$ but also Q has at least as many τ-moves as P.

Definition 2.5 (Expansion)**.** A relation $\mathcal{R}$ over processes is an *expansion* if $P\ \mathcal{R}\ Q$ implies:

(1) Whenever $P \xrightarrow{\mu} P'$, there exists Q' such that $Q \stackrel{\mu}{\Longrightarrow} Q'$ and $P'\ \mathcal{R}\ Q'$;
(2) whenever $Q \xrightarrow{\mu} Q'$, there exists P' such that $P \xrightarrow{\widehat{\mu}} P'$ and $P'\ \mathcal{R}\ Q'$.

Here $\xrightarrow{\widehat{\mu}}$ is the strong version of $\stackrel{\hat{\mu}}{\Longrightarrow}$, that is, $\xrightarrow{\widehat{\mu}}$ is $\xrightarrow{\mu}$ if $\mu \neq \tau$ and is $=$ or $\xrightarrow{\tau}$ if $\mu = \tau$. Recall that $\stackrel{\tau}{\Longrightarrow}$ means at least one τ-action. Therefore, when $P \xrightarrow{\tau} P'$, Q must respond using at least one τ-action whereas, when $Q \xrightarrow{\tau} Q'$, P must respond using at most one τ-action. We say that Q *expands* P, written $P \lesssim Q$, if $P\ \mathcal{R}\ Q$, for some expansion $\mathcal{R}$.

Finally, we sometimes use *structural congruence*, when we need to emphasize that two processes are the same modulo some structural rewriting of their syntax.

Definition 2.6 (Structural congruence)**.** The *structural congruence* $\equiv$ is the smallest congruence relation over processes that includes the α-equivalence and the following equivalence:

$$P \mid \mathbf{0} \equiv P \quad P \mid Q \equiv Q \mid P \quad (P \mid Q) \mid R \equiv P \mid (Q \mid R)$$
$$\boldsymbol{\nu} x\, \boldsymbol{\nu} y\, P \equiv \boldsymbol{\nu} y\, \boldsymbol{\nu} x\, P \quad (x \neq y)$$
$$\boldsymbol{\nu} x\, (P \mid Q) \equiv \boldsymbol{\nu} x\, P \mid Q \quad (x \notin \mathsf{fv}(Q)).$$

We summarise the inclusion order between the process relations that we use:

$$\equiv \subsetneq \sim \subsetneq \lesssim \subsetneq \approx.$$

In Iπ, *all the relations above are (pre)congruences* [San96b]. All behavioural relations are extended to abstractions by requiring ground instantiation of the parameters. For instance, $(x)P \approx (x)Q$ if $P \approx Q$. Appendix A summarises notations used in the paper (term relations, encodings, etc.).

2.2.1. *Proof techniques.* Here we review known proof-techniques for Iπ that we will use in this paper. These are well-known algebraic laws, notably laws for private replications, up-techniques for bisimilarity, and unique solutions of equations.

Algebraic laws. We will often use a group of laws about private replication. That is, laws for processes of the form $\boldsymbol{\nu} a\, (P \mid !a(\widetilde{b}).\, Q)$, where a appears free only in output position of P and Q. Since the process $!a(\widetilde{b}).\, Q$ is replicated and private it can be distributed or pushed inside prefixes. These laws are known as the replication theorems [SW01, Section 2.2.3].

Lemma 2.7. *Suppose x occurs free only in output subject position of P, Q and R. Then we have*

(1) $\boldsymbol{\nu} x\, (Q \mid R \mid !x(p).\, P) \sim \boldsymbol{\nu} x\, (Q \mid !x(p).\, P) \mid \boldsymbol{\nu} x\, (R \mid !x(p).\, P)$;
(2) $\boldsymbol{\nu} x\, (\pi.\, Q \mid !x(p).\, P) \sim \pi.\, \boldsymbol{\nu} x\, (Q \mid !x(p).\, P)$, *if π is a non-replicated prefix, subject of π is not x and* $\mathsf{bn}(\pi) \cap \mathsf{fn}(!x(p).\, P) = \emptyset$;
(3) $\boldsymbol{\nu} x\, (!y(q).\, Q \mid !x(p).\, P) \sim !y(q).\, \boldsymbol{\nu} x\, (Q \mid !x(p).\, P)$;
(4) $\boldsymbol{\nu} x\, (Q \mid !x(p).\, P) \sim Q$, *if* $x \notin \mathsf{fn}(Q)$;
(5) $\boldsymbol{\nu} x\, (\overline{x}(p).\, Q \mid !x(p).\, P) \gtrsim \boldsymbol{\nu} x\, \boldsymbol{\nu} p\, (Q \mid P \mid !x(p).\, P)$.

We often call the law (4) the 'garbage collection law'.

Up-to techniques. Our main up-to technique will be *up-to context and expansion* [San96a], which admits the use of contexts and of behavioural equivalences such as expansion to achieve the closure of a relation in the bisimulation game.

Definition 2.8 (Bisimulation up-to context and $\lesssim$)**.** A symmetric relation $\mathcal{R}$ on Iπ-processes is a *bisimulation up-to context and up-to $\lesssim$ if $P\, \mathcal{R}\, Q$ and $P \xrightarrow{\mu} P''$ imply that there are a (possibly multi-hole) context C and processes $\widetilde{P}'$ and $\widetilde{Q}'$ such that $P'' \gtrsim C[\widetilde{P}']$, $Q \stackrel{\widehat{\mu}}{\Longrightarrow} \gtrsim C[\widetilde{Q}']$ and $\widetilde{P}'\, \mathcal{R}\, \widetilde{Q}'$. Here $\widetilde{P}'\, \mathcal{R}\, \widetilde{Q}'$ means $P'_i\, \mathcal{R}\, Q'_i$ for each component.*

A special instance of this technique is the *up-to expansion* technique where the common context C is taken as the empty-context.

Theorem 2.9. *If $\mathcal{R}$ is a bisimulation up-to context and expansion then $\mathcal{R} \subseteq \approx$.*

We refer the readers to [San96a] for the proof. (Adapting the proof to Iπ and arbitrary multi-hole contexts is straightforward.) We also use the technique of *expansion* up-to context and $\lesssim$, which is defined analogously to Definition 2.8, as a proof technique to prove expansion results (the main difference is that one now requires $P'' \lesssim C[\widetilde{P'}]$).

Unique solution of equations. We briefly recall the 'unique solution of equations' technique [DHS19]. *Equation variables* X, Y, Z are used to write equations. The body of an equation is a name-closed abstraction possibly containing equation variables (that is, applications can also be of the form $X\langle\widetilde{a}\rangle$). We use E to range over expression bodies; and $\mathcal{E}$ to range over systems of equations, defined as follows. In all the definitions, the indexing set I can be infinite.

Definition 2.10. Assume that, for each i of a countable indexing set I, we have a variable X_i, and an expression E_i, possibly containing variables. Then $\{X_i = E_i\}_{i\in I}$ (sometimes written $\widetilde{X} = \widetilde{E}$) is a *system of equations*. (There is one equation for each variable X_i.) A system of equations is *guarded* if each occurrence of a variable in the body of an equation is underneath a prefix.

We write $E[\widetilde{F}]$ for the abstraction obtained by replacing in E each occurrence of the variable X_i with the abstraction F_i. This is a syntactic replacement, with instantiation of the parameters: e.g., replacing X with $(\widetilde{x})P$ in $X\langle\widetilde{a}\rangle$ amounts to replacing $X\langle\widetilde{a}\rangle$ with the process $P\{\widetilde{a}/\widetilde{x}\}$.

Definition 2.11. Suppose $\{X_i = E_i\}_{i\in I}$ is a system of equations. We say that:

- $\widetilde{F}$ is a *solution of the system of equations for* $\approx$ if for each i it holds that $F_i \approx E_i[\widetilde{F}]$.
- The system has *a unique solution for* $\approx$ if whenever $\widetilde{F}$ and $\widetilde{G}$ are both solutions for $\approx$, we have $\widetilde{F} \approx \widetilde{G}$.

Definition 2.12 (Syntactic solutions)**.** The *syntactic solutions* of a system of equations $\{X_i = E_i\}_{i\in I}$ are the recursively defined constants $\mathtt{K}_{\widetilde{E},i} \stackrel{\text{def}}{=} E_i[\widetilde{\mathtt{K}_{\widetilde{E}}}]$, for $i \in I$.

The syntactic solutions of a system of equations are indeed solutions of it. The unique-solution technique relies on an analysis of divergences. A process P *diverges* if it can perform an infinite sequence of internal moves, possibly after some visible ones (i.e., actions different from τ). Formally, this holds if there are processes P_i, $i \geq 0$, and some n such that $P = P_0 \xrightarrow{\mu_0} P_1 \xrightarrow{\mu_1} P_2 \xrightarrow{\mu_2} \ldots$ and for all $i > n$, $\mu_i = \tau$. We call a *divergence of* P the sequence of transitions $\left(P_i \xrightarrow{\mu_i} P_{i+1}\right)_{i\geq 0}$. An abstraction F has a divergence if the process $F\langle\widetilde{a}\rangle$ has a divergence, where $\widetilde{a}$ are fresh names.

Theorem 2.13 [DHS22]**.** *A guarded system of equations whose syntactic solutions are agents with no divergences has a unique solution for* $\approx$*.*

3. Wires and Permeable Prefixes

We introduce the abstract notion of *wire* process; and, as a syntactic sugar, the process constructs for *permeable prefixes*. Wires and permeable prefixes will play a central role in the technical development in the following sections.

Wires. We use the notation $a \leftrightarrow \bar{b}$ for an *abstract wire*; this is, intuitively, a special process whose purpose is to connect the output-end of a with the input-end of b (thus $a \leftrightarrow \bar{b}$ itself will use a in input and b in output). We call such wires 'abstract' because we will not give a definition for them. We only state (Section 4) some behavioural properties that are expected to hold, and that have mainly to do with substitutions; approximately:

(1) if P uses b only in input, then $\boldsymbol{\nu} b\,(a \leftrightarrow \bar{b} \mid P) \gtrsim P\{a/b\}$

(2) dually, if P uses a only in output, then $\boldsymbol{\nu} a\,(a \leftrightarrow \bar{b} \mid P) \gtrsim P\{b/a\}$

Further conditions will however be needed on P for such properties to hold (e.g., in (1), the input at b in P should be 'at the top-level', and in (2), the outputs at a in P should be 'asynchronous'.) Special cases of (1) and (2) are forms of transitivity for wires, with the common name restricted:

(3) $\boldsymbol{\nu} b\,(a \leftrightarrow \bar{b} \mid b \leftrightarrow \bar{c}) \gtrsim (b \leftrightarrow \bar{c})\{a/b\} = (a \leftrightarrow \bar{b})\{c/b\} = a \leftrightarrow \bar{c}$.

When (1) holds we say that P is *I-respectful with respect to* $a \leftrightarrow \bar{b}$; similarly when (2) holds we say that P is *O-respectful with respect to* $a \leftrightarrow \bar{b}$. When (3) holds, for any a, b, c of the same sort, we say that *wires are transitive*.

As we have two sorts of names in the paper (location names and variable names), we will correspondingly deal with two sorts of wires, *location wires* and *variable wires*. In fact, location wires will be the key structures. Thus when, in Section 6, we consider *concrete* instantiations of the location wires, the corresponding definitions of the variable wires will be adjusted so to maintain the expected properties of the location wires.

Permeable prefixes. We write $a(\widetilde{b})\colon P$ and $\bar{a}(\widetilde{b})\colon P$ for a *permeable input* and a *permeable output*. Intuitively, a permeable prefix only blocks actions involving the bound names of the prefix. For instance, a permeable input $a(\widetilde{b})\colon P$, as an ordinary input, is capable of producing action $a(\widetilde{b})$ thus yielding the derivative P. However, in contrast with the ordinary input, in $a(\widetilde{b})\colon P$ the process P is active and running, and can thus interact with the outside environment as well as with the prefix $a(\widetilde{b})\colon P$ (though such a form of interaction will not occur in processes encoding of λ-terms); the only constraint is that the actions from P involving the bound names $\widetilde{b}$ cannot fire for as long as the top prefix $a(\widetilde{b})$ is not consumed.

Given the two sorts of names that will be used in the paper, the possible forms of permeable prefixes are:

$$p(x,q)\colon P \;\;\big|\;\; \bar{p}(x,q)\colon P \;\;\big|\;\; x(p)\colon P \;\;\big|\;\; \bar{x}(p)\colon P$$

Moreover, it will always be the case that, in a prefix $p(x,q)\colon P$, process P uses x only in output and q only once in input; and conversely, P uses x only in output and q only once in output in $\bar{p}(x,q)\colon P$. Similarly, in $x(p)\colon P$, process P uses p only once in input; and conversely, in $\bar{x}(p)\colon P$, name p is used only once in output position in P.

We stress that permeable prefixes should be taken as syntactic sugar; formally they are defined from ordinary prefixes and wires as follows

$$\begin{aligned}
p(x,q)\colon P &\stackrel{\text{def}}{=} (\boldsymbol{\nu} x,q)\;\big(p(x',q').\,(x \leftrightarrow \bar{x}' \mid q' \leftrightarrow \bar{q})\mid P\big)\\
x(p)\colon P &\stackrel{\text{def}}{=} \boldsymbol{\nu} p\;\big(x(p').\,p' \leftrightarrow \bar{p} \mid P\big)\\
\bar{p}(x,q)\colon P &\stackrel{\text{def}}{=} (\boldsymbol{\nu} x,q)\;\big(\bar{p}(x',q').\,(x' \leftrightarrow \bar{x} \mid q \leftrightarrow \bar{q}')\mid P\big)\\
\bar{x}(p)\colon P &\stackrel{\text{def}}{=} \boldsymbol{\nu} p\;\big(\bar{x}(p').\,p \leftrightarrow \bar{p}' \mid P\big)
\end{aligned}$$

Such definitions thus depend on the concrete forms of wires adopted. The definitions behave as intended only when the processes underneath the permeable prefixes are respectful. For example, we have

$$p(x,q)\colon P \xrightarrow{p(x,q)} \equiv (\boldsymbol{\nu} x', q')\ \left(x' \leftrightarrow \bar{x} \mid q \leftrightarrow \bar{q}' \mid P\{x', q'/x, q\}\right) \gtrsim P$$

if P is I-respectful with respect to $q' \leftrightarrow \bar{q}$ and O-respectful with respect to $x \leftrightarrow \bar{x}'$, for fresh q' and x'.

Later, when the abstract wires will be instantiated to concrete wires, we will study properties of I-respectfulness and O-respectfulness, as well as, correspondingly, properties of permeable prefixes, in the setting of encodings of functions.

We end this section by introducing some algebraic laws for permeable prefixes. These laws allow us to avoid desugaring while proving the properties of the encodings.

Lemma 3.1. *Suppose that wires, which are used to define the permeable prefixes, are transitive.*

(1) $\boldsymbol{\nu} p\ (p(x,q)\colon P \mid \bar{p}(x,q)\colon Q) \gtrsim (\boldsymbol{\nu} x, q)\ (P \mid Q)$ *if either P is I-respectful with $q \leftrightarrow \bar{q}'$ and O-respectful with $x' \leftrightarrow \bar{x}$ or P is O-respectful with $q \leftrightarrow \bar{q}'$ and I-respectful with $x' \leftrightarrow \bar{x}$.*

(2) $\boldsymbol{\nu} x\ (!x(p).\, P \mid \bar{x}(p)\colon Q \mid R) \gtrsim \boldsymbol{\nu} x\ (!x(p).\, P \mid \boldsymbol{\nu} p\ (P \mid Q) \mid R)$ *if $p \notin \mathsf{fn}(R)$, x does not appear in an input subject position of P, Q, R and either P is I-respectful with $p \leftrightarrow \bar{p}'$ or Q is O-respectful with $p' \leftrightarrow \bar{p}$.*

Lemma 3.2. *The following structural laws hold.*

(1) *If $a \neq b$ and $a \notin \widetilde{c}$ then $\boldsymbol{\nu} a\ b(\widetilde{c})\colon P \equiv b(\widetilde{c})\colon \boldsymbol{\nu} a\ P$ and $\boldsymbol{\nu} a\ \bar{b}(\widetilde{c})\colon P \equiv \bar{b}(\widetilde{c})\colon \boldsymbol{\nu} a\ P$*

(2) *If $\widetilde{b} \cap \mathsf{fn}(Q) = \emptyset$ then $Q \mid a(\widetilde{b})\colon P \equiv a(\widetilde{b})\colon (Q \mid P)$ and $Q \mid \bar{a}(\widetilde{b})\colon P \equiv \bar{a}(\widetilde{b})\colon (Q \mid P)$.*

Lemma 3.1 expresses a property of a restricted interaction consuming permeable prefixes. Lemma 3.2 shows two structural laws for permeable prefixes, one concerning restriction, the other concerning parallel composition. These lemmas simply follow from the syntactic definition of the permeable prefixes. The assumption about the transitivity of wires, in Lemma 3.1, is harmless as all the wires we use in this paper are transitive. In what follows, we will use these structural rules without explicitly mentioning Lemma 3.2.

4. Abstract Encoding

This section introduces the abstract encoding of λ-terms into Iπ-processes. We call the encoding 'abstract' because it uses the abstract wires discussed in the previous section. In other words, the encoding is *parametric* with respect to the concrete definition of the wires. We then prove that, whenever the wires satisfy a few basic laws, the encoding yields a λ-theory.

4.1. Definition of the abstract encoding.

We begin by recalling Milner's original encoding $\mathcal{M}$ of (call-by-name) λ-calculus into the π-calculus [Mil92, Mil93]:

$$\begin{aligned}
\mathcal{M}[\![x]\!]_p &\stackrel{\text{def}}{=} \overline{x}\langle p\rangle \\
\mathcal{M}[\![\lambda x.\, M]\!]_p &\stackrel{\text{def}}{=} p(x,q).\, \mathcal{M}[\![M]\!]_q \\
\mathcal{M}[\![M\, N]\!]_p &\stackrel{\text{def}}{=} (\boldsymbol{\nu} q, x)\ (\mathcal{M}[\![M]\!]_q \mid \overline{q}\langle x, p\rangle \mid\ !x(r).\, \mathcal{M}[\![N]\!]_r)
\end{aligned}$$

$$\begin{aligned}
\mathcal{A}[\![x]\!]_p &\stackrel{\text{def}}{=} \bar{x}(p') \colon p \leftrightarrow \bar{p}' \\
\mathcal{A}[\![\lambda x.\, M]\!]_p &\stackrel{\text{def}}{=} p(x,q) \colon \mathcal{A}[\![M]\!]_q \\
\mathcal{A}[\![M\, N]\!]_p &\stackrel{\text{def}}{=} \boldsymbol{\nu} q\, (\mathcal{A}[\![M]\!]_q \mid \bar{q}(x,p') \colon (!x(r).\, \mathcal{A}[\![N]\!]_r \mid p \leftrightarrow \bar{p}'))
\end{aligned}$$

Figure 2: The abstract encoding $\mathcal{A}$.

The encoding of a λ-term M is parametric over a port p, which can be thought of as the *location* of M, for p represents the unique port along which M may be called by its environment, thus receiving two names: (a trigger for) its argument and the location to be used for the next interaction. Hence, $\mathcal{M}$ (as well as all the encodings in the paper) is a function from λ-terms to abstractions of the form $(p)\, P$. We write $\mathcal{M}[\![M]\!]_p$ as a shorthand for $\mathcal{M}[\![M]\!]\langle p \rangle$. A function application of the λ-calculus becomes, in the π-calculus, a particular form of parallel combination of two agents, the function and its argument; β-reduction is then modelled as process interaction. An argument of an application is translated as a replicated server, that can be used as many times as needed, each time providing a name to be used as location for the following computation.

In Figure 2 we report the abstract encoding $\mathcal{A}$. There are two main modifications from Milner's encoding $\mathcal{M}$:

(1) The encoding uses Iπ, rather than π-calculus; for this, all free outputs are replaced by a combination of bound outputs and wires (as hinted in Introduction).
(2) A permeable input is used, in place of an ordinary input, in the translation of abstraction so to allow reductions underneath a λ-abstraction. (We thus implement a *strong* call-by-name strategy.)

We report a few basic conditions that will be required on wires. The main ones concern the behaviour of wires as substitutions and transitivity of wires.

Definition 4.1 (Wires)**.** As a convention, we assume that names a, b, c are of the same sort, either location names or variable names. *Wires* $a \leftrightarrow \bar{b}$ are processes that satisfy the following properties:

(1) The free names of $a \leftrightarrow \bar{b}$ are a and b. Furthermore, $a \leftrightarrow \bar{b}$ only uses a in input and b in output.
(2) If $a \leftrightarrow \bar{b} \xrightarrow{\mu} P$ for some P, then $\mu \neq \tau$.
(3) $\boldsymbol{\nu} b\, (a \leftrightarrow \bar{b} \mid b \leftrightarrow \bar{c}) \gtrsim a \leftrightarrow \bar{c}$.
(4) $\boldsymbol{\nu} q\, (p \leftrightarrow \bar{q} \mid q(x,r) \colon P) \gtrsim p(x,r) \colon P$, provided that $(\boldsymbol{\nu} x, r\,)(x \leftrightarrow \bar{x}' \mid r' \leftrightarrow \bar{r} \mid P) \gtrsim P\{x', r'/x, r\}$, where x', r' are fresh names.
(5) $\boldsymbol{\nu} p\ (p \leftrightarrow \bar{q} \mid \bar{p}(x,r) \colon P) \gtrsim \bar{q}(x,r) : P$, provided that $(\boldsymbol{\nu} x, r\,)(x' \leftrightarrow \bar{x} \mid r \leftrightarrow \bar{r}' \mid P) \gtrsim P\{x', r'/x, r\}$, where x', r' are fresh names.
(6) $\boldsymbol{\nu} y\, (x \leftrightarrow \bar{y} \mid !y(p).\, P) \gtrsim !x(p).\, P$, provided that $y \notin \mathsf{fn}(P)$ and $\boldsymbol{\nu} p\, (p' \leftrightarrow \bar{p} \mid P) \gtrsim P\{p'/p\}$, where p' is fresh.
(7) $\boldsymbol{\nu} x\, (x \leftrightarrow \bar{y} \mid \bar{x}(p) \colon P) \gtrsim \bar{y}(p) \colon P$, provided that $x \notin \mathsf{fn}(P)$ and $\boldsymbol{\nu} p\, (p \leftrightarrow \bar{p}' \mid P) \gtrsim P\{p'/p\}$, where p' is fresh.
(8) $x \leftrightarrow \bar{y}$ is a replicated input process at x, i.e. $x \leftrightarrow \bar{y} = !x(p).\, P$ for some P.

Condition 1 is a simple syntactic requirement. Condition 2 says that wires are 'optimised' in that they cannot do any immediate internal interaction (this requirement, while not

mandatory, facilitates a few proofs). Law 3 is about the transitivity of wires. Laws 4-7 show that wires act as substitutions for permeable inputs, permeable outputs and replicated input prefixes. We do not require similar laws for ordinary prefixes, e.g., as in

$$\boldsymbol{\nu} p\ (p \leftrightarrow \bar{q} \mid \overline{p}(x,r).\, P) \gtrsim \overline{q}(x,r).\, P$$

because wires break the strict sequentiality imposed by such prefixes (essentially transforming an ordinary prefix into a permeable one: only for the process on the right any action from P is blocked until the environment accepts an interaction at q). Condition 8 requires $x \leftrightarrow \bar{y}$ to be an input replicated processes, and is useful so to be able to use the replication laws (Lemma 2.7). Location wires, on the other hand, are linear and hence should not be duplicated. Moreover, we do not even require them to be an input process because we shall consider location wires with different control flows.

Hereafter we assume that $p \leftrightarrow \bar{q}$ and $x \leftrightarrow \bar{y}$ are indeed wires, i.e., processes that satisfy the requirements of Definition 4.1. We can therefore exploit such requirements to derive properties of the abstract encoding.

Lemma 4.2 shows that the processes encoding functions are I-respectful with respect to the location wires, and O-respectful with respect to the variable wires.

Lemma 4.2.

(1) $\boldsymbol{\nu} p\, (q \leftrightarrow \bar{p} \mid \mathcal{A}[\![M]\!]_p) \gtrsim \mathcal{A}[\![M]\!]_q$
(2) $\boldsymbol{\nu} x\, (x \leftrightarrow \bar{y} \mid \mathcal{A}[\![M]\!]_p) \gtrsim \mathcal{A}[\![M\{y/x\}]\!]_p$

Proof. We prove 1 and 2 simultaneously by induction on the structure of M.

Case $M = x$**:** We begin with the case of location names.

$$\begin{aligned}
\boldsymbol{\nu} p\, ([\![x]\!]_p \mid q \leftrightarrow \bar{p}) &= \boldsymbol{\nu} p\, (\bar{x}(p') \colon p \leftrightarrow \bar{p}' \mid q \leftrightarrow \bar{p}) \\
&\equiv \bar{x}(p') \colon \boldsymbol{\nu} p\, (p \leftrightarrow \bar{p}' \mid q \leftrightarrow \bar{p}) \\
&\gtrsim \bar{x}(p') \colon q \leftrightarrow \bar{p}' && \text{(3 of Definition 4.1)} \\
&= [\![x]\!]_q
\end{aligned}$$

Next we show that $\boldsymbol{\nu} x\, ([\![M]\!]_p \mid x \leftrightarrow \bar{y}) = [\![M\{y/x\}]\!]_p$ holds when M is a variable. First, we consider the case where $M = z \neq x$. Since $x \leftrightarrow \bar{y}$ is of the form $!x(p).\, P$ by 8 of Definition 4.1, it follows that $\boldsymbol{\nu} x\, (x \leftrightarrow \bar{y}) \sim \mathbf{0}$, and this concludes this case. If $M = x$, then

$$\begin{aligned}
\boldsymbol{\nu} x\, ([\![x]\!]_p \mid x \leftrightarrow \bar{y}) &= \boldsymbol{\nu} x\, (\bar{x}(p') \colon p \leftrightarrow \bar{p}' \mid x \leftrightarrow \bar{y}) \\
&\gtrsim \bar{y}(p') \colon p \leftrightarrow \bar{p}' && \text{(7 of Definition 4.1)} \\
&= [\![y]\!]_p
\end{aligned}$$

The premise of 7 of Definition 4.1 is satisfied because of the transitivity of wires (3 of Definition 4.1).

Case $M = \lambda x.\, N$**:** We first show that 1 holds. This is a direct consequence of 4 of Definition 4.1.

$$\begin{aligned}
\boldsymbol{\nu} p\, ([\![\lambda x.\, N]\!]_p \mid q \leftrightarrow \bar{p}) &= \boldsymbol{\nu} p\, (p(x,r) \colon [\![N]\!]_r \mid q \leftrightarrow \bar{p}) \\
&\gtrsim q(x,r) \colon [\![N]\!]_r && \text{(4 of Definition 4.1 together with the i.h.)} \\
&= [\![\lambda x.\, N]\!]_q
\end{aligned}$$

The proof for 2 is also a direct consequence of the induction hypothesis. That is,

$$\begin{aligned}
\boldsymbol{\nu} y\,(\llbracket \lambda x.\, N \rrbracket_p \mid y \leftrightarrow \bar{z}) &= \boldsymbol{\nu} y\,(p(x,q)\colon \llbracket N \rrbracket_q \mid y \leftrightarrow \bar{z}) \\
&\equiv p(x,q)\colon \boldsymbol{\nu} y\,(\llbracket N \rrbracket_q \mid y \leftrightarrow \bar{z}) \\
&\gtrsim p(x,q)\colon (\llbracket N\{z/y\} \rrbracket_q) && \text{(i.h.)} \\
&= \llbracket (\lambda x.\, N)\{z/y\} \rrbracket_p
\end{aligned}$$

Here we assumed that x, y and z are pairwise distinct; the general case can be proved using α-conversion.

Case $M = N\,L$**:** The case for location names follows from the transitivity of wires. Observe that

$$\begin{aligned}
&\boldsymbol{\nu} p\ \big(\bar{r}(x,p')\colon (!x(r').\,\llbracket L \rrbracket_{r'} \mid p \leftrightarrow \bar{p}') \mid q \leftrightarrow \bar{p}\big) \\
&\equiv (\boldsymbol{\nu} p, x, p'\,)(\bar{r}(x',p'').\,(x' \leftrightarrow \bar{x} \mid p' \leftrightarrow \bar{p}'') \mid !x(r').\,\llbracket L \rrbracket_{r'} \mid p \leftrightarrow \bar{p}' \mid q \leftrightarrow \bar{p}) \\
&\qquad \text{(definition of permeable prefix)} \\
&\gtrsim (\boldsymbol{\nu} x, p'\,)(\bar{r}(x',p'').\,(x' \leftrightarrow \bar{x} \mid p' \leftrightarrow \bar{p}'') \mid !x(r').\,\llbracket L \rrbracket_{r'} \mid q \leftrightarrow \bar{p}') \qquad \text{(3 of Definition 4.1)} \\
&= \bar{r}(x,p')\colon (!x(r').\,\llbracket L \rrbracket_{r'} \mid q \leftrightarrow \bar{p}') \qquad \text{(definition of permeable prefix)}
\end{aligned}$$

Using this, we get

$$\begin{aligned}
\boldsymbol{\nu} p\ (\llbracket N\,L \rrbracket_p \mid q \leftrightarrow \bar{p}) &\equiv \boldsymbol{\nu} r\ \big(\llbracket N \rrbracket_r \mid \boldsymbol{\nu} p\ \big(\bar{r}(x,p')\colon (!x(r').\,\llbracket L \rrbracket_{r'} \mid p \leftrightarrow \bar{p}') \mid q \leftrightarrow \bar{p}\big)\big) \\
&\gtrsim \boldsymbol{\nu} r\ \big(\llbracket N \rrbracket_r \mid \bar{r}(x,p')\colon (!x(r').\,\llbracket L \rrbracket_{r'} \mid q \leftrightarrow \bar{p}')\big) \\
&= \llbracket N\,L \rrbracket_q
\end{aligned}$$

The proof for $x \leftrightarrow \bar{y}$ follows from the replication theorem and the induction hypothesis. First, observe that $x \leftrightarrow \bar{y}$ must be of the form $!x(p).\,P$ because of 8 of Definition 4.1. We, therefore, can apply the replication theorem to $x \leftrightarrow \bar{y}$. Hence, we have

$$\begin{aligned}
\boldsymbol{\nu} x\,(!z(p).\,\llbracket L \rrbracket_p \mid x \leftrightarrow \bar{y}) &\sim !z(p).\,\boldsymbol{\nu} x\,(\llbracket L \rrbracket_p \mid x \leftrightarrow \bar{y}) && \text{(replication theorem)} \\
&\gtrsim !z(p).\,\llbracket L\{y/x\} \rrbracket_p && \text{(i.h.)}
\end{aligned}$$

The claim follows by applying this expansion relation, the replication theorem for parallel composition and the induction hypothesis:

$$\begin{aligned}
&\boldsymbol{\nu} x\,(\llbracket N\,L \rrbracket_p \mid x \leftrightarrow \bar{y}) \\
&\equiv (\boldsymbol{\nu} x, r\,)\,\big(\llbracket N \rrbracket_r \mid \bar{r}(z,p')\colon (!z(r').\,\llbracket L \rrbracket_{r'} \mid p \leftrightarrow \bar{p}') \mid x \leftrightarrow \bar{y}\big) \\
&\sim \boldsymbol{\nu} r\,(\boldsymbol{\nu} x\ (\llbracket N \rrbracket_r \mid x \leftrightarrow \bar{y}) && \text{(replication theorem)} \\
&\qquad \mid \bar{r}(z,p')\colon (\boldsymbol{\nu} x\ \big(!z(r').\,\llbracket L \rrbracket_{r'} \mid x \leftrightarrow \bar{y}\big) \mid p \leftrightarrow \bar{p}')) \\
&\gtrsim \boldsymbol{\nu} r\ \big(\llbracket N\{y/x\} \rrbracket_r \mid \bar{r}(z,p')\colon (!z(r').\,\llbracket L\{y/x\} \rrbracket_{r'} \mid p \leftrightarrow \bar{p}')\big) \\
&\qquad \text{(i.h. and the above expansion relation)} \\
&= \llbracket N\{y/x\}\ L\{y/x\} \rrbracket_p \\
&= \llbracket (N\,L)\{y/x\} \rrbracket_p.
\end{aligned}$$

□

4.2. **Validity of β-reduction.** The abstract encoding $\mathcal{A}$ validates β-reduction with respect to the expansion relation. This is proved by showing that substitution of a λ-term M is implemented as a communication to a replicated server that owns M. The proof is similar to that for Milner's encoding, except that we exploit Lemma 4.2. We recall that $M \to N$ is the *full* β-reduction.

Lemma 4.3 (Substitution)**.** *If $x \notin \mathsf{fv}(N)$, then $\boldsymbol{\nu} x\,(\mathcal{A}[\![M]\!]_p \mid !x(q).\,\mathcal{A}[\![N]\!]_q) \gtrsim \mathcal{A}[\![M\{^N\!/\!_x\}]\!]_p$.*

Proof. By induction on the structure of M using the replication theorem.

For the base case, we consider the case $M = x$; if $x \notin \mathsf{fv}(M)$, then we just need to apply the garbage collection law.

$$\begin{aligned}
\boldsymbol{\nu} x\,([\![x]\!]_p \mid !x(q).\,[\![N]\!]_q) &= \boldsymbol{\nu} x\,(\bar{x}(p')\colon p \leftrightarrow \vec{p}' \mid !x(q).\,[\![N]\!]_q) \\
&\gtrsim \boldsymbol{\nu} p'\,(p \leftrightarrow \vec{p}' \mid [\![N]\!]_{p'}) \quad \text{(Lemma 3.1, and garbage collection on } x) \\
&\gtrsim [\![N]\!]_p. \qquad \text{(Lemma 4.2)}
\end{aligned}$$

The case $M = \lambda y.\,M'$ is a straightforward consequence of the induction hypothesis:

$$\begin{aligned}
\boldsymbol{\nu} x\,([\![\lambda y.\,M']\!]_p \mid !x(q).\,[\![N]\!]_q) &= \boldsymbol{\nu} x\,(p(y,r)\colon [\![M']\!]_r \mid !x(q).\,[\![N]\!]_q) \\
&\equiv p(y,r)\colon \boldsymbol{\nu} x\,([\![M']\!]_r \mid !x(q).\,[\![N]\!]_q) \qquad \text{(Lemma 3.2)} \\
&\gtrsim p(y,r)\colon [\![M'\{^N\!/\!_x\}]\!]_r \qquad \text{(i.h.)} \\
&= [\![(\lambda y.\,M')\{^N\!/\!_x\}]\!]_p.
\end{aligned}$$

It is the case where $M = M_1\,M_2$ that needs the replication theorem. If $M = M_1\,M_2$, we have

$$\begin{aligned}
&\boldsymbol{\nu} x\ ([\![M_1\,M_2]\!]_p \mid !x(q).\,[\![N]\!]_q) \\
&\equiv (\boldsymbol{\nu} x, q)\,([\![M_1]\!]_q \mid \bar{q}(y,p')\colon \big(!y(r).\,[\![M_2]\!]_r \mid p \leftrightarrow \vec{p}'\big) \mid !x(q).\,[\![N]\!]_q) \\
&\sim \boldsymbol{\nu} q\,(\boldsymbol{\nu} x\ ([\![M_1]\!]_q \mid !x(q).\,[\![N]\!]_q) \mid \\
&\qquad \bar{q}(y,p')\colon \big(\boldsymbol{\nu} x\ (!y(r).\,[\![M_2]\!]_r \mid !x(q).\,[\![N]\!]_q) \mid p \leftrightarrow \vec{p}'\big)) \\
&\qquad\qquad \text{(replication theorem for parallel composition)} \\
&\sim \boldsymbol{\nu} q\,(\boldsymbol{\nu} x\ ([\![M_1]\!]_q \mid !x(q).\,[\![N]\!]_q) \mid \\
&\qquad \bar{q}(y,p')\colon \big(!y(r).\,\boldsymbol{\nu} x\ ([\![M_2]\!]_r \mid !x(q).\,[\![N]\!]_q) \mid p \leftrightarrow \vec{p}'\big)) \\
&\qquad\qquad \text{(replication theorem for replicated input)} \\
&\gtrsim \boldsymbol{\nu} q\ \big([\![M_1\{^N\!/\!_x\}]\!]_q \mid \bar{q}(y,p')\colon \big(!y(r).\,[\![M_2\{^N\!/\!_x\}]\!]_r \mid p \leftrightarrow \vec{p}'\big)\big) \qquad \text{(i.h.)} \\
&= [\![(M_1\,M_2)\{^N\!/\!_x\}]\!]_p \qquad \square
\end{aligned}$$

Theorem 4.4. *If $M \to N$, then $\mathcal{A}[\![M]\!]_p \gtrsim \mathcal{A}[\![N]\!]_p$.*

Proof. It suffices to show that $[\![(\lambda x.\,M)\,N]\!]_p \gtrsim [\![M\{^N\!/\!_x\}]\!]_p$ because the other cases follow from the precongruence of $\gtrsim$. We have

$$\begin{aligned}
[\![(\lambda x.\,M)\,N]\!]_p &= \boldsymbol{\nu} q\ \big(q(x,r)\colon [\![M]\!]_r \mid \bar{q}(y,p')\colon \big(!y(r').\,[\![N]\!]_{r'} \mid p \leftrightarrow \vec{p}'\big)\big) \qquad (q \text{ is fresh}) \\
&\gtrsim (\boldsymbol{\nu} y, p'\,)\,\big([\![M\{^y\!/\!_x\}]\!]_{p'} \mid !y(r').\,[\![N]\!]_{r'} \mid p \leftrightarrow \vec{p}'\big) \qquad \text{(Lemma 3.1 and 4.2)} \\
&\gtrsim \boldsymbol{\nu} y\ \big([\![M\{^y\!/\!_x\}]\!]_p \mid !y(r').\,[\![N]\!]_{r'}\big) \qquad \text{(Lemma 4.2)} \\
&\gtrsim [\![M\{^N\!/\!_x\}]\!]_p \qquad \text{(Lemma 4.3)}
\end{aligned}$$

as desired. $\square$

$$
\begin{aligned}
\mathcal{O}[\![x]\!]_p &\stackrel{\text{def}}{=} \bar{x}(p') \colon p \leftrightarrow \bar{p}' \\
\mathcal{O}[\![\lambda x.\, M]\!]_p &\stackrel{\text{def}}{=} p(x,q) \colon \mathcal{O}[\![M]\!]_q \\
\mathcal{O}[\![x\, M_1 \cdots M_n]\!]_p &\stackrel{\text{def}}{=} \bar{x}(p_0) \colon \mathcal{O}^n\langle p_0, p, \mathcal{O}[\![M_1]\!] \cdots \mathcal{O}[\![M_n]\!]\rangle \\
\mathcal{O}[\![(\lambda x.\, M_0)\, M_1 \cdots M_n]\!]_p &\stackrel{\text{def}}{=} \boldsymbol{\nu} p_0 \ (p_0(x,q) \colon \mathcal{O}[\![M_0]\!]_q \mid \mathcal{O}^n\langle p_0, p, \mathcal{O}[\![M_1]\!] \cdots \mathcal{O}[\![M_n]\!]\rangle) \\
\mathcal{O}^n\langle p_0, p, \mathcal{O}[\![M_1]\!] \cdots \mathcal{O}[\![M_n]\!]\rangle &\stackrel{\text{def}}{=} \begin{array}{l} \bar{p}_0(x_1,p_1) \colon \cdots \bar{p}_{n-1}(x_n,p_n) \colon \\ (!x_1(r_1).\, \mathcal{O}[\![M_1]\!]_{r_1} \mid \cdots \mid !x_n(r_n).\, \mathcal{O}[\![M_n]\!]_{r_n} \mid p \leftrightarrow \bar{p}_n) \end{array}
\end{aligned}
$$

Figure 3: Optimised encoding. (The number n is greater than 0 in the last three cases.)

Since bisimilarity is a congruence in Iπ and our encoding is compositional, the validity of β-reduction implies that the equivalence induced by the encoding is a λ-theory.

Corollary 4.5. *Let* $=_\pi \stackrel{def}{=} \{(M,N) \mid \mathcal{A}[\![M]\!] \approx \mathcal{A}[\![N]\!]\}$. *Then* $=_\pi$ *is a* λ*-theory, that is, a congruence on* λ*-terms that contains* β*-equivalence.*

Remark 4.6. From a λ-theory, a λ-model can be extracted [Bar84], hence Corollary 4.5 implies that we can construct a λ-model out of the process terms. The domain of the model would be the processes that are in the image of the encoding, quotiented with bisimilarity. We could not define the domain of the model out of all process terms (as in [San00], as opposed to the processes in the image of the encoding) because our proofs rely on Lemma 4.2, and such a lemma cannot be extended to the set of all processes.

5. Optimised Encoding

We introduce an optimised version of the abstract encoding, which removes certain 'administrative steps' on the process terms. This will allow us to have a sharper operational correspondence between λ-terms and processes, which will be needed in proofs in later sections. As in the previous section, we work with abstract wires, only assuming the requirements in Definition 4.1.

To motivate the need of the optimised encoding, let us consider the encoding of a term $(x\, M)\, N$:

$$
\begin{aligned}
(\boldsymbol{\nu} p_0, p_1)\, (&\bar{x}(p_0') \colon p_0' \leftrightarrow \bar{p}_0 \\
&\mid \bar{p}_0(x_1.\, p_1') \colon \big(!x_1(r_1).\, [\![M]\!]_{r_1} \mid p_1 \leftrightarrow \bar{p}_1'\big) \\
&\mid \bar{p}_1(x_2,p_2) \colon (!x_2(r_2).\, [\![N]\!]_{r_2} \mid p \leftrightarrow \bar{p}_2))
\end{aligned}
$$

This process has, potentially (i.e., depending on the concrete instantiations of the wires), some initial administrative reductions. For instance, the output at p_1 may interact with the input end of the wire $p_1 \leftrightarrow \bar{p}_1'$.

In the optimised encoding $\mathcal{O}$, in Figure 3, any initial reduction of a process has a direct correspondence with a (strong call-by-name) reduction of the source λ-term. With respect to the unoptimised encoding $\mathcal{A}$, the novelties are in the clauses for application, where the case of a head normal form $x\, M_1 \cdots M_n$ (for $n \geq 1$) and of an application $(\lambda x.\, M_0)\, M_1 \cdots M_n$ with a head redex are distinguished. In both cases, $\mathcal{O}^n\langle p_0, p, \mathcal{O}[\![M_1]\!] \cdots \mathcal{O}[\![M_n]\!]\rangle$ is used for a

compact representation of the encoding of the trailing arguments $M_1, \ldots, M_n$, as a sequence of nested permeable prefixes and a bunch of replications embracing the terms M_i.

Analogous properties to those in Section 4 for the unoptimised encoding $\mathcal{A}$ hold for $\mathcal{O}$. For instance, $\mathcal{O}$ validates β-reduction (Lemma 5.1). Using such properties, and reasoning by induction of the structure of a λ-term, we can prove that $\mathcal{O}$ is indeed an optimisation.

Lemma 5.1. *If $M \to N$, then $\mathcal{O}[\![M]\!]_p \gtrsim \mathcal{O}[\![N]\!]_p$.*

Lemma 5.2. *$\mathcal{A}[\![M]\!]_p \gtrsim \mathcal{O}[\![M]\!]_p$.*

The details about the operational behaviour of $\mathcal{O}[\![M]\!]_p$, and its operational correspondence with M, are described in Appendix B. We only report here the statements of a few important lemmas.

Lemma 5.3. *If $\mathcal{O}[\![M]\!]_p \xrightarrow{\tau} P$ then there exists N such that $M \to_{\mathtt{sn}} N$ and $P \gtrsim \mathcal{O}[\![N]\!]_p$.*

Lemma 5.4. *If $\mathcal{O}[\![M]\!]_p \xrightarrow{\mu} P$ and μ is an input action, then μ is an input at p.*

If $M = \lambda x.\, M'$, then its process encoding $\mathcal{O}[\![M]\!]_p$ can always do an input at p, as the encoding of a abstraction begins with an input at its location name. Lemma 5.4 says that such an input at p is indeed the only possible input; that is, we cannot observe an inner λ-abstraction (i.e., a λ-abstraction in M').

Later, when we relate our encoding to trees of the λ-calculus, the notions of head normal form and (un)solvable term will be important. Hence some of our operational correspondence results concern them.

Lemma 5.5. *Let M be a λ-term. If $\mathcal{O}[\![M]\!]_p \xrightarrow{\overline{x}(q)} P$ for some P, then M has a head normal form $\lambda\widetilde{y}.\, x\, \widetilde{M}$, for some (possibly empty) sequence of terms $\widetilde{M}$ and variables $\widetilde{y}$ with $x \notin \widetilde{y}$.*

Lemma 5.6. *Let M be an unsolvable term. Then there does not exist an output action μ such that $\mathcal{O}[\![M]\!]_p \stackrel{\mu}{\Longrightarrow} P$ for some P.*

Corollary 5.7. *If M is solvable then there are input actions $\mu_1, \ldots, \mu_n$ ($n \geq 0$) and an output action μ such that $\mathcal{O}[\![M]\!]_p \stackrel{\mu_1}{\Longrightarrow} \ldots \stackrel{\mu_n}{\Longrightarrow} \stackrel{\mu}{\Longrightarrow} P$, for some P.*

By Lemma 5.2, Corollary 5.7 also holds for $\mathcal{A}$. The converse of Corollary 5.7 will also hold, in all three concrete encodings that will be studied in the next section. However we believe the result cannot be derived from the assumptions on wires in Definition 4.1.

We conclude by looking, as an example, at the unsolvable term $\Omega \stackrel{\text{def}}{=} (\lambda x.\, x\, x)\, (\lambda x.\, x\, x)$.

Example 5.8. The process $\mathcal{O}[\![\Omega]\!]_p$ is

$$\begin{aligned}\boldsymbol{\nu} p_0\, (p_0(x,q) \colon \bar{x}(q_0) \colon \bar{q}_0(y_1,q_1) \colon (!y_1(r_1).\, \mathcal{O}[\![x]\!]_{r_1} \mid q \leftrightarrow \bar{q}_1)\\ \mid \bar{p}_0(x_1,p_1) \colon (!x_1(r_1).\, \mathcal{O}[\![\lambda x.\, x\, x]\!]_{r_1} \mid p \leftrightarrow \bar{p}_1))\end{aligned}$$

The only action $\mathcal{O}[\![\Omega]\!]_p$ can do is a τ-action or an input at p. Whether the input can be performed or not will depend on the concrete definition of the wire $p \leftrightarrow \bar{q}$. The possibility of an input action from an unsolvable of order 0 such as Ω is a major difference between our encoding and encodings in the literature, where the encoding of such unsolvables are usually purely divergent processes.

6. Concrete Wires

We now examine concrete instantiations of the abstract wires in the encoding $\mathcal{A}$ (and its optimisation $\mathcal{O}$). In each case we have to define the wires for location and variable names. The location wires are the important ones: the definition of the variable wires will follow from them, with the goal of guaranteeing their expected properties. We consider three concrete wires: *I-O wires*, *O-I wires*, and *P wires*. The main difference among them is in the order in which the input and output of the location wires are performed.

6.1. **Definition of the concrete wires.** Location and variable wires will be defined by means of recursion. In contrast with the variable wires, the location wires are non-replicated processes, reflecting the linear usage of such names. We recall that the choice of a certain kind of concrete wires (I-O wires, O-I wires, or P wires) also affects the definition of the permeable prefixes (as it refers to the wires), including the permeable prefixes that may be used within the wires themselves. We will also show de-sugared definitions of the concrete wires, i.e., without reference to permeable prefixes. We add a subscript ($\mathtt{IO}$, $\mathtt{OI}$, $\mathtt{P}$) to indicate a concrete wire (as opposed to an abstract one). For readability, in the definitions of the concrete wires the name parameters are instantiated (e.g., writing $a \leftrightarrow_{\mathtt{IO}} \bar{b} \stackrel{\text{def}}{=} P$ rather than $\leftrightarrow_{\mathtt{IO}} \stackrel{\text{def}}{=} (a, b)\, P$).

I-O wires. In the I-O wires, the input of a wire precedes the output.

$$p \leftrightarrow_{\mathtt{IO}} \bar{q} \stackrel{\text{def}}{=} p(y, p_1).\, \bar{q}(x, q_1) \colon (p_1 \leftrightarrow_{\mathtt{IO}} \bar{q}_1 \mid x \leftrightarrow_{\mathtt{IO}} \bar{y})$$

$$x \leftrightarrow_{\mathtt{IO}} \bar{y} \stackrel{\text{def}}{=} !x(p).\, \bar{y}(q) \colon p \leftrightarrow_{\mathtt{IO}} \bar{q}$$

Inlining the abbreviations for permeable prefixes (as they are, in turn, defined using wires, in this specific case, the I-O wires), we obtain:

$$p \leftrightarrow_{\mathtt{IO}} \bar{q} \stackrel{\text{def}}{=} p(y, p_1).\, (\boldsymbol{\nu} x, q_1)\, (\overline{q}(x', q_1').\, (q_1 \leftrightarrow_{\mathtt{IO}} \bar{q}_1' \mid x' \leftrightarrow_{\mathtt{IO}} \bar{x}) \mid p_1 \leftrightarrow_{\mathtt{IO}} \bar{q}_1 \mid x \leftrightarrow_{\mathtt{IO}} \bar{y})$$

$$x \leftrightarrow_{\mathtt{IO}} \bar{y} \stackrel{\text{def}}{=} !x(p).\, \boldsymbol{\nu} q\, (\overline{y}(q').\, q \leftrightarrow_{\mathtt{IO}} \bar{q}' \mid p \leftrightarrow_{\mathtt{IO}} \bar{q})$$

I-O wires, beginning with an input and proceeding with an output, are similar to the ordinary wires in the literature, sometimes called *forwarders*, and used to prove properties about asynchronous and localised π-calculi (or encodings of them) [HY95, Mer01, MS04, Bor98]. An important technical difference, within location wires, is the appearance of a permeable prefix, in place of an ordinary prefix, and the inner wire $p_1 \leftrightarrow_{\mathtt{IO}} \bar{q}_1$ that, in a forwarder, would have p_1 and q_1 swapped. The reason for these differences is that location wires are used with processes that are not localised (the recipient of a location name will use it in input, rather than in output). The difference also shows up in the semantic properties: forwarders in the literature are normally used to obtain properties of O-respectfulness (Section 3), with the input-end of the wire restricted; in contrast, I-O wires will be used to obtain properties of I-respectfulness, with the output-end of the wire restricted.

In the definition above of location wires, the permeable prefix cannot be replaced by an ordinary prefix: the transitivity of the wires (property 3 in Definitions 4.1) would be lost.

O-I wires. The symmetry of Iπ enable us to consider the dual form of (location) wire, with the opposite control flow, namely 'from output to input':

$$p \leftrightarrows_{\mathtt{OI}} \overline{q} \stackrel{\text{def}}{=} \overline{q}(x, q_1).\, p(y, p_1) \colon (p_1 \leftrightarrows_{\mathtt{OI}} \overline{q_1} \mid x \leftrightarrows_{\mathtt{OI}} \overline{y})$$

$$x \leftrightarrows_{\mathtt{OI}} \overline{y} \stackrel{\text{def}}{=} !x(p).\, \bar{y}(q) \colon p \leftrightarrows_{\mathtt{OI}} \overline{q}$$

Remark 6.1 (Duality)**.** If duality is taken to mean the exchange between input and output prefixes, then the set of location I-O wires is the dual of the set of O-I wires. Indeed, the location O-I wires are obtained from the corresponding location I-O wires by swapping input and output particles; variable wires, in contrast are left unchanged. This means that we obtain an O-I wire from an I-O wire if any input $p(\widetilde{b})$ is made into an output $\overline{p}(\widetilde{b})$, and conversely (moreover, accordingly, the parameters of the variable wires are swapped).

P wires. In the third form of wire, the sequentiality in location wires is broken: input and output execute concurrently. This is achieved by using, in the definition of location wires, only permeable prefixes.

$$p \leftrightarrows_{\mathtt{P}} \bar{q} \stackrel{\text{def}}{=} p(y, p_1) \colon \bar{q}(x, q_1) \colon (p_1 \leftrightarrows_{\mathtt{P}} \bar{q}_1 \mid x \leftrightarrows_{\mathtt{P}} \bar{y})$$

$$x \leftrightarrows_{\mathtt{P}} \bar{y} \stackrel{\text{def}}{=} !x(p).\, \bar{y}(q) \colon p \leftrightarrows_{\mathtt{P}} \bar{q}$$

Without the syntactic sugar of permeable prefixes, the definition of the location and variable P wires are thus:

$$\begin{aligned} p \leftrightarrows_{\mathtt{P}} \bar{q} \stackrel{\text{def}}{=} (\boldsymbol{\nu} p_1, q_1 x, y)\, (p(y', p_1').\, (p_1' \leftrightarrows_{\mathtt{P}} \bar{p}_1 \mid y \leftrightarrows_{\mathtt{P}} \bar{y}') \mid \\ \overline{q}(q_1', x').\, (q_1 \leftrightarrows_{\mathtt{P}} \bar{q}_1' \mid x' \leftrightarrows_{\mathtt{P}} \bar{x}) \mid \\ p_1 \leftrightarrows_{\mathtt{P}} \bar{q}_1 \mid x \leftrightarrows_{\mathtt{P}} \bar{y}) \end{aligned}$$

$$x \leftrightarrows_{\mathtt{P}} \bar{y} \stackrel{\text{def}}{=} !x(p).\, \boldsymbol{\nu} q\, (\overline{y}(q').\, q \leftrightarrows_{\mathtt{P}} \bar{q}' \mid p \leftrightarrows_{\mathtt{P}} \bar{q})$$

The wire $p \leftrightarrows_{\mathtt{P}} \bar{q}$ is dual of itself: due to the use of permeable prefixes, swapping input and output prefixes has no behavioural effect.

6.2. **Transitivity and well-definedness of the concrete wires.** All three wires introduced in the previous subsection are well defined:

Lemma 6.2. *The I-O wires, O-I wires and P wires are indeed wires; that is, they satisfy the laws of Definition 4.1.*

For each kind of wires, the proof is carried out in two steps. First, we show that the wires are transitive, using up-to techniques for bisimilarity. Then, the other laws are proved by algebraic reasoning (including the use of transitivity of wires). The algebraic reasoning is done in a similar manner for all the three wires.

Here, *we shall only give the proof of the transitivity of P wires* since it is the most delicate one, because of the concurrency allowed by permeable prefixes and because permeable prefixes are defined in terms of the wires themselves. The proofs of the transitivity of the other two wires, and the proofs for the other laws are reported in Appendix C.

To illustrate the difficulty of P wires, let us consider the wires $\boldsymbol{\nu} q\,(p \leftrightarrows_{\mathtt{P}} \bar{q} \mid q \leftrightarrows_{\mathtt{P}} \bar{r})$. This process can immediately reduce at the internal name q. Moreover, the derivative

$$p(p_1, y) \colon \bar{r}(q_1, x) \colon ((\boldsymbol{\nu} z_1, z_2)\,(x \leftrightarrows_{\mathtt{P}} \bar{z}_1 \mid z_1 \leftrightarrows_{\mathtt{P}} \bar{z}_2 \mid z_2 \leftrightarrows_{\mathtt{P}} \bar{y}) \mid (\boldsymbol{\nu} s_1, s_2)\,(p_1 \leftrightarrows_{\mathtt{P}} \bar{s}_1 \mid s_1 \leftrightarrows_{\mathtt{P}} \bar{s}_2 \mid s_2 \leftrightarrows_{\mathtt{P}} \bar{q}_1))$$

shows that the reduction has made the chain of wires longer. (Further reductions are then possible; indeed infinitely-many reductions may thus be produced.) To deal with these cases, we strengthen the claim and show the transitivity of chains of wires. In doing so, we crucially rely on up-to proof techniques for Iπ, notably 'expansion up-to expansion and context', and several algebraic laws (cf. Section 2.2.1). The 'up-to context' is used to cut out common contexts such as $p(p_1, y) \colon \bar{r}(q_1, x) \colon ([\cdot] \mid [\cdot])$. The algebraic laws are used to transform the processes so to be able to apply the 'up-to context', mainly by performing internal interactions. It is unclear how the proof could be carried out without such proof techniques.

As we need to consider chains of P wires of arbitrary length, we introduce a notation for them. We thus set:

$$\mathrm{chain}^1_{\mathtt{P}}(p, q) \stackrel{\text{def}}{=} p \leftrightarrows_{\mathtt{I0}} \bar{q} \qquad \mathrm{chain}^{n+1}_{\mathtt{P}}(p, q) \stackrel{\text{def}}{=} \boldsymbol{\nu} r\,(\mathrm{chain}^n_{\mathtt{P}}(p, r) \mid r \leftrightarrows_{\mathtt{I0}} \bar{q})$$

$$\mathrm{chain}^1_{\mathtt{P}}(x, y) \stackrel{\text{def}}{=} x \leftrightarrows_{\mathtt{I0}} \bar{y} \qquad \mathrm{chain}^{n+1}_{\mathtt{P}}(x, y) \stackrel{\text{def}}{=} \boldsymbol{\nu} z\,(\mathrm{chain}^n_{\mathtt{P}}(x, z) \mid z \leftrightarrows_{\mathtt{I0}} \bar{y})$$

where $r \notin \{p, q\}$ and $z \notin \{x, y\}$.

Now we are ready to show the transitivity of P wires.

Lemma 6.3. *The P wires $p \leftrightarrows_{\mathtt{P}} \bar{q}$ and $x \leftrightarrows_{\mathtt{P}} \bar{y}$ are transitive, that is, $\boldsymbol{\nu} q(p \leftrightarrows_{\mathtt{P}} \bar{q} \mid q \leftrightarrows_{\mathtt{P}} \bar{r}) \gtrsim p \leftrightarrows_{\mathtt{P}} \bar{r}$ and $\boldsymbol{\nu} y\,(x \leftrightarrows_{\mathtt{P}} \bar{y} \mid y \leftrightarrows_{\mathtt{P}} \bar{z}) \gtrsim x \leftrightarrows_{\mathtt{P}} \bar{z}$.*

Proof. For the proof, we strengthen the statement and prove transitivity for chains of wires of arbitrary length n. Let

$$\mathcal{R}_1 \stackrel{\text{def}}{=} \left\{ \left(p_0 \leftrightarrows_{\mathtt{P}} \bar{p}_n, \mathrm{chain}^n_{\mathtt{P}}(p_0, p_n) \right) \;\middle|\; n \geq 2 \right\}$$

$$\mathcal{R}_2 \stackrel{\text{def}}{=} \left\{ \left(x_0 \leftrightarrows_{\mathtt{P}} \bar{x}_n, \mathrm{chain}^n_{\mathtt{P}}(x_0, x_n) \right) \;\middle|\; n \geq 2 \right\}$$

We show that $\mathcal{R}_1 \cup \mathcal{R}_2$ is an expansion up-to $\lesssim$ and context.

Before considering how processes in the relation can match each other's transition, we present some useful observations that will be used throughout the proof. Recall that $p_i \leftrightarrows_{\mathtt{P}} \bar{p}_{i+1}$ is of the form

$$\begin{aligned} &(\boldsymbol{\nu} x_i^+, q_i^+, x_{i+1}^-, q_{i+1}^-) \\ &(p_i(x_i, q_i).\,(x_i^+ \leftrightarrows_{\mathtt{P}} \bar{x}_i \mid q_i \leftrightarrows_{\mathtt{P}} \bar{q}_i^+) \\ &\quad \mid \overline{p_{i+1}}(x_{i+1}, q_{i+1}).\,(x_{i+1} \leftrightarrows_{\mathtt{P}} \bar{x}_{i+1}^- \mid q_{i+1}^- \leftrightarrows_{\mathtt{P}} \bar{q}_{i+1}) \\ &\quad \mid x_{i+1}^- \leftrightarrows_{\mathtt{P}} \bar{x}_i^+ \mid q_i^+ \leftrightarrows_{\mathtt{P}} \bar{q}_{i+1}^-) \end{aligned}$$

Therefore, given

$$\mathrm{chain}^n_{\mathtt{P}}(p_0, p_n) \equiv (\boldsymbol{\nu} p_1, \ldots, p_{n-1})\,(p_0 \leftrightarrows_{\mathtt{P}} \bar{p}_1 \mid \cdots \mid p_{n-1} \leftrightarrows_{\mathtt{P}} \bar{p}_n)\,,$$

reducing the process by executing all the (internal) interactions at the p_i's, gives us a process of the form

$$\begin{aligned}
&(\boldsymbol{\nu} x_0^+, x_1^-, x_1, x_1^+, \ldots, x_{n-1}^-, x_{n-1}, x_{n-1}^+, x_n^-\,)\\
&(\boldsymbol{\nu} q_0^+, q_1^-, q_1, q_1^+, \ldots, q_{n-1}^-, q_{n-1}, q_{n-1}^+, q_n^-\,)\\
&(p_0(x_0, q_0).\,(x_0^+ \leftrightarrows_{\mathtt{P}} \bar{x}_0 \mid q_0 \leftrightarrows_{\mathtt{P}} \bar{q}_0^+)\\
&\mid \overline{p_n}(x_n, q_n).\,(x_n \leftrightarrows_{\mathtt{P}} \bar{x}_n^- \mid q_n^- \leftrightarrows_{\mathtt{P}} \bar{q}_n)\\
&\mid q_0^+ \leftrightarrows_{\mathtt{P}} \bar{q}_1^- \mid q_1^- \leftrightarrows_{\mathtt{P}} \bar{q}_1 \mid q_1 \leftrightarrows_{\mathtt{P}} \bar{q}_1^+ \mid \cdots\\
&\mid q_{n-1}^- \leftrightarrows_{\mathtt{P}} \bar{q}_{n-1} \mid q_{n-1} \leftrightarrows_{\mathtt{P}} \bar{q}_{n-1}^+ \mid q_{n-1}^+ \leftrightarrows_{\mathtt{P}} \bar{q}_n^-\\
&\mid x_n^- \leftrightarrows_{\mathtt{P}} \bar{x}_{n-1}^+ \mid x_{n-1}^+ \leftrightarrows_{\mathtt{P}} \bar{x}_{n-1} \mid x_{n-1} \leftrightarrows_{\mathtt{P}} \bar{x}_{n-1}^- \mid \cdots\\
&\mid x_1^+ \leftrightarrows_{\mathtt{P}} \bar{x}_1 \mid x_1 \leftrightarrows_{\mathtt{P}} \bar{x}_1^- \mid x_1^- \leftrightarrows_{\mathtt{P}} \bar{x}_0^+)
\end{aligned}$$

Up to structural congruence, the above process can be written as

$$\begin{aligned}
&(\boldsymbol{\nu} x_0^+, x_n^-\,)(\boldsymbol{\nu} q_0^+, q_n^-\,)\\
&(p_0(x_0, q_0).\,(x_0^+ \leftrightarrows_{\mathtt{P}} \bar{x}_0 \mid q_0 \leftrightarrows_{\mathtt{P}} \bar{q}_0^+)\\
&\mid \overline{p_n}(x_n, q_n).\,(x_n \leftrightarrows_{\mathtt{P}} \bar{x}_n^- \mid q_n^- \leftrightarrows_{\mathtt{P}} \bar{q}_n)\\
&\mid \mathrm{chain}_{\mathtt{P}}^{3n+1}(q_0^+, q_{n+1}^-) \mid \mathrm{chain}_{\mathtt{P}}^{3n+1}(x_{n+1}^-, x_0^+))\\
&\equiv p_0(x_0, q_0) \colon \bar{p}_{n+1}(x_{n+1}, q_{n+1}) \colon (\mathrm{chain}_{\mathtt{P}}^{3n-2}(x_n, x_0) \mid \mathrm{chain}_{\mathtt{P}}^{3n-2}(q_0, q_n))
\end{aligned}$$

Moreover, since the reductions performed are all at restricted linear names, in each reduction the initial process is in the relation $\gtrsim$ with the derivative process; that is, for any $n \geq 2$, we have

$$\mathrm{chain}_{\mathtt{P}}^{n}(p_0, p_n) \gtrsim p_0(x_0, q_0) \colon \bar{p}_n(x_n, q_n) \colon (\mathrm{chain}_{\mathtt{P}}^{3n-2}(x_n, x_0) \mid \mathrm{chain}_{\mathtt{P}}^{3n-2}(q_0, q_n)) \tag{6.1}$$

Exploiting this property, we can prove that $\mathcal{R}_1 \cup \mathcal{R}_2$ is an expansion up-to expansion and context. We first consider the case where $p_0 \leftrightarrows_{\mathtt{P}} \bar{p}_n \; \mathcal{R}_1 \; \mathrm{chain}_{\mathtt{P}}^{n}(p_0, p_n)$. We only consider the case where the process on the right-hand side makes the challenge; the opposite direction can be proved similarly. There are three possible actions that the process on the right-hand side can make: (1) τ-action, (2) input at p_0, and (3) output at p_{n+1}. We start by proving the first case. If

$$(\boldsymbol{\nu} p_1, \ldots, p_{n-1}\,)(p_0 \leftrightarrows_{\mathtt{P}} \bar{p}_1 \mid \cdots \mid p_{n-1} \leftrightarrows_{\mathtt{P}} \bar{p}_n) \xrightarrow{\tau} P\,,$$

then the action must have been caused by an interaction at p_i, for some i such that $1 \leq i \leq n-1$. We can execute the interactions at the remaining p_i's, and then, using the property (6.1) above, we have

$$P \gtrsim p_0(x_0, q_0) \colon \bar{p}_{n+1}(x_n, q_n) \colon (\mathrm{chain}_{\mathtt{P}}^{3n-2}(x_n, x_0) \mid \mathrm{chain}_{\mathtt{P}}^{3n-2}(q_0, q_n)).$$

For the matching transition we take the 0-step transition, i.e. the identity relation. Since

$$\begin{aligned}
&p_0 \leftrightarrows_{\mathtt{P}} \bar{p}_n = p_0(x_0, q_0) \colon \bar{p}_n(x_n, q_n) \colon (x_n \leftrightarrows_{\mathtt{P}} \bar{x}_0 \mid q_n \leftrightarrows_{\mathtt{P}} \bar{q}_0)\,,\\
&x_n \leftrightarrows_{\mathtt{P}} \bar{x}_0 \; \mathcal{R}_2 \; \mathrm{chain}_{\mathtt{P}}^{3n-2}(x_n, x_0)\,,\\
&q_0 \leftrightarrows_{\mathtt{P}} \bar{q}_n \; \mathcal{R}_1 \; \mathrm{chain}_{\mathtt{P}}^{3n-2}(q_0, q_n)
\end{aligned}$$

we can conclude this case using the up-to expansion and context technique. Similarly, if

$$\mathrm{chain}^n_{\mathtt{P}}(p_0, p_n) \xrightarrow{p_0(x_0,q_0)} P,$$

then we can show that

$$P \gtrsim (\boldsymbol{\nu} x_0^+, q_0^+)(x_0^+ \leftrightarrows_{\mathtt{P}} \bar{x}_0 \mid q_0 \leftrightarrows_{\mathtt{P}} \bar{q}_0^+ \mid \bar{p}_n(x_n, q_n) \colon (\mathrm{chain}^{3n-2}_{\mathtt{P}}(x_n, x_0^+) \mid \mathrm{chain}^{3n-2}_{\mathtt{P}}(q_0^+, q_n))).$$

We can match this transition with

$$p_0 \leftrightarrows_{\mathtt{P}} \bar{p}_n \xrightarrow{p_0(x_0,q_0)} (\boldsymbol{\nu} x_0^+, q_0^+)(x_0^+ \leftrightarrows_{\mathtt{P}} \bar{x}_0 \mid q_0 \leftrightarrows_{\mathtt{P}} \bar{q}_0^+ \mid \bar{p}_n(x_n, q_n) \colon (x_n \leftrightarrows_{\mathtt{P}} \bar{x}_0^+ \mid q_0^+ \leftrightarrows_{\mathtt{P}} \bar{q}_n)).$$

Once again, since

$$x_n \leftrightarrows_{\mathtt{P}} \bar{x}_0 \; \mathcal{R}_2 \; ,$$

$$q_0 \leftrightarrows_{\mathtt{P}} \bar{q}_n \; \mathcal{R}_1 \; \mathrm{chain}^{3n-2}_{\mathtt{P}}(q_0, q_n)$$

we can appeal to the up-to expansion and context technique to finish the case.

The remaining case (the case where process makes an output at p_n) can be proved by the same reasoning.

Now we sketch the case for variable names. Take

$$x_0 \leftrightarrows_{\mathtt{P}} \bar{x}_n \; \mathcal{R}_2 \; \mathrm{chain}^n_{\mathtt{P}}(x_0, x_n).$$

There is only one possible action that the process on the right-hand side can make: an input at x_0. By a reasoning similar to that of the location wires, we can show that if

$$\mathrm{chain}^n_{\mathtt{P}}(x_0, x_n) \xrightarrow{x_0(p_0)} P$$

then

$$P \gtrsim \mathrm{chain}^n_{\mathtt{P}}(x_0, x_n) \mid \bar{x}_n(p_n) \colon \mathrm{chain}^{2n-1}_{\mathtt{P}}(p_0, p_n)$$

by executing the interactions at the x_i's. The above transition can be matched by the transition $x_0 \leftrightarrows_{\mathtt{P}} \bar{x}_n \xrightarrow{x_0(p_0)} x_0 \leftrightarrows_{\mathtt{P}} \bar{x}_n \mid \bar{x}_n(p_n) \colon p_0 \leftrightarrows_{\mathtt{P}} \bar{p}_n$, and we can conclude by using the up-to context technique. □

Remark 6.4. The duality between I-O wires and O-I wires also shows up in proofs. For instance, for I-O wires the proof of law 4 of Definitions 4.1 does not use the premise of the law (i.e., the respectfulness of P), whereas the proof of the dual law 5 does. In the case of O-I wires, the opposite happens: the proof of law 5 uses the premise, whereas that of law 4 does not.

In the following sections we examine the concrete encodings obtained by instantiating the wires of the abstract encoding $\mathcal{A}$ (Figure 2) with the I-O wires, O-I wires, and P wires. We denote the resulting (concrete) encodings as $\mathcal{A}_{\mathtt{IO}}$, $\mathcal{A}_{\mathtt{OI}}$, and $\mathcal{A}_{\mathtt{P}}$, respectively. Similarly $\mathcal{O}_{\mathtt{IO}}$, $\mathcal{O}_{\mathtt{OI}}$, and $\mathcal{O}_{\mathtt{P}}$ are the instantiations of the abstract optimised encoding $\mathcal{O}$. For instance, in $\mathcal{A}_{\mathtt{IO}}$ and $\mathcal{O}_{\mathtt{IO}}$ an abstract wire $a \leftrightarrow \bar{b}$ is instantiated with the corresponding concrete wire $a \leftrightarrows_{\mathtt{IO}} \bar{b}$; and similarly for O-I wires and P wires.

Having shown that all the wires satisfy the requirements of Axiom 4.1, we can use, in the proofs about all concrete encodings (optimised and not) the results in Sections 4 and 5 for the abstract encoding and its abstract optimisation.

7. Full Abstraction for LTs and BTs

In this section we consider $\mathcal{A}_{\mathtt{IO}}$ and $\mathcal{A}_{\mathtt{OI}}$, and prove full abstraction with respect to the BTs and LTs, respectively. The main proof is given in Section 7.2. Before that, in Section 7.1, we discuss the difference between $\mathcal{A}_{\mathtt{IO}}$ and $\mathcal{A}_{\mathtt{OI}}$ on the encoding of unsolvable terms because it highlights the difference between the two encodings. Some lemmas about the encoding of unsolvable terms given in Section 7.1 will also play key roles in the main proof. In the proofs we exploit the optimised encodings $\mathcal{O}_{\mathtt{OI}}$ and $\mathcal{O}_{\mathtt{IO}}$.

7.1. Encoding of unsolvable terms.

We recall that the differences between BTs and LTs are due to the treatment of unsolvable terms (cf. Section 2.1). BTs equate all the unsolvable terms, whereas LTs distinguish unsolvables of different order, such as Ω and $\lambda x.\,\Omega$. We begin, as an example, with the terms Ω and $\lambda x.\,\Omega$. As we have seen in Example 5.8, in the abstract optimised encoding $\mathcal{O}$ process $\mathcal{O}[\![\Omega]\!]_p$ is:

$$\begin{aligned}\boldsymbol{\nu} p_0\,(p_0(x,q)\colon \bar{x}(q_0)\colon \bar{q}_0(y_1,q_1)\colon (!y_1(r_1).\,\mathcal{O}[\![x]\!]_{r_1} \mid q \leftrightarrow \bar{q}_1)\\ \mid \bar{p}_0(x_1,p_1)\colon (!x_1(r_1).\,\mathcal{O}[\![\lambda x.\,x\,x]\!]_{r_1} \mid p \leftrightarrow \bar{p}_1)).\end{aligned}$$

Its instantiation with O-I wires, $\mathcal{O}_{\mathtt{OI}}[\![\Omega]\!]_p$, cannot do any input action: as $p \leftrightarrow \bar{p}_1$ becomes the O-I wire $p \underset{\mathtt{OI}}{\leftrightarrow} \overline{p_1}$, the input occurrence of the free name p is guarded by p_1, which in turn is bound by the (permeable) prefix at p_0. Indeed, the only action that $\mathcal{O}_{\mathtt{OI}}[\![\Omega]\!]_p$ can perform is (up-to expansion) $\mathcal{O}_{\mathtt{OI}}[\![\Omega]\!]_p \stackrel{\tau}{\to}\gtrsim \mathcal{O}_{\mathtt{OI}}[\![\Omega]\!]_p$, which corresponds to the reduction $\Omega \to \Omega$. Hence, $\mathcal{O}_{\mathtt{OI}}[\![\Omega]\!]_p$ cannot match the input action $\mathcal{O}_{\mathtt{OI}}[\![\lambda x.\,\Omega]\!]_p \xrightarrow{p(x,q)} \mathcal{O}_{\mathtt{OI}}[\![\Omega]\!]_q$, and the two processes are distinguished.

In contrast, with I-O wires, processes $\mathcal{O}_{\mathtt{IO}}[\![\lambda x.\,\Omega]\!]_p$ and $\mathcal{O}_{\mathtt{IO}}[\![\Omega]\!]_p$ are indistinguishable. As before, the former process can exhibit an input transition $\mathcal{O}_{\mathtt{IO}}[\![\lambda x.\,\Omega]\!]_p \xrightarrow{p(x,q)} \mathcal{O}_{\mathtt{IO}}[\![\Omega]\!]_q$. However, now $\mathcal{O}_{\mathtt{IO}}[\![\Omega]\!]_p$ has a matching input transition, because when $p \leftrightarrow \bar{p}_1$ is the I-O wire $p \underset{\mathtt{IO}}{\leftrightarrow} \bar{p}_1$, the input at p is not guarded. The derivative of the input $p(y,q)$ is

$$\begin{aligned}&\boldsymbol{\nu} p_0\,(p_0(x,q)\colon \mathcal{O}_{\mathtt{IO}}[\![x\,x]\!]_q\\ &\qquad \mid \bar{p}_0(x_1,p_1)\colon (!x_1(r_1).\,\mathcal{O}_{\mathtt{IO}}[\![\lambda x.\,x\,x]\!]_{r_1}\\ &\qquad \mid \bar{p}_1(x_2,p_2)\colon (x_2 \underset{\mathtt{IO}}{\leftrightarrow} \bar{y} \mid q \underset{\mathtt{IO}}{\leftrightarrow} \bar{p}_2)))\\ &= \boldsymbol{\nu} p_0\,(p_0(x,q)\colon \mathcal{O}_{\mathtt{IO}}[\![x\,x]\!]_q\\ &\qquad\quad \mid \bar{p}_0(x_1,p_1)\colon (!x_1(r_1).\,\mathcal{O}_{\mathtt{IO}}[\![\lambda x.\,x\,x]\!]_{r_1}\\ &\qquad\quad \mid \bar{p}_1(x_2,p_2)\colon (!x_2(r_2).\,\mathcal{O}_{\mathtt{IO}}[\![y]\!]_{r_2} \mid q \underset{\mathtt{IO}}{\leftrightarrow} \bar{p}_2)))\\ &\equiv \mathcal{O}_{\mathtt{IO}}[\![\Omega\,y]\!]_q\end{aligned}$$

(exploiting the definitions of $\mathcal{O}_{\mathtt{IO}}[\![y]\!]_{r_2}$ and $x_2 \underset{\mathtt{IO}}{\leftrightarrow} \bar{y}$). In a similar manner, one then shows that $\mathcal{O}_{\mathtt{IO}}[\![\Omega]\!]_q$ and $\mathcal{O}_{\mathtt{IO}}[\![\Omega\,y]\!]_q$ can match each other's transitions, and iteratively so, on the resulting derivatives. These observations are not limited to Ω and $\lambda x.\,\Omega$, but can be said against general unsolvable terms. Below we state these properties as lemmas. Some of the proofs of the lemmas are given in Appendix D.

As indicated in the preceding example, in $\mathcal{O}_{\mathtt{OI}}$, a term $\mathcal{O}_{\mathtt{OI}}[\![M]\!]_p$ can perform an input transition if and only if M is, or may reduce to, a function, say $M = \lambda x.\,M'$, and the input action intuitively corresponds to consuming the outermost 'λx'.

Lemma 7.1. *Let M be an unsolvable term of order n, where $0 < n \leq \omega$. Then $\mathcal{O}_{\mathtt{OI}}[\![M]\!]_p$ can do a weak input transition at p. Moreover, if $\mathcal{O}_{\mathtt{OI}}[\![M]\!]_p \xRightarrow{p(x,q)} P$, then there exists N such that $P \gtrsim \mathcal{O}_{\mathtt{OI}}[\![N]\!]_q$ and N is an unsolvable of order $n-1$ (under the assumption $\omega - 1 = \omega$).*

In addition, a process $\mathcal{O}_{\mathtt{OI}}[\![M]\!]_p$ is bisimilar to $\mathbf{0}$ iff M is an unsolvable of order 0.

Lemma 7.2. *Let M be an unsolvable term of order 0. Then the only action $\mathcal{O}_{\mathtt{OI}}[\![M]\!]_p$ can do is a τ-action.*

Therefore, we have:

Lemma 7.3. *Let M and N be unsolvables of order m and n respectively, where $0 \leq m, n \leq \omega$. Then $\mathcal{O}_{\mathtt{OI}}[\![M]\!]_p \approx \mathcal{O}_{\mathtt{OI}}[\![N]\!]_p$ iff $m = n$.*

Proof. The only if direction is proved by contraposition. To see that unsolvable terms M and N with different orders are distinguished, we just need to count the number of consecutive inputs that $\mathcal{O}_{\mathtt{OI}}[\![M]\!]_p$ and $\mathcal{O}_{\mathtt{OI}}[\![N]\!]_p$ can do. By Lemmas 7.1 and 7.2, it follows that $\mathcal{O}_{\mathtt{OI}}[\![M]\!]_p$ can do n consecutive weak input transitions if the order of M is n; if $n = \omega$, the number of consecutive inputs that $\mathcal{O}_{\mathtt{OI}}[\![M]\!]_p$ can do is unbounded.

We now prove the if direction. We first prove the case for $n = m = 0$, and use that result to give the proof for arbitrary n.

For the case $n = m = 0$, we show that the relation $\mathcal{R}$ defined as

$$\bigcup_p \{(\mathcal{O}_{\mathtt{OI}}[\![M]\!]_p, \mathcal{O}_{\mathtt{OI}}[\![N]\!]_p) \mid \ M, N \text{ unsolvables of order } 0\}$$

is a bisimulation up-to expansion. Suppose that $\mathcal{O}_{\mathtt{OI}}[\![M]\!]_p \ \mathcal{R} \ \mathcal{O}_{\mathtt{OI}}[\![N]\!]_p$. If $\mathcal{O}_{\mathtt{OI}}[\![M]\!]_p$ makes a transition $\mathcal{O}_{\mathtt{OI}}[\![M]\!]_p \xrightarrow{\mu} P$, then $\mu = \tau$ by Lemma 7.2. Hence, we have $P \gtrsim \mathcal{O}[\![M']\!]_p$ with $M \to M'$ by Lemma 5.3. Since a term obtained by reducing an unsolvable term of order 0 must also be an unsolvable of order 0, we have $P \gtrsim \mathcal{O}_{\mathtt{OI}}[\![M']\!]_p \ \mathcal{R} \ \mathcal{O}_{\mathtt{OI}}[\![N]\!]_p$. In other words, we can take $\mathcal{O}_{\mathtt{OI}}[\![N]\!]_p \Rightarrow \mathcal{O}_{\mathtt{OI}}[\![N]\!]_p$ as the matching transition.

To conclude we show that the relation $\mathcal{R}$ defined as

$$\bigcup_p \{(\mathcal{O}_{\mathtt{OI}}[\![M]\!]_p, \mathcal{O}_{\mathtt{OI}}[\![N]\!]_p) \mid \ M, N \text{ are unsolvables of the same order}\}$$

is a bisimulation up-to expansion. Suppose that $\mathcal{O}_{\mathtt{OI}}[\![M]\!]_p \ \mathcal{R} \ \mathcal{O}_{\mathtt{OI}}[\![N]\!]_p$. The order 0 case is exactly what we proved above, so let us assume that M and N are unsolvable terms whose order is $n \neq 0$ (where n may be ω). Assume that $\mathcal{O}_{\mathtt{OI}}[\![M]\!]_p$ makes a transition $\mathcal{O}_{\mathtt{OI}}[\![M]\!]_p \xrightarrow{\mu} P$. If $\mu = \tau$, then we can reason as we did for the order 0 case. The only other possibility is the case where $\mu = p(x, q)$ with x, q being fresh. Then, by Lemma 7.1, there exists M' such that $P \gtrsim \mathcal{O}_{\mathtt{OI}}[\![M']\!]_q$ and M' is an unsolvable of order $n-1$. By Lemma 7.1, we have $\mathcal{O}_{\mathtt{OI}}[\![N]\!]_p \xRightarrow{p(x,q)} Q \gtrsim \mathcal{O}_{\mathtt{OI}}[\![N']\!]_q$ for some unsolvable term whose order coincides with that of M'. Since M' and N' are unsolvables of the same order, we have $\mathcal{O}_{\mathtt{OI}}[\![M']\!]_q \ \mathcal{R} \ \mathcal{O}_{\mathtt{OI}}[\![N']\!]_q$. □

We have discussed above why, in contrast, $\mathcal{O}_{\mathtt{IO}}$ equates $\lambda x.\, \Omega$ and Ω. Similarly, $\mathcal{O}_{\mathtt{IO}}$ equates all the unsolvable terms.

Lemma 7.4.

(1) *If M is an unsolvable of order 0, then $\mathcal{O}_{\mathtt{IO}}[\![M]\!]_p$ can do an input at p. Moreover, if $\mathcal{O}_{\mathtt{IO}}[\![M]\!]_p \xrightarrow{p(x,q)} P$, then $P \gtrsim \mathcal{O}_{\mathtt{IO}}[\![M\, x]\!]_q$.*

(2) *If M is unsolvable, then $\mathcal{O}_{\mathtt{IO}}[\![M]\!]_p$ can do an input at p. Moreover, if $\mathcal{O}_{\mathtt{IO}}[\![M]\!]_p \xrightarrow{p(x,q)} P$, then there exists an unsolvable term M' such that $P \gtrsim \mathcal{O}_{\mathtt{IO}}[\![M']\!]_q$.*

The case (1) of Lemma 7.4 is used within the proof of case (2); the order 0 case is spelled out since it is the most interesting case in which an input action does not merely correspond to consuming a λ.

Lemma 7.5. *For any unsolvable term M, we have $\mathcal{O}_{\mathtt{IO}}[\![M]\!]_p \approx \mathcal{O}_{\mathtt{IO}}[\![\Omega]\!]_p$.*

Proof. We show that the relation $\mathcal{R}$ defined as

$$\bigcup_p \{(\mathcal{O}_{\mathtt{IO}}[\![M]\!]_p, \mathcal{O}_{\mathtt{IO}}[\![N]\!]_p) \mid M \text{ and } N \text{ are unsolvables}\}$$

is a bisimulation up-to expansion.

Suppose that $\mathcal{O}_{\mathtt{IO}}[\![M]\!]_p \mathcal{R} \mathcal{O}_{\mathtt{IO}}[\![N]\!]_p$. We consider the case where $\mathcal{O}_{\mathtt{IO}}[\![M]\!]_p$ makes the challenge; we omit the opposite case as it is symmetrical. By Lemmas 5.6 and 5.4, the only actions $\mathcal{O}_{\mathtt{IO}}[\![M]\!]_p$ can do is either a τ-action or an input at p.

If $\mathcal{O}_{\mathtt{IO}}[\![M]\!]_p \xrightarrow{\tau} P$ then we can take $\mathcal{O}_{\mathtt{IO}}[\![N]\!]_p \Rightarrow \mathcal{O}_{\mathtt{IO}}[\![N]\!]_p$ as the matching transition because we have $P \gtrsim \mathcal{O}_{\mathtt{IO}}[\![M']\!]_p \mathcal{R} \mathcal{O}_{\mathtt{IO}}[\![N]\!]_p$ for some unsolvable term M' such that $M \to M'$ by Lemma 5.3.

Assume that $\mathcal{O}_{\mathtt{IO}}[\![M]\!]_p \xrightarrow{p(x,q)} P$. Then, thanks to 2 of Lemma 7.4, there exists an unsolvable term M' such that $P \gtrsim \mathcal{O}_{\mathtt{IO}}[\![M']\!]_q$. Similarly, by 2 of Lemma 7.4, we have $\mathcal{O}_{\mathtt{IO}}[\![N]\!]_p \xrightarrow{p(x,q)}\gtrsim \mathcal{O}_{\mathtt{IO}}[\![N']\!]_q$ for some unsolvable term N'. The claim follows because $\mathcal{O}_{\mathtt{IO}}[\![M']\!]_q \mathcal{R} \mathcal{O}_{\mathtt{IO}}[\![N']\!]_q$. □

7.2. **Full abstraction proofs.** The goal of this subsection is to prove the following two theorems:

Theorem 7.6 (Full abstraction for LT)**.** *$LT(M) = LT(N)$ if and only if $\mathcal{A}_{\mathtt{OI}}[\![M]\!] \approx \mathcal{A}_{\mathtt{OI}}[\![N]\!]$.*

Theorem 7.7 (Full abstraction for BT)**.** *$BT(M) = BT(N)$ if and only if $\mathcal{A}_{\mathtt{IO}}[\![M]\!] \approx \mathcal{A}_{\mathtt{IO}}[\![N]\!]$.*

The proofs exploit work by Sangiorgi and Xu [SX18], which sets conditions for obtaining full abstraction with respect to LTs and BTs in an encoding of the λ-calculus into a process calculus. Our proofs go through each such condition, showing that it is satisfied.

The rest of this sections is organised as follows. We first review the general conditions given in [SX18] (Section 7.2.1). Then we show that $\mathcal{A}_{\mathtt{OI}}$ satisfies the conditions for LTs and $\mathcal{A}_{\mathtt{IO}}$ satisfies that for BTs (Section 7.2.2).

7.2.1. *General conditions.* Here we present a simplified version of the conditions given in [SX18], tailored to our needs, where the calculus is Iπ, and the relations to be considered are bisimilarity and the expansion relation for Iπ (the conditions in [SX18] are parametric with respect to the calculus and its behavioural relations). We also show that some conditions can be proved at the level of the abstract (optimised) encoding $\mathcal{O}$.

We begin with reviewing some needed terminology. An *abstraction context* of an encoding $\mathcal{E}$ is the context obtained by encoding the λ-calculus context $\lambda x.\,[\cdot]$, that is, $\mathcal{E}[\![\lambda x.\,[\cdot]]\!]$ (assuming that λ-calculus holes are mapped onto identical process holes); similarly, a *variable context* of $\mathcal{E}$ is the encoding of the λ-calculus n-hole context $x\,[\cdot]_1 \cdots [\cdot]_n$. A context is *guarded*

if each hole appears underneath some proper (i.e., non-permeable) prefix. A n-hole context C *has an inverse with respect to* $\lesssim$ if, for every $i \in \{1, \ldots, n\}$, there exists a π-context D_i such that $D_i[C[\widetilde{A}]] \gtrsim \overline{a}(\widetilde{b}).\, b(z).\, A_i\langle z\rangle$ for fresh names a, b, z such that $b \in \widetilde{b}$. Intuitively, D_i allows us to bring up the content of the $i-$th hole of the context C, which can be reached and activated using the prefixes at a and b.

Theorem 7.8 [SX18]**.** *Let $\mathcal{E}$ be an encoding of the λ-calculus into $I\pi$. Suppose that the encoding satisfies the following conditions.*

(1) *The variable contexts of $\mathcal{E}$ are guarded;*
(2) *The abstraction and variable contexts of $\mathcal{E}$ have an inverse with respect to $\lesssim$, provided that the every abstraction F that fills the holes of the context satisfies $F = \mathcal{E}[\![M]\!]$ for some λ-term M;*
(3) *$\mathcal{E}$ and $\lesssim$ validate the β rule;*
(4) *If M is an unsolvable of order 0 then $\mathcal{E}[\![M]\!] \approx \mathcal{E}[\![\Omega]\!]$;*
(5) *The terms $\mathcal{E}[\![\Omega]\!]$, $\mathcal{E}[\![x\,\widetilde{M}]\!]$, $\mathcal{E}[\![x\,\widetilde{M'}]\!]$, and $\mathcal{E}[\![y\,\widetilde{M''}]\!]$ are pairwise unrelated by $\approx$, assuming that $x \neq y$ and that tuples $\widetilde{M}$ and $\widetilde{M'}$ have different lengths.*

Then we have:

(LT): *if*
  (1) *M, N unsolvable of order ω implies that $\mathcal{E}[\![M]\!] \approx \mathcal{E}[\![N]\!]$ and*
  (2) *for any M the term $\mathcal{E}[\![\lambda x.\, M]\!]$ is unrelated by $\approx$ to $\mathcal{E}[\![\Omega]\!]$ and to any term of the form $\mathcal{E}[\![x\,\widetilde{M}]\!]$,*

  then $\mathcal{E}$ and $\approx$ are fully abstract for LT equality;

(BT): *if*
  (1) *M solvable implies that the term $\mathcal{E}[\![\lambda x.\, M]\!]$ is unrelated by $\approx$ to $\mathcal{E}[\![\Omega]\!]$ and to any term of the form $\mathcal{E}[\![x\,\widetilde{M}]\!]$, and*
  (2) *$\mathcal{E}[\![M]\!] \approx \mathcal{E}[\![\Omega]\!]$ whenever M is unsolvable of order ω,*

  then $\mathcal{E}$ and $\approx$ are fully abstract for BT equality.

Remark 7.9. The condition (2) is weaker than the original condition used in [SX18] (i.e., the new condition does not imply the old one). The original condition did not have the assumption that 'abstractions that fills the hole must be encodings of λ-terms'. We need this condition because the encodings of abstraction and variable use permeable prefixes, and these only behave well with I-/O-respectful processes, such as those resulting from the encoding of λ-terms (Lemma 4.2). However, this does not cause a problem because, in [SX18], whenever this condition about the inverse context is used, the holes are indeed filled with encodings of λ-terms.

The existence of the inverse context can be proved at the abstract level (i.e., for $\mathcal{O}$ and $\mathcal{A}$), and hence, it need not be proved independently for $\mathcal{A}_{\mathtt{OI}}$ and $\mathcal{A}_{\mathtt{IO}}$.

Lemma 7.10. *The abstraction and variable contexts of $\mathcal{O}$ have inverse with respect to $\lesssim$,* under the assumption that the every abstraction F that fills the context satisfies $F = \mathcal{O}[\![M]\!]$ for some λ-term M.

The proof is by simple algebraic reasonings such as I- and O-respectfulness; details are given in Appendix D.2.

7.2.2. *Checking conditions for LT and BT.* Thanks to the general conditions we described, to prove the full-abstraction results, we only need to show that $\mathcal{A}_{\mathtt{OI}}$ and $\mathcal{A}_{\mathtt{IO}}$ satisfy the required conditions. We first show that $\mathcal{A}_{\mathtt{OI}}$ indeed satisfies the conditions for LT.

Theorem 7.6 (Full abstraction for LT)**.** *$LT(M) = LT(N)$ if and only if $\mathcal{A}_{\mathtt{OI}}[\![M]\!] \approx \mathcal{A}_{\mathtt{OI}}[\![N]\!]$.*

Proof. We check the conditions given in Theorem 7.8. It suffices to give the proof for the optimised encoding $\mathcal{O}_{\mathtt{OI}}$.

Some conditions are trivial or have already been checked. The variable contexts of $\mathcal{O}_{\mathtt{OI}}$ is clearly guarded, the condition about the inverse context is Lemma 7.10, the validity of β is Lemma 5.1 and we have proved that unsolvable terms of order 0 are equated (Lemma 7.3).

We first see that $\mathcal{O}_{\mathtt{OI}}[\![\Omega]\!]$, $\mathcal{O}_{\mathtt{OI}}[\![x\,\widetilde{M}]\!]$, $\mathcal{O}_{\mathtt{OI}}[\![x\,\widetilde{M'}]\!]$ and $\mathcal{O}_{\mathtt{OI}}[\![y\,\widetilde{M''}]\!]$ are pairwise unrelated under the assumption that $x \neq y$ and $\widetilde{M}$ and $\widetilde{M'}$ have different lengths. Since $\mathcal{O}_{\mathtt{OI}}[\![\Omega]\!]_p$ cannot do an output action (Lemma 5.6), this process is different from the rest of the processes. It is also obvious that $\mathcal{O}[\![y\,\widetilde{M''}]\!]_p$ is different from $\mathcal{O}_{\mathtt{OI}}[\![x\,\widetilde{M}]\!]$ and $\mathcal{O}_{\mathtt{OI}}[\![x\,\widetilde{M'}]\!]$. We are left with checking that $\mathcal{O}_{\mathtt{OI}}[\![x\,\widetilde{M}]\!]$ and $\mathcal{O}_{\mathtt{OI}}[\![x\,\widetilde{M'}]\!]$ are not bisimilar. Let $m \stackrel{\text{def}}{=} |\widetilde{M}|$ and $n \stackrel{\text{def}}{=} |\widetilde{M'}|$, and without loss of generality, assume that $m < n$. Then $\mathcal{O}_{\mathtt{OI}}[\![x\,\widetilde{M}]\!]_p$ can do an input at p after $m+2$ outputs. More concretely, we have $\mathcal{O}_{\mathtt{OI}}[\![x\,\widetilde{M}]\!]_p \xrightarrow{\overline{x}(p_0)} \xrightarrow{\overline{p_0}(x_1,p_1)} \cdots \xrightarrow{\overline{p_n}(x_{n+1},p_{n+1})} \xrightarrow{p(y,q)} P$ for some P. However, $\mathcal{O}_{\mathtt{OI}}[\![x\,\widetilde{M'}]\!]_p$ cannot do an input at p after $m+2$ outputs, and thus these two processes are not bisimilar.

Now we look at the conditions in **(LT)** of Theorem 7.8. The condition 1 holds because we have proved that unsolvable terms of order ω are equated (Lemma 7.3). It remains to show the condition (2). Clearly, $\mathcal{O}_{\mathtt{IO}}[\![\lambda x.\, M]\!]_p$ is not bisimilar to $\mathcal{O}_{\mathtt{IO}}[\![\Omega]\!]_p$ or to the encoding of any term of the form $x\,\widetilde{M}$ because $\mathcal{O}_{\mathtt{IO}}[\![\lambda x.\, M]\!]_p$ can do an input at p, but the others cannot. □

We conclude by proving that $\mathcal{A}_{\mathtt{IO}}$ is fully abstract with respect to BTs.

Theorem 7.7 (Full abstraction for BT)**.** *$BT(M) = BT(N)$ if and only if $\mathcal{A}_{\mathtt{IO}}[\![M]\!] \approx \mathcal{A}_{\mathtt{IO}}[\![N]\!]$.*

Proof. Conditions 1-5 of Theorem 7.8 are checked similarly as in the proof of the previous theorem. One difference is that for the encoding $\mathcal{O}_{\mathtt{IO}}$ we use Lemma 7.5 to say that unsolvable terms of order 0 are equated. Another (minor) difference is how to show that $x\,\widetilde{M}$ and $x\,\widetilde{N}$ are different when the length of $\widetilde{M}$, say m, and the length of $\widetilde{N}$, say n, are different. Without loss of generality suppose that $m < n$. Then $\mathcal{O}_{\mathtt{IO}}[\![x\,\widetilde{M}]\!]_p$ can only do $m+1$ consecutive outputs:

$$\begin{aligned} &\mathcal{O}_{\mathtt{IO}}[\![x\,\widetilde{M}]\!]_p \xrightarrow{\overline{x}(p_0)} \xrightarrow{\overline{p_0}(x_1,p_1)} \cdots \xrightarrow{\overline{p_n}(x_{n+1},p_{n+1})} \\ &\gtrsim\ !x(r_1).\,\mathcal{O}_{\mathtt{IO}}[\![M_1]\!]_{r_1} \mid \cdots \mid !x(r_n).\,\mathcal{O}_{\mathtt{IO}}[\![M_n]\!]_{r_n} \mid p \underset{\mathtt{IO}}{\leftrightarrow} \bar{p}_n \end{aligned} \tag{7.1}$$

because $p \underset{\mathtt{IO}}{\leftrightarrow} \bar{p}_n$ cannot make an output action. On the other hand, $\mathcal{O}_{\mathtt{IO}}[\![x\,\widetilde{N}]\!]_p$ can do $n+1$ consecutive outputs.

Now we check the conditions in **(BT)** of Theorem 7.8. Condition (2) holds because we proved that all the unsolvable terms are equated (Lemma 7.5). So we are left with checking condition (1). Let M be a solvable term. We need to check that $\lambda x.\, M$ is unrelated to Ω and any term of the form $w\,\widetilde{N}$. Since M is solvable, we have $\lambda x.\, M \Longrightarrow \lambda x.\, \lambda\widetilde{y}.\, z\,\widetilde{M}$, where $\widetilde{y}$ and $\widetilde{M}$ are possibly empty. Since the encoding is valid with respect to β-reduction (Lemma 5.1),

it suffices to show that the encoding of $\lambda x.\, \lambda\widetilde{y}.\, z\widetilde{M}$ is not bisimilar with the encoding of Ω and $w\,\widetilde{N}$. The process $\mathcal{O}_{\mathtt{IO}}[\![\lambda x.\, \lambda\widetilde{y}.\, z\widetilde{M}\,]\!]_p$ can do an output on z after doing some inputs that correspond to the leading λs. However, $\mathcal{O}_{\mathtt{IO}}[\![\Omega]\!]_p$ cannot do an output action even after some τ or input actions . Hence, the encoding of $\lambda x.\, M$ and Ω are not bisimilar.

Finally, we show that the encodings of $\lambda x.\, \lambda\widetilde{y}.\, z\,\widetilde{M}$ and $w\,\widetilde{N}$ are not related by $\approx$. First, note that z needs to be a free variable and equal to w, otherwise $\mathcal{O}_{\mathtt{IO}}[\![\lambda x.\, \lambda\widetilde{y}.\, z\,\widetilde{M}]\!]_p$ is not bisimilar with $\mathcal{O}_{\mathtt{IO}}[\![w\,\widetilde{N}]\!]_p$. Second, $\widetilde{N}$ and $\widetilde{M}$ must have the same length. Otherwise we can show that the two processes are not bisimilar by investigating the number of outputs that the two process can do as we did in (7.1); the fact that we have a leading λ in $\lambda x.\, \lambda\widetilde{y}.\, w\,\widetilde{M}$ does not change the argument, as w must be a free variable and λx is encoded as a permeable input. So, we need to show that $\mathcal{O}_{\mathtt{IO}}[\![\lambda x\widetilde{y}.\, w\,\widetilde{M}]\!]_p$ and $\mathcal{O}_{\mathtt{IO}}[\![w\,\widetilde{N}]\!]_p$ are not equated when $\widetilde{M}$ and $\widetilde{N}$ have the same length. Observe that $\mathcal{O}_{\mathtt{IO}}[\![\lambda x\widetilde{y}.\, w\,\widetilde{M}]\!]_p \xrightarrow{p(x,q)} \gtrsim \mathcal{O}_{\mathtt{IO}}[\![\lambda\widetilde{y}.\, w\,\widetilde{M}]\!]_q$. The only matching transition $\mathcal{O}_{\mathtt{IO}}[\![w\,\widetilde{N}]\!]_p$ can do is

$$\begin{aligned}\mathcal{O}_{\mathtt{IO}}[\![w\,\widetilde{N}]\!]_p \xrightarrow{p(x,q)} &\equiv \bar{w}(p_0)\colon \mathcal{O}^n\langle p_0, q, \mathcal{O}_{\mathtt{IO}}[\![N_1]\!], \ldots, \mathcal{O}_{\mathtt{IO}}[\![N_n]\!], \mathcal{O}_{\mathtt{IO}}[\![x]\!]\rangle\\ &= \mathcal{O}_{\mathtt{IO}}[\![w\,\widetilde{N}\;x]\!]_q\end{aligned}$$

where $\widetilde{N} = N_1, \ldots, N_n$ because

$$\begin{aligned}p \underset{\mathtt{IO}}{\leftrightarrow} \bar{p}_n \xrightarrow{p(x,q)} &\bar{p}_n(x_{n+1}, p_{n+1})\colon (q \underset{\mathtt{IO}}{\leftrightarrow} \bar{p}_{n+1} \mid x_{n+1} \underset{\mathtt{IO}}{\leftrightarrow} \bar{x})\\ &= \bar{p}_n(x_{n+1}, p_{n+1})\colon (q \underset{\mathtt{IO}}{\leftrightarrow} \bar{p}_{n+1} \mid !x_{n+1}(r_{n+1}).\, \mathcal{O}_{\mathtt{IO}}[\![x]\!]_{r_{n+1}}).\end{aligned}$$

Once again, we look at the number of consecutive outputs these process can do. The process $\mathcal{O}_{\mathtt{IO}}[\![\lambda\widetilde{y}.\, w\,\widetilde{M}]\!]_q$ can do $n+1$ outputs whereas $\mathcal{O}_{\mathtt{IO}}[\![w\,\widetilde{M}\;x]\!]_q$ can do $n+2$ outputs. We therefore conclude that $\lambda x.\, \lambda\widetilde{y}.\, z\,\widetilde{M}$ can never be equated to a term of the form $w\,\widetilde{N}$. □

Remark 7.11. Neither $\mathcal{A}_{\mathtt{IO}}$ nor $\mathcal{A}_{\mathtt{OI}}$ validates the η-rule (i.e., the λ-theories induced are not extensional); this follows from Theorems 7.6 and 7.7 (specifically, conditions 2 of **(LT)** and 1 of **(BT)** mentioned in their proofs, respectively).

8. Full Abstraction for $\mathrm{BT}_{\eta\infty}$s

In this section we show that the encoding $\mathcal{A}_{\mathtt{P}}$ obtained by instantiating the wires of the abstract encoding $\mathcal{A}$ of Section 4 with the parallel wires (the P wires) yields an encoding that is fully abstract with respect to $\mathrm{BT}_{\eta\infty}$s (Böhm trees with infinite η-expansion).

We begin by showing that $\mathcal{A}_{\mathtt{P}}$ induces an *extensional* λ-theory. The result is a useful stepping stone towards the full abstraction result with respect to $\mathrm{BT}_{\eta\infty}$, and may help the readers to understand the differences with respect to the other two concrete encodings examined in Section 7. As we know (Section 6) that $\mathcal{A}_{\mathtt{P}}$ induces a λ-theory, we remain to check the validity of η-expansion.

Theorem 8.1. *For every M and $x \notin \mathsf{fv}(M)$, we have $\mathcal{A}_{\mathtt{P}}[\![M]\!]_p \lesssim \mathcal{A}_{\mathtt{P}}[\![\lambda x.\, M\,x]\!]_p$.*

Proof. The process $\mathcal{A}_{\mathtt{P}}[\![\lambda x.\, M\,x]\!]_p$ is

$$p(x,q)\colon \boldsymbol{\nu} r\, (\mathcal{A}_{\mathtt{P}}[\![M]\!]_r \mid \bar{r}(x',q')\colon (!x'(r').\, \mathcal{A}_{\mathtt{P}}[\![x]\!]_{r'} \mid q \underset{\mathtt{P}}{\leftrightarrow} \bar{q}')).$$

As $!x'(r').\,\mathcal{A}_{\mathtt{P}}[\![x]\!]_{r'} = x' \underset{\mathtt{P}}{\leftrightarrow} \bar{x}$, we have:

$$\begin{aligned}\mathcal{A}_{\mathtt{P}}[\![\lambda x.\, M\, x]\!]_p &\equiv \boldsymbol{\nu} r\, (\mathcal{A}_{\mathtt{P}}[\![M]\!]_r \mid p(x,q)\colon \bar{r}(x',q')\colon (x' \underset{\mathtt{P}}{\leftrightarrow} \bar{x} \mid q \underset{\mathtt{P}}{\leftrightarrow} \bar{q}'))\\ &= \boldsymbol{\nu} r\, (\mathcal{A}_{\mathtt{P}}[\![M]\!]_r \mid p \underset{\mathtt{P}}{\leftrightarrow} \bar{r}) \gtrsim \mathcal{A}_{\mathtt{P}}[\![M]\!]_p\end{aligned}$$

using Lemma 4.2. □

The above result relies on the use of permeable prefixes, both in the encoding of λ-abstraction, and within the P wires.

Corollary 8.2. *Let* $=_\pi \stackrel{def}{=} \{(M,N) \mid \mathcal{A}_{\mathtt{P}}[\![M]\!] \approx \mathcal{A}_{\mathtt{P}}[\![N]\!]\}$. *Then* $=_\pi$ *is an extensional* λ*-theory; that is, a congruence on* λ*-terms that contains* β *and* η*-equivalence.*

We are now ready to prove that $\mathcal{A}_{\mathtt{P}}$ is fully abstract with respect to $\mathrm{BT}_{\eta\infty}$s. We focus on completeness: if $\mathrm{BT}_{\eta\infty}(M) = \mathrm{BT}_{\eta\infty}(N)$ then $\mathcal{A}_{\mathtt{P}}[\![M]\!] \approx \mathcal{A}_{\mathtt{P}}[\![N]\!]$. Soundness will then be essentially derived from completeness, as $\mathrm{BT}_{\eta\infty}$ equality is the maximal consistent sensible λ-theory (see e.g. [Bar84]). To show completeness, we rely on the 'unique solution of equation' technique, reviewed in Section 2.2.1.

Remark 8.3 (Unique solutions versus up-to techniques)**.** Results about encodings of λ-calculus into process calculi, in previous sections of this paper and in the literature, usually employ up-to techniques for bisimilarity, notably *up-to context and expansion*. In the techniques, expansion is used to manipulate the derivatives of two transitions so to bring up a common context. Such techniques do not seem powerful enough for $\mathrm{BT}_{\eta\infty}$. The reason is that some of the required transformations would violate expansion (i.e., they would require to replace a term by a 'less efficient' one), for instance 'η-expanding' a term $\mathcal{A}_{\mathtt{P}}[\![z]\!]_p$ into $\mathcal{A}_{\mathtt{P}}[\![\lambda\, y.\, z\, y]\!]_p$. A similar problem has been observed in the case of Milner's call-by-value encoding [DHS22].

Suppose $\mathcal{R}$ is a $\mathrm{BT}_{\eta\infty}$-bisimulation (Definition 2.1). We define a (possibly infinite) system of equations $\mathcal{E}_{\mathcal{R}}$, solutions of which will be obtained from the encodings of the pairs in $\mathcal{R}$. There is one equation for each pair $(M,N) \in \mathcal{R}$. We describe how each equation is defined, following the clauses of $\mathrm{BT}_{\eta\infty}$-bisimulation. Take $(M,N) \in \mathcal{R}$ and assume $\widetilde{y} = \mathsf{fv}(M,N)$.

(1) If M and N are unsolvable, then, for the right-hand side of the equation, we pick a non-divergent process that is bisimilar to the encoding of Ω:

$$X_{M,N}\, \widetilde{y} = \mathtt{K}_\Omega$$

For instance, we may choose $\mathtt{K}_\Omega \stackrel{\mathrm{def}}{=} (p)\, p(x,q)\colon \mathtt{K}_\Omega\langle q\rangle$.

(2) If $M \Longrightarrow_{\mathtt{h}} \lambda x_1 \ldots x_{l+m}.\, z\, M_1 \cdots M_{n+m}$ and $N \Longrightarrow_{\mathtt{h}} \lambda x_1 \ldots x_l.\, z\, N_1 \cdots N_n$, then the equation is:

$$\begin{aligned}&X_{M,N}\, \widetilde{y}\, p \stackrel{\mathrm{def}}{=}\\ &p(x_1,p_1)\colon \cdots p_{l+m-1}(x_{l+m},p_{l+m})\colon \bar{z}(w,q)\colon\\ &\mathcal{O}^{n+m}_{\mathtt{P}} \left\langle \begin{matrix} q, p_{l+m}, X_{M_1,N_1}\langle \widetilde{y}_1\rangle, \ldots, X_{M_n,N_n}\langle \widetilde{y}_n\rangle, \\ X_{M_{n+1},x_{l+1}}\langle \widetilde{y}_{n+1}\rangle, \ldots, X_{M_{n+m},x_{l+m}}\langle \widetilde{y}_{n+m}\rangle \end{matrix} \right\rangle\end{aligned}$$

where $\widetilde{y}_i = \mathsf{fv}(M_i,N_i)$ for $1 \le i \le n$, $\widetilde{y}_i = \mathsf{fv}(M_i, x_{i-n+l})$ for $n+1 \le i \le n+m$. and where $\mathcal{O}^r_{\mathtt{P}}$ is the instantiation with P wires of $\mathcal{O}^r$ in Figure 3.

(3) For the case symmetric to (2), where N reduces to a head normal form with more leading λ-abstractions, the equation is defined similarly to (2).

In (1), the use of a divergent-free term $\mathtt{K}_\Omega$ allows us to meet the condition about divergence of the unique-solution technique. The right-hand side of (2) intuitively amounts to having, as a body of the equation, the process $\mathcal{O}_{\mathtt{P}}[\![\lambda x_1 \dots x_{l+m}.\, z\, X_{M_1,N_1} \cdots X_{M_{n+m},x_{l+m}}]\!]$.

We show that the system $\mathcal{E}_{\mathcal{R}}$ of equations has the desired solutions.

Lemma 8.4. *For any M unsolvable, we have: $\mathcal{O}_{\mathtt{P}}[\![M]\!]_p \approx \mathcal{O}_{\mathtt{P}}[\![\Omega]\!]_p \approx \mathtt{K}_\Omega\langle p\rangle$.*

Proof. The reasoning is similar to that in Lemma 7.5 (whose proof is given in Appendix D.1). □

Lemma 8.5. *Let $\mathcal{R}$ be a $BT_{\eta\infty}$-bisimulation and $\mathcal{E}_{\mathcal{R}}$ be the system of equations defined from $\mathcal{R}$ as above. For each $(M,N) \in \mathcal{R}$, we define $F_{M,N} \stackrel{def}{=} (\widetilde{x}, p)\ \mathcal{O}_{\mathtt{P}}[\![M]\!]_p$ and $G_{M,N} \stackrel{def}{=} (\widetilde{x}, p)\ \mathcal{O}_{\mathtt{P}}[\![N]\!]_p$, where $\widetilde{x} = \mathsf{fv}(M,N)$. Then $\{F_{M,N}\}_{(M,N)\in\mathcal{R}}$ and $\{G_{M,N}\}_{(M,N)\in\mathcal{R}}$ are solutions of $\mathcal{E}_{\mathcal{R}}$.*

Proof. Take $(M,N) \in \mathcal{R}$. There are three cases to consider following Definition 2.1.

If M and N are unsolvables then we have

$$F_{M,N} \approx (\widetilde{y}, p)\ \mathcal{O}_{\mathtt{P}}[\![\Omega]\!]_p \approx G_{M,N}$$

by Lemma 8.4.

If the second clause of Definition 2.1 holds, then we have

$$\begin{aligned}
&F_{M,N}\,\widetilde{y}\,p \\
&= \mathcal{O}_{\mathtt{P}}[\![M]\!]_p \\
&\gtrsim \mathcal{O}_{\mathtt{P}}[\![\lambda x_1 \dots x_{l+m}.\, y\, M_1 \cdots M_{n+m}]\!]_p && \text{(Lemma 5.1)} \\
&= p(x_1,p_1)\colon \cdots p_{l+m-1}(x_{l+m},p_{l+m})\colon \bar{y}(w,q)\colon \\
&\quad \mathcal{O}_{\mathtt{P}}^{n+m}\langle q, p_{l+m}, \mathcal{O}_{\mathtt{P}}[\![M_1]\!], \dots, \mathcal{O}_{\mathtt{P}}[\![M_{n+m}]\!]\rangle \\
&= p(x_1,p_1)\colon \cdots p_{l+m-1}(x_{l+m},p_{l+m})\colon \bar{y}(w,q)\colon && \text{(by def. of } F_{M,N}) \\
&\quad \mathcal{O}_{\mathtt{P}}^{n+m}\langle q, p_{l+m}, F_{M_1,N_1}\langle\widetilde{y}_1\rangle, \dots, F_{M_{n+m},x_{l+m}}\langle\widetilde{y}_{n+m}\rangle\rangle \\
&= p(x_1,p_1)\colon \cdots p_{l+m-1}(x_{l+m},p_{l+m})\colon \bar{y}(w,q)\colon \\
&\quad \mathcal{O}_{\mathtt{P}}^{n+m}\langle q, p_{l+m}, X_{M_1,N_1}\langle\widetilde{y}_1\rangle, \dots, X_{M_{n+m},x_{l+m}}\langle\widetilde{y}_{n+m}\rangle\rangle \\
&\quad \{F_{M_1,N_1}/X_{M_1,N_1}, \dots, F_{M_{n+m},x_{l+m}}/X_{M_{n+m},x_{l+m}}\}
\end{aligned}$$

as desired. The proof for $G_{M,N}$ is similar, but we need validity of η-expansion (Theorem 8.1). In detail, we have

$$\begin{aligned}
&G_{M,N}\,\widetilde{y}\,p \\
&= \mathcal{O}_{\mathtt{P}}[\![N]\!]_p \\
&\gtrsim \mathcal{O}_{\mathtt{P}}[\![\lambda x_1 \dots x_l.\, y\, N_1 \cdots N_n]\!]_p && \text{(Lemma 5.1)} \\
&= p(x_1,p_1)\colon \cdots p_{l-1}(x_l,p_l)\colon \bar{y}(w,q)\colon \mathcal{O}_{\mathtt{P}}^{n}\langle q, p_l, \mathcal{O}_{\mathtt{P}}[\![N_1]\!], \dots, \mathcal{O}_{\mathtt{P}}[\![N_n]\!]\rangle \\
&\approx p(x_1,p_1)\colon \cdots p_{l-1}(x_l,p_l)\colon && \text{(Theorem 8.1)} \\
&\quad p_l(x_{l+1},p_{l+1})\colon \cdots p_{l+m-1}(x_{l+m},p_{l+m})\colon \bar{y}(w,q)\colon \\
&\quad \mathcal{O}_{\mathtt{P}}^{n+m}\langle q, p_{l+m}, \mathcal{O}_{\mathtt{P}}[\![N_1]\!], \dots, \mathcal{O}_{\mathtt{P}}[\![N_n]\!], \mathcal{O}_{\mathtt{P}}[\![x_{l+1}]\!], \dots, \mathcal{O}_{\mathtt{P}}[\![x_{l+m}]\!]\rangle
\end{aligned}$$

$$
\begin{aligned}
&= p(x_1, p_1)\colon \cdots p_{l+m-1}(x_{l+m}, p_{l+m})\colon \bar{y}(w, q)\colon && \text{(by def. of } G_{M,N}\text{)}\\
&\quad \mathcal{O}_{\mathtt{P}}^{n+m}\langle q, p_{l+m}, G_{M_1,N_1}\langle\widetilde{y}_1\rangle, \ldots, G_{M_{n+m},x_{l+m}}\langle\widetilde{y}_{n+m}\rangle\rangle\\
&= (p(x_1, p_1)\colon \cdots p_{l+m-1}(x_{l+m}, p_{l+m})\colon \bar{y}(w, q)\colon\\
&\quad \mathcal{O}_{\mathtt{P}}^{n+m}\langle q, p_{l+m}, X_{M_1,N_1}\langle\widetilde{y}_1\rangle, \ldots, X_{M_{n+m},x_{l+m}}\langle\widetilde{y}_{n+m}\rangle\rangle)\\
&\quad \{G_{M_1,N_1}/X_{M_1,N_1}, \ldots, G_{M_{n+m},x_{l+m}}/X_{M_{n+m},x_{l+m}}\}.
\end{aligned}
$$

The case where the third clause of Definition 2.1 holds can be proved similarly. □

We also have to show that the system $\mathcal{E}_{\mathcal{R}}$ of equations we defined has a *unique* solution.

Lemma 8.6. *The system of equations $\mathcal{E}_{\mathcal{R}}$ is guarded and the syntactic solution of $\mathcal{E}_{\mathcal{R}}$ is divergence-free. Therefore, $\mathcal{E}_{\mathcal{R}}$ has a unique solution.*

Proof. The system $\mathcal{E}_{\mathcal{R}}$ is guarded because all the occurrences of a variable in the right-hand side of an equation are underneath a replicated input prefixing. Divergence-freedom follows from the fact that the use of each name (bound or free) is strictly polarised in the sense that a name is either used as an input or as an output. In a strictly polarised setting, no τ-transitions can be performed even after some visible actions because in Iπ only fresh names may be exchanged. □

Theorem 8.7 (Completeness for $\mathrm{BT}_{\eta\infty}$)**.** *If $BT_{\eta\infty}(M) = BT_{\eta\infty}(N)$ then $\mathcal{A}_{\mathtt{P}}[\![M]\!] \approx \mathcal{A}_{\mathtt{P}}[\![N]\!]$.*

Proof. Consider a $\mathrm{BT}_{\eta\infty}$-bisimulation $\mathcal{R}$ that equates M and N. Take the system of equations $\mathcal{E}_{\mathcal{R}}$ corresponding to $\mathcal{R}$ as defined above. By Lemma 8.5, $\mathcal{O}_{\mathtt{P}}[\![M]\!]$ and $\mathcal{O}_{\mathtt{P}}[\![N]\!]$ are (components) of the solutions of $\mathcal{E}_{\mathcal{R}}$. Since the solution is unique (Lemma 8.6), we derive $\mathcal{O}_{\mathtt{P}}[\![M]\!] \approx \mathcal{O}_{\mathtt{P}}[\![N]\!]$. We also have $\mathcal{A}_{\mathtt{P}}[\![M]\!] \approx \mathcal{A}_{\mathtt{P}}[\![N]\!]$ (equivalence on the non-optimised encodings) because of Lemma 5.2. □

Theorem 8.8 (Soundness for $\mathrm{BT}_{\eta\infty}$)**.** *If $\mathcal{A}_{\mathtt{P}}[\![M]\!] \approx \mathcal{A}_{\mathtt{P}}[\![N]\!]$ then $BT_{\eta\infty}(M) = BT_{\eta\infty}(N)$.*

Proof. Let $=_\pi$ be the equivalence induced by $\mathcal{A}_{\mathtt{P}}$ and Iπ bisimilarity. The equivalence $=_\pi$ is a *sensible* λ-theory by Corollary 4.5 and Lemma 8.4. This theory is consistent: for example we have $\mathcal{A}_{\mathtt{P}}[\![x]\!]_p \not\approx \mathcal{A}_{\mathtt{P}}[\![\Omega]\!]_p$. By completeness (Theorem 8.7), it contains $\mathrm{BT}_{\eta\infty}$equality. Then it must be equal to $\mathrm{BT}_{\eta\infty}$ equality because the latter is the maximal consistent sensible λ-theory [Bar84]. □

9. Concluding Remarks

In the paper we have presented a refinement of Milner's original encoding of functions as processes that is parametric on certain abstract components called wires. Whenever wires satisfy a few algebraic properties, the encoding yields a λ-theory. We have studied instantiations of the abstract wires with three kinds of concrete wires, that differ on the direction and/or sequentiality of the control flow produced. We have shown that such instantiations allow us to obtain full abstraction results for LTs, BTs, and $\mathrm{BT}_{\eta\infty}$s, (and hence for λ-models such as P_ω, free lazy Plotkin-Scott-Engeler models and D_∞). In the case of $\mathrm{BT}_{\eta\infty}$, this implies that the encoding validates the η-rule, i.e., it yields an extensional λ-theory.

Following Milner's seminal paper [Mil90], the topic of functions as processes has produced a rich bibliography. Below we comment on the works that seem closest to ours. We have mentioned, in the introduction, related work concerning LTs and BTs. We are unaware of

results about validity of the η-rule, let alone $\mathrm{BT}_{\eta\infty}$, in encodings of functions as processes. The only exception is [BHY01], where a type system for the π-calculus is introduced so to derive full abstraction for an encoding of PCF (which implies that η-expansion for PCF is valid). However, in [BHY01], types are used to constrain process behaviours, so to remain with processes that represent 'sequential functional computations'. Accordingly, the behavioural equivalence for processes is a typed contextual equivalence in which the legal contexts must respect the typing discipline and are therefore 'sequential'. In contrast, in our work η is validated under ordinary (unconstrained) process equivalence in which, for instance, equalities are preserved by arbitrary process contexts. (We still admit polyadic communications and hence a sorting system, for readability — we believe that the same results hold in a monadic setting.)

In the paper we have considered the theory of the pure untyped λ-calculus. Hence, our encodings model the call-by-name reduction strategy. A study of the theory induced by process encodings of the call-by-value strategy is [DHS22].

Our definitions and proofs about encodings of permeable prefixes using wires follows, and is inspired by, encodings of forms of permeable prefixes in asynchronous and localised variants of the π-calculus using forwarders, e.g. [MS04, Yos02]. As commented in the main text, the technicalities are however different, both because our processes are not localised, and because we employ distinct kinds of wires.

We have worked with bisimilarity, as it is the standard behavioural equivalence in Iπ; moreover, we could then use some powerful proof techniques for it (up-to techniques, unique solution of equations). The results presented also hold for other behavioural equivalences (e.g., may testing), since processes encoding functions are confluent. It would be interesting to extend our work to preorders, i.e., looking at preorder relations for λ-trees and λ-models.

In our work, we derived our abstract encoding from Milner's original encoding of functions. It is unclear how to transport the same methodology to other variants of Milner's encoding in the literature, in particular those that closely mimics the CPS translations [San99, Thi97].

We have derived λ-tree equalities, in parametric manner, by different instantiations of the abstract wires. Van Bakel et al. [vBBDdV02] use an intersection type system, parametric with respect to the subtyping relations, to (almost) uniformly characterise λ-tree equalities (the trees considered are those in our paper together with Böhm trees up-to *finite* η-expansion and Beraducci trees). Relationships between *non-idempotent* intersection type systems (or relational models) and λ-trees has also been investigated [MR14, PPR17]. We believe that non-idempotent intersection types would be easier to compare with our parametric encoding, but we leave a type theoretic study on the different wires as future work.

We would like to investigate the possible relationship between our work and game semantics. In particular, we are interested in the 'HO/N style' as it is known to be related to process representations (e.g., [HO95, HY99, CY19, JS22]). HO/N game semantics for the three trees considered in this paper have been proposed [KNO02, KNO03, OG04]. The technical differences with our work are substantial. For instance, the game semantics are not given in a parametric manner; and the D_∞ equality is obtained via Nakajima trees rather than $\mathrm{BT}_{\eta\infty}$. Nakajima trees are a different 'infinite η-expansion' of Böhm trees, in the sense that $\lambda\widetilde{x}.\, y\,\widetilde{M}$ is expanded to $\lambda\widetilde{x}z_0z_1\ldots.\, y\,\widetilde{M}\ z_0z_1\cdots$; that is, trees may be infinitely branching. In processes, this would mean, for instance, having input prefixes that receive infinitely-many names at the same time. We would like to understand whether the three kinds of wires we considered are meaningful in game semantics. The game semantic counterpart of process wires are the *copycat strategies*, and they intuitively correspond to I-O wires, in that

they begin with an O-move (i.e., an input action). This does not change even in concurrent game semantics [MM07, CCRW17]. We are not aware of game models that use strategies corresponding to the O-I wires or the P wires studied in our paper.

Similarly, we would like to investigate relationships with call-by-name translations of the λ-calculus into (pure) proof-nets [Dan90]. We think that our encoding could be factorised into the translation from λ-calculus into proof-nets and a variant of Abramsky translation [Abr94, BS94]. In this way, a P wire for location names would correspond to an infinitely η-expanded form of the axiom link for the type o according to its recursive equation $o \cong (!o)^{\perp} \parr o$. Infinite η-expansions of the identity axioms have also been considered in Girard's ludics [Gir01], where they are called faxes. Faxes are different from P wires because faxes satisfy an alternation condition on polarity; polarity, as observed by Honda and Laurent [HL10], corresponds to locality in π-calculus.

There are some general conditions for λ-models to be fully abstract for $\mathrm{BT}_{\eta\infty}$s [GFH99, Man09, Bre16]. These cover many λ-models arising from the study of intersection types, game semantics, and (semantics) of linear logic. As described, all these areas are known to be closely related to π-calculus (and to each other). Our full abstraction result for $\mathrm{BT}_{\eta\infty}$s may be explained through these conditions as well. We would also like to pursue a more conceptual understanding of our full abstraction results for LTs and BTs, potentially by investigating the aforementioned connections.

Processes like wires (often called *links*) appear in session-typed process calculi focusing on the Curry-Howard isomorphism, as primitive process constructs used to represent the identity axiom [Wad14]. Some of our assumptions for wires, cf. the substitution-like behaviour when an end-point of the wire is restricted, are then given as explicit rules of the operational semantics of such links.

We have studied the properties of the concrete wires used in the paper on processes encoding functions. We would like to establish more general properties, on arbitrary processes, possibly subject to constraints on the usage of the names of the wires. We would also like to see if other kinds of wires are possible, and which properties they yield.

Acknowledgment

We thank the referees, both those of LICS 2023, and those of LMCS, for insightful comments and valuable suggestions. This work has been supported by the MIUR-PRIN projects 'Analysis of Program Analyses' (ASPRA, ID 201784YSZ5_004), 'Resource Awareness in Programming: Algebra, Rewriting, and Analysis' (RAP, ID P2022HXNSC), by JSPS KAKENHI Grant Number JP24K20731, and by the European Research Council (ERC) Grant DLV-818616 DIAPASoN.

References

[Abr94] Samson Abramsky. Proofs as processes. *Theor. Comput. Sci.*, 135(1):5–9, 1994. doi:10.1016/0304-3975(94)00103-0.

[Bar84] Henk Barendregt. *The Lambda Calculus: Its Syntax and Semantics*. North-Holland Linguistic Series. North-Holland, 1984.

[BHY01] Martin Berger, Kohei Honda, and Nobuko Yoshida. Sequentiality and the pi-calculus. In Samson Abramsky, editor, *Typed Lambda Calculi and Applications TLCA 2001*, volume 2044 of *Lecture Notes in Computer Science*, pages 29–45. Springer, 2001. doi:10.1007/3-540-45413-6_7.

[Bor98] Michele Boreale. On the expressiveness of internal mobility in name-passing calculi. *Theor. Comput. Sci.*, 195(2):205–226, 1998. doi:10.1016/S0304-3975(97)00220-X.

[Bre16] Flavien Breuvart. On the characterization of models of $\mathcal{H}^*$: The semantical aspect. *Log. Methods Comput. Sci.*, 12(2), 2016. doi:10.2168/LMCS-12(2:4)2016.

[BS94] Gianluigi Bellin and Philip J. Scott. On the pi-calculus and linear logic. *Theor. Comput. Sci.*, 135(1):11–65, 1994. doi:10.1016/0304-3975(94)00104-9.

[CCRW17] Simon Castellan, Pierre Clairambault, Silvain Rideau, and Glynn Winskel. Games and strategies as event structures. *Log. Methods Comput. Sci.*, 13(3), 2017. doi:10.23638/LMCS-13(3:35)2017.

[CPT16] Luís Caires, Frank Pfenning, and Bernardo Toninho. Linear logic propositions as session types. *Mathematical Structures in Computer Science*, 26(3):367–423, 2016. doi:10.1017/S0960129514000218.

[CY19] Simon Castellan and Nobuko Yoshida. Two sides of the same coin: session types and game semantics: a synchronous side and an asynchronous side. *Proc. ACM Program. Lang.*, 3(POPL):27:1–27:29, 2019. doi:10.1145/3290340.

[Dan90] Vincent Danos. *La Logique Linéaire appliquée à l'étude de divers processus de normalisation (principalement du Lambda-calcul)*. PhD thesis, Université Paris 7, France, 1990.

[DHS19] Adrien Durier, Daniel Hirschkoff, and Davide Sangiorgi. Divergence and unique solution of equations. *Log. Methods Comput. Sci.*, 15(3), 2019. doi:10.23638/LMCS-15(3:12)2019.

[DHS22] Adrien Durier, Daniel Hirschkoff, and Davide Sangiorgi. Eager functions as processes. *Theor. Comput. Sci.*, 913:8–42, 2022. doi:10.1016/j.tcs.2022.01.043.

[Eng81] E. Engeler. Algebras and combinators. *Algebra Universalis*, 13:389–392, 1981.

[Fu97] Y. Fu. A proof theoretical approach to communication. In *24th ICALP*, volume 1256 of *Lecture Notes in Computer Science*. Springer Verlag, 1997. doi:10.1007/3-540-63165-8_189.

[GFH99] Pietro Di Gianantonio, Gianluca Franco, and Furio Honsell. Game semantics for untyped $\lambda\beta\eta$-calculus. In Jean-Yves Girard, editor, *Typed Lambda Calculi and Applications, 4th International Conference, TLCA'99, L'Aquila, Italy, April 7-9, 1999, Proceedings*, volume 1581 of *Lecture Notes in Computer Science*, pages 114–128. Springer, 1999. doi:10.1007/3-540-48959-2_10.

[Gir01] Jean-Yves Girard. Locus solum: From the rules of logic to the logic of rules. *Math. Struct. Comput. Sci.*, 11(3):301–506, 2001. doi:10.1017/S096012950100336X.

[HL10] Kohei Honda and Olivier Laurent. An exact correspondence between a typed pi-calculus and polarised proof-nets. *Theor. Comput. Sci.*, 411(22-24):2223–2238, 2010. doi:10.1016/J.TCS.2010.01.028.

[HO95] J. M. E. Hyland and C.-H. Luke Ong. Pi-calculus, dialogue games and PCF. In John Williams, editor, *Proceedings of conf. on Functional programming languages and computer architecture (FPCA)*, pages 96–107. ACM, 1995. doi:10.1145/224164.224189.

[HY95] K. Honda and N. Yoshida. On reduction-based process semantics. *Theor. Comput. Sci.*, 152(2):437–486, 1995. doi:10.1016/0304-3975(95)00074-7.

[HY99] Kohei Honda and Nobuko Yoshida. Game-theoretic analysis of call-by-value computation. *Theor. Comput. Sci.*, 221(1-2):393–456, 1999. doi:10.1016/S0304-3975(99)00039-0.

[JS22] Guilhem Jaber and Davide Sangiorgi. Games, mobile processes, and functions. In Florin Manea and Alex Simpson, editors, *30th EACSL Annual Conference on Computer Science Logic, CSL 2022*, volume 216 of *LIPIcs*, pages 25:1–25:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.CSL.2022.25.

[KNO02] Andrew D. Ker, Hanno Nickau, and C.-H. Luke Ong. Innocent game models of untyped lambda-calculus. *Theor. Comput. Sci.*, 272(1-2):247–292, 2002. doi:10.1016/S0304-3975(00)00353-4.

[KNO03] Andrew D. Ker, Hanno Nickau, and C.-H. Luke Ong. Adapting innocent game models for the Böhm tree lambda-theory. *Theor. Comput. Sci.*, 308(1-3):333–366, 2003. doi:10.1016/S0304-3975(02)00849-6.

[Las99] Søren B. Lassen. Bisimulation in untyped lambda calculus: Böhm trees and bisimulation up to context. In Stephen D. Brookes, Achim Jung, Michael W. Mislove, and Andre Scedrov, editors, *Mathematical Foundations of Progamming Semantics MFPS 1999*, volume 20 of *Electronic Notes in Theoretical Computer Science*, pages 346–374. Elsevier, 1999. doi:10.1016/S1571-0661(04)80083-5.

[Lév76] Jean-Jacques Lévy. An algebraic interpretation of the $\lambda\beta K$-calculus; and an application of a labelled λ-calculus. *Theor. Comput. Sci.*, 2(1):97–114, 1976. doi:10.1016/0304-3975(76)90009-8.

[Lon83] G. Longo. Set theoretical models of lambda calculus: Theory, expansions and isomorphisms. *Annales of Pure and Applied Logic*, 24:153–188, 1983.

[Man09] Giulio Manzonetto. A general class of models of $\mathcal{H}^*$. In Rastislav Královic and Damian Niwinski, editors, *Mathematical Foundations of Computer Science 2009, 34th International Symposium, MFCS 2009, Novy Smokovec, High Tatras, Slovakia, August 24-28, 2009. Proceedings*, volume 5734 of *Lecture Notes in Computer Science*, pages 574–586. Springer, 2009. doi:10.1007/978-3-642-03816-7_49.

[Mer01] M. Merro. Locality in the π-calculus and applications to object-oriented languages. PhD thesis, Ecoles des Mines de Paris, 2001.

[Mil90] Robin Milner. Functions as processes. Research Report RR-1154, INRIA, 1990. URL: https://hal.inria.fr/inria-00075405.

[Mil92] Robin Milner. Functions as processes. *Math. Struct. Comput. Sci.*, 2(2):119–141, 1992. doi:10.1017/S0960129500001407.

[Mil93] Robin Milner. The polyadic π-calculus: a tutorial. In FriedrichL. Bauer, Wilfried Brauer, and Helmut Schwichtenberg, editors, *Logic and Algebra of Specification*, volume 94 of *NATO ASI Series*, pages 203–246. Springer Berlin Heidelberg, 1993. doi:10.1007/978-3-642-58041-3_6.

[MM07] Paul-André Melliès and Samuel Mimram. Asynchronous games: Innocence without alternation. In Luís Caires and Vasco Thudichum Vasconcelos, editors, *CONCUR 2007 - Concurrency Theory, 18th International Conference*, volume 4703 of *Lecture Notes in Computer Science*, pages 395–411. Springer, 2007. doi:10.1007/978-3-540-74407-8_27.

[MR14] Giulio Manzonetto and Domenico Ruoppolo. Relational graph models, taylor expansion and extensionality. In Bart Jacobs, Alexandra Silva, and Sam Staton, editors, *Proceedings of the 30th Conference on the Mathematical Foundations of Programming Semantics, MFPS 2014, Ithaca, NY, USA, June 12-15, 2014*, volume 308 of *Electronic Notes in Theoretical Computer Science*, pages 245–272. Elsevier, 2014. doi:10.1016/J.ENTCS.2014.10.014.

[MS04] Massimo Merro and Davide Sangiorgi. On asynchrony in name-passing calculi. *Mathematical Structures in Computer Science*, 14(5):715–767, 2004. doi:10.1017/S0960129504004323.

[OG04] C.-H. Luke Ong and Pietro Di Gianantonio. Games characterizing Levy-Longo trees. *Theor. Comput. Sci.*, 312(1):121–142, 2004. doi:10.1016/S0304-3975(03)00405-5.

[Plo72] G.D. Plotkin. A set theoretical definition of application. Technical Report Tech. Rep. MIP-R-95, School of A.I., Univ. of Edinburgh, 1972.

[PPR17] Luca Paolini, Mauro Piccolo, and Simona Ronchi Della Rocca. Essential and relational models. *Math. Struct. Comput. Sci.*, 27(5):626–650, 2017. doi:10.1017/S0960129515000316.

[PV98] J. Parrow and B. Victor. The fusion calculus: Expressiveness and symmetry in mobile processes. In *13th LICS Conf.* IEEE Computer Society Press, 1998.

[San93] Davide Sangiorgi. An investigation into functions as processes. In Stephen D. Brookes, Michael G. Main, Austin Melton, Michael W. Mislove, and David A. Schmidt, editors, *Mathematical Foundations of Programming Semantics (MFPS)*, volume 802 of *Lecture Notes in Computer Science*, pages 143–159. Springer, 1993. doi:10.1007/3-540-58027-1_7.

[San96a] Davide Sangiorgi. Locality and interleaving semantics in calculi for mobile processes. *Theor. Comput. Sci.*, 155(1):39–83, 1996. doi:10.1016/0304-3975(95)00020-8.

[San96b] Davide Sangiorgi. π-Calculus, internal mobility, and agent-passing calculi. *Theor. Comput. Sci.*, 167(1&2):235–274, 1996. doi:10.1016/0304-3975(96)00075-8.

[San99] Davide Sangiorgi. From λ to π; or, Rediscovering continuations. *Mathematical Structures in Computer Science*, 9(4):367–401, 1999. doi:10.1017/S0960129599002881.

[San00] Davide Sangiorgi. Lazy functions and mobile processes. In Gordon D. Plotkin, Colin Stirling, and Mads Tofte, editors, *Proof, Language, and Interaction, Essays in Honour of Robin Milner*, pages 691–720. The MIT Press, 2000.

[Sco76] Dana S. Scott. Data types as lattices. *SIAM J. Comput.*, 5(3):522–587, 1976. doi:10.1137/0205037.

[SW01] Davide Sangiorgi and David Walker. *The π-calculus—A Theory of Mobile Processes.* Cambridge University Press, 2001.

[SX18] Davide Sangiorgi and Xian Xu. Trees from functions as processes. *Log. Methods Comput. Sci.*, 14(3), 2018. doi:10.23638/LMCS-14(3:11)2018.

[Thi97] Hayo Thielecke. *Categorical structure of continuation passing style*. PhD thesis, University of Edinburgh, UK, 1997.

[vBBDdV02] Steffen van Bakel, Franco Barbanera, Mariangiola Dezani-Ciancaglini, and Fer-Jan de Vries. Intersection types for lambda-trees. *Theor. Comput. Sci.*, 272(1-2):3–40, 2002. doi:10.1016/S0304-3975(00)00346-7.

[Wad76] Christopher P. Wadsworth. The relation between computational and denotational properties for Scott's D_∞-models of the lambda-calculus. *SIAM J. Comput.*, 5(3):488–521, 1976. doi:10.1137/0205036.

[Wad14] Philip Wadler. Propositions as sessions. *J. Funct. Program.*, 24(2-3):384–418, 2014. doi:10.1017/S095679681400001X.

[Yos02] Nobuko Yoshida. Minimality and separation results on asynchronous mobile processes - representability theorems by concurrent combinators. *Theor. Comput. Sci.*, 274(1-2):231–276, 2002. doi:10.1016/S0304-3975(00)00310-8.

Appendix A. List of notations

The following tables summarise the notations (for encodings, equivalence etc.) used in this paper. Some of the notations are only used in Appendix.

Encodings

$\mathcal{A}$	Abstract encoding	Figure 2
$\mathcal{A}_{\mathtt{IO}}$	$\mathcal{A}$ instantiated with I-O wires	Section 7
$\mathcal{A}_{\mathtt{OI}}$	$\mathcal{A}$ instantiated with O-I wires	Section 7
$\mathcal{A}_{\mathtt{P}}$	$\mathcal{A}$ instantiated with P wires	Section 8
$\mathcal{O}$	(Abstract) Optimised encoding	Figure 3
$\mathcal{O}^n$	'Optimised encoding of arguments'	Figure 3
$\mathcal{O}_{\mathtt{IO}}$	$\mathcal{O}$ instantiated with I-O wires	Section 7
$\mathcal{O}_{\mathtt{OI}}$	$\mathcal{O}$ instantiated with O-I wires	Section 7
$\mathcal{O}_{\mathtt{P}}$	$\mathcal{O}$ instantiated with P wires	Section 7
$\mathcal{M}$	Milner's encoding	Section 4.1
$\mathcal{E}$	Metavariable for encodings	—

Wires

$a \leftrightarrow \bar{b}$	(Abstract) Wire	Definition 4.1
$a \underset{\mathtt{IO}}{\leftrightarrow} \bar{b}$	I-O wire	Section 6
$b \underset{\mathtt{OI}}{\leftrightarrow} \bar{a}$	O-I wire	Section 6
$a \underset{\mathtt{P}}{\leftrightarrow} \bar{b}$	P wire	Section 6

Equivalence and Preorders

$\approx$	Weak bisimilarity for Iπ	Definition 2.4
$\sim$	Strong bisimilarity for Iπ	Definition 2.4
$\lesssim$	Expansion relation for Iπ	Definition 2.5
$\equiv$	Structural congruence for Iπ	Definition 2.6
$\equiv_\alpha$	α-equivalence	—

λ-trees

LT	Lévy-Longo tree	Section 2.1
BT	Böhm tree	Section 2.1
$\mathrm{BT}_{\eta\infty}$	Böhm tree up-to infinite η-expansion	Section 2.1

Reduction of λ-calculus

$\rightarrow$	β-reduction	Section 2.1
$\rightarrow_{\mathtt{sn}}$	strong call-by-name reduction	Section 2.1
$\rightarrow_{\mathtt{h}}$	head reduction	Section 2.1

Appendix B. Proofs for Section 5

We present the proofs of properties of the abstract optimised encoding $\mathcal{O}$.

B.1. **Proofs for the basic properties of $\mathcal{O}$.** This section first proves the properties of $\mathcal{O}$ that are analogous to those for the unoptimised encoding $\mathcal{A}$. Then, using these properties, we prove that $\mathcal{O}$ is indeed an optimisation of $\mathcal{A}$.

Lemma B.1 and B.2 say that $\mathcal{O}[\![M]\!]_p$ and $\mathcal{O}^n\langle p_0, p, \mathcal{O}[\![M_1]\!] \cdots \mathcal{O}[\![M_n]\!]\rangle$ are respectful.

Lemma B.1.
(1) $\boldsymbol{\nu} q\,(p \leftrightarrow \bar{q} \mid \mathcal{O}[\![M]\!]_q) \gtrsim \mathcal{O}[\![M]\!]_p$.
(2) $\boldsymbol{\nu} x\,(x \leftrightarrow \bar{y} \mid \mathcal{O}[\![M]\!]_p) \gtrsim \mathcal{O}[\![M\{y/x\}]\!]_p$.

Proof. Similar to Lemma 4.2 □

Lemma B.2. *For $n \geq 1$, we have*

$$\boldsymbol{\nu} p_0\ (p_0 \leftrightarrow \bar{q} \mid \mathcal{O}^n\langle p_0, p, \mathcal{O}[\![M_1]\!], \ldots, \mathcal{O}[\![M_n]\!]\rangle) \gtrsim \mathcal{O}^n\langle q, p, \mathcal{O}[\![M_1]\!], \ldots, \mathcal{O}[\![M_n]\!]\rangle.$$

Proof. By induction on n. First observe that

$$\boldsymbol{\nu} y\,(x \leftrightarrow \bar{y} \mid !y(p).\,\mathcal{O}[\![M]\!]_p) \gtrsim !x(p).\,\mathcal{O}[\![M]\!]_p \tag{B.1}$$

under the assumption that $y \notin \mathsf{fv}(M)$. This is derived from 6 of Definition 4.1 together with Lemma B.1.

The base case is the case where $n = 1$. In this case, we need to show

$$\begin{aligned}&\boldsymbol{\nu} p_0\ (p_0 \leftrightarrow \bar{q} \mid \bar{p}_0(x_1, p_1)\colon (!x_1(r_1).\,\mathcal{O}[\![M_1]\!]_{r_1} \mid p \leftrightarrow \bar{p}_1))\\ &\gtrsim \bar{q}(x_1, p_1)\colon (!x_1(r_1).\,\mathcal{O}[\![M_1]\!]_{r_1} \mid p \leftrightarrow \bar{p}_1)\,.\end{aligned}$$

Using (B.1) and the transitivity of wires, we derive the expansion by applying 5 of Definition 4.1.

Now we consider the case $n \geq 1$. We apply 5 of Definition 4.1. The premise of this law is satisfied because of (B.1) and the induction hypothesis. □

We now prove that the optimised encoding validates β-reduction.

Lemma B.3. *Let $m, n \geq 0$. Then*

$$\begin{aligned}&\boldsymbol{\nu} q\,(\mathcal{O}^m\langle p, q, \mathcal{O}[\![M_1]\!], \ldots, \mathcal{O}[\![M_m]\!]\rangle \mid \mathcal{O}^n\langle q, r, \mathcal{O}[\![N_1]\!], \ldots, \mathcal{O}[\![N_n]\!]\rangle)\\ &\gtrsim \mathcal{O}^{m+n}\langle p, r, \mathcal{O}[\![M_1]\!], \ldots, \mathcal{O}[\![M_m]\!], \mathcal{O}[\![N_1]\!], \ldots, \mathcal{O}[\![N_n]\!]\rangle.\end{aligned}$$

Proof. Follows from Lemma B.2. □

Lemma B.4. *Suppose that $x \notin \mathsf{fv}(N)$. Then $\boldsymbol{\nu} x\ (\mathcal{O}[\![M]\!]_p \mid !x(q).\,\mathcal{O}[\![N]\!]_q) \gtrsim \mathcal{O}[\![M\{N/x\}]\!]_p$*

Proof. By induction on the structure of M. The proof is similar to that of the unoptimised case (Lemma 4.3). Indeed the proof for the base case, namely the case $M = x$, is exactly the same as that of Lemma 4.3 since $\mathcal{O}[\![x]\!]_p = \mathcal{A}[\![x]\!]_p$. The inductive case follows from the induction hypothesis and the replication theorems. □

Lemma B.5. *If $n \geq 1$, then*

$$\boldsymbol{\nu} q\,(\mathcal{O}[\![M_0]\!]_q \mid \mathcal{O}^n\langle q, p, \mathcal{O}[\![M_1]\!], \ldots, \mathcal{O}[\![M_n]\!]\rangle) \gtrsim \mathcal{O}[\![M_0\, M_1 \cdots M_n]\!]_p.$$

Proof. **Case** $M_0 = x$**:** This case follows from Lemma B.2. More precisely, we have

$$\begin{aligned}
&\boldsymbol{\nu} q\,(\mathcal{O}[\![x]\!]_q \mid \mathcal{O}^n\langle q, p, \mathcal{O}[\![M_1]\!], \ldots, \mathcal{O}[\![M_n]\!]\rangle)\\
&= \boldsymbol{\nu} q\,(\bar{x}(q') \colon q \leftrightarrow \bar{q}' \mid \mathcal{O}^n\langle q, p, \mathcal{O}[\![M_1]\!], \ldots, \mathcal{O}[\![M_n]\!]\rangle)\\
&\equiv \bar{x}(q') \colon \boldsymbol{\nu} q\,(q \leftrightarrow \bar{q}' \mid \mathcal{O}^n\langle q, p, \mathcal{O}[\![M_1]\!], \ldots, \mathcal{O}[\![M_n]\!]\rangle)\\
&\gtrsim \bar{x}(q') \colon \mathcal{O}^n\langle q', p, \mathcal{O}[\![M_1]\!], \ldots, \mathcal{O}[\![M_n]\!]\rangle. && \text{(Lemma B.2)}\\
&= \mathcal{O}[\![x\, M_1 \cdots M_n]\!]_p.
\end{aligned}$$

Case $M_0 = \lambda x.\, M$**:** By definition,

$$\begin{aligned}
&\boldsymbol{\nu} q\,(\mathcal{O}[\![\lambda x.\, M]\!]_q \mid \mathcal{O}^n\langle q, p, \mathcal{O}[\![M_1]\!], \ldots, \mathcal{O}[\![M_n]\!]\rangle)\\
&\equiv \mathcal{O}[\![(\lambda x.\, M)\, M_1 \cdots M_n]\!]_p
\end{aligned}$$

Case $M_0 = x\, N_1 \cdots N_m$ **with** $m \geq 1$**:** In this case, we have

$$\begin{aligned}
&\boldsymbol{\nu} q\,(\mathcal{O}[\![M_0]\!]_q \mid \mathcal{O}^n\langle q, p, \mathcal{O}[\![M_1]\!], \ldots, \mathcal{O}[\![M_n]\!]\rangle)\\
&= \boldsymbol{\nu} q\,(\bar{x}(q_0) \colon \mathcal{O}^m\langle q_0, q, \mathcal{O}[\![N_1]\!], \ldots, \mathcal{O}[\![N_m]\!]\rangle \mid \mathcal{O}^n\langle q, p, [\![M_1]\!], \ldots, [\![M_n]\!]\rangle)\\
&\equiv \bar{x}(q_0) \colon \boldsymbol{\nu} q\,(\mathcal{O}^m\langle q_0, q, \mathcal{O}[\![N_1]\!], \ldots, \mathcal{O}[\![N_m]\!]\rangle \mid \mathcal{O}^n\langle q, p, \mathcal{O}[\![M_1]\!], \ldots, \mathcal{O}[\![M_n]\!]\rangle)\\
&\gtrsim \bar{x}(q_0) \colon \mathcal{O}^{m+n}\langle q_0, p, \mathcal{O}[\![N_1]\!], \ldots, \mathcal{O}[\![N_m]\!], [\![M_1]\!], \ldots, [\![M_n]\!]\rangle && \text{(Lemma B.3)}\\
&= \mathcal{O}[\![x\, N_1 \cdots N_m\, M_1 \cdots M_n]\!]_p
\end{aligned}$$

Case $M_0 = (\lambda x.\, N_0)\, N_1 \cdots N_m$ **with** $m \geq 1$**:** Similar to the previous case. □

Lemma 5.1. *If* $M \to N$*, then* $\mathcal{O}[\![M]\!]_p \gtrsim \mathcal{O}[\![N]\!]_p$*.*

Proof. It suffices to consider the case where $M = (\lambda x.\, M_0)\, M_1 \ldots M_n$, where $n \geq 1$, because the other cases follow from the precongruence of $\lesssim$. We only consider the case where $n \geq 2$ because the case $n = 1$ can be proved as in the case for the unoptimised encoding. By definition, $\mathcal{O}[\![M]\!]_p$ is

$$\boldsymbol{\nu} p_0\ (p_0(x, q) \colon \mathcal{O}[\![M_0]\!]_q \mid \mathcal{O}^n\langle p_0, p, \mathcal{O}[\![M_1]\!] \cdots \mathcal{O}[\![M_n]\!]\rangle)$$

By interaction on p_0 (Lemma 3.1), we have

$$\mathcal{O}[\![M]\!]_p \gtrsim (\boldsymbol{\nu} x, q)\,(\mathcal{O}[\![M_0]\!]_q \mid !x(r_1).\, \mathcal{O}[\![M_1]\!]_{r_1} \mid \mathcal{O}^{n-1}\langle q, p, \mathcal{O}[\![M_2]\!] \cdots \mathcal{O}[\![M_n]\!]\rangle).$$

Note that the assumption of Lemma 3.1 is satisfied by Lemma B.1. The claim follows because

$$\begin{aligned}
&(\boldsymbol{\nu} x, q)\,(\mathcal{O}[\![M_0]\!]_q \mid !x(r_1).\, \mathcal{O}[\![M_1]\!]_{r_1} \mid \mathcal{O}^{n-1}\langle q, p, \mathcal{O}[\![M_2]\!] \cdots \mathcal{O}[\![M_n]\!]\rangle)\\
&\gtrsim \boldsymbol{\nu} q\,(\mathcal{O}[\![M_0\{^{M_1}\!/_x\}]\!]_q \mid \mathcal{O}^{n-1}\langle q, p, \mathcal{O}[\![M_2]\!] \cdots \mathcal{O}[\![M_n]\!]\rangle) && \text{(Lemma B.4)}\\
&\gtrsim \mathcal{O}[\![M_0\{^{M_1}\!/_x\}\, M_2 \cdots M_n]\!]_p && \text{(Lemma B.5)}
\end{aligned}$$

□

Finally, we prove that $\mathcal{O}$ is indeed an optimisation.

Lemma 5.2. $\mathcal{A}[\![M]\!]_p \gtrsim \mathcal{O}[\![M]\!]_p$*.*

Proof. By induction on the structure of M. The cases of variables and abstraction are straightforward. Consider now $M = y\, N_1 \cdots N_n$. We use induction on n. For the base case,

i.e. $M = y\,N_1$ we have:

$$\begin{aligned}
&\mathcal{A}[\![y\,N_1]\!]_p \\
&= \boldsymbol{\nu} q\ \big(\mathcal{A}[\![y]\!]_q \mid \bar{q}(x,p')\colon \big(!x(r).\,\mathcal{A}[\![N_1]\!]_r \mid p \leftrightarrow \bar{p}'\big)\big) \\
&= \boldsymbol{\nu} q\ \big(\bar{y}(q')\colon q \leftrightarrow \bar{q}' \mid \bar{q}(x,p')\colon \big(!x(r).\,\mathcal{A}[\![N_1]\!]_r \mid p \leftrightarrow \bar{p}'\big)\big) \\
&\equiv \bar{y}(q')\colon \boldsymbol{\nu} q\ \big(q \leftrightarrow \bar{q}' \mid \bar{q}(x,p')\colon \big(!x(r).\,\mathcal{A}[\![N_1]\!]_r \mid p \leftrightarrow \bar{p}'\big)\big) \\
&\gtrsim \bar{y}(q')\colon \boldsymbol{\nu} q\ \big(q \leftrightarrow \bar{q}' \mid \bar{q}(x,p')\colon \big(!x(r).\,\mathcal{O}[\![N_1]\!]_r \mid p \leftrightarrow \bar{p}'\big)\big) && \text{(i.h.)} \\
&\gtrsim \bar{y}(q')\colon \bar{q}'(x,p')\colon \big(!x(r).\,\mathcal{O}[\![N_1]\!]_r \mid p \leftrightarrow \bar{p}'\big) && \text{(Lemma B.2)} \\
&\equiv_\alpha \bar{y}(p_0)\colon \bar{p}_0(x,p_1)\colon (!x(r).\,\mathcal{O}[\![N_1]\!]_r \mid p \leftrightarrow \bar{p}_1) \\
&= \mathcal{O}[\![y\,N_1]\!]_p
\end{aligned}$$

The inductive case for n can be proved similarly using the induction hypothesis and Lemma B.3.

The case of $M = (\lambda x.\,N)\,N_1 \cdots N_n$ is also handled in the same manner. □

B.2. **Properties about transitions.** Now we prove the properties about the transitions $\mathcal{O}[\![M]\!]_p$ can do. The first thing we prove is the operational correspondence for τ-transitions.

Lemma 5.3. *If $\mathcal{O}[\![M]\!]_p \xrightarrow{\tau} P$ then there exists N such that $M \to_{\mathtt{sn}} N$ and $P \gtrsim \mathcal{O}[\![N]\!]_p$.*

Proof. By induction on the structure of M. The case where $M = x\,\widetilde{M}$, where $\widetilde{M}$ is a possibly empty sequence of terms, is trivial since $\mathcal{O}[\![M]\!]_p$ cannot make any τ-action. The case for $M = \lambda x.\,M_0$ is also straightforward: it follows from the induction hypothesis.

We now consider the remaining case where $M = (\lambda x.\,M_0)\,M_1 \cdots M_n$ and $n \geq 1$. Recall that $\mathcal{O}[\![(\lambda x.\,M_0)\,M_1 \cdots M_n]\!]_p$ is

$$\begin{aligned}
&\boldsymbol{\nu} p_0\ (p_0(x,q)\colon \mathcal{O}[\![M_0]\!]_q \mid \bar{p}_0(x_1,p_1)\colon \cdots \bar{p}_{n-1}(x_n,p_n)\colon \\
&\qquad (!x_1(r_1).\,\mathcal{O}[\![M_1]\!]_{r_1} \mid \cdots \mid !x_n(r_1).\,\mathcal{O}[\![M_n]\!]_{r_n} \mid p \leftrightarrow \bar{p}_n))
\end{aligned}$$

There are two cases to consider: (1) the case where the τ-action originates from the τ-action on $\mathcal{O}[\![M_0]\!]_q$ and (2) the case where the τ-action is caused by the interaction at p_0. The former case can be easily proved by using the induction hypothesis. The latter case is the most important case. Since $\mathcal{O}[\![M_0]\!]_q$ is I-respectful with respect to $q' \leftrightarrow \bar{q}$ and O-respectful with respect to $x \leftrightarrow \bar{x}'$ (Lemma B.1), we can use the communication law for the permeable prefixes on p_0. Therefore, we have

$$\begin{aligned}
&\boldsymbol{\nu} p_0\ (p_0(x,q)\colon \mathcal{O}[\![M_0]\!]_q \mid \bar{p}_0(x_1,p_1)\colon \cdots \bar{p}_{n-1}(x_n,p_n)\colon \\
&\qquad (!x_1(r_1).\,\mathcal{O}[\![M_1]\!]_{r_1} \mid \cdots \mid !x_n(r_1).\,\mathcal{O}[\![M_n]\!]_{r_n} \mid p \leftrightarrow \bar{p}_n)) \\
&\gtrsim (\boldsymbol{\nu} x_1, p_1)\ (\mathcal{O}[\![M_0\{^{x_1}\!/_x\}]\!]_q \mid !x_1(r_1).\,\mathcal{O}[\![M_1]\!]_{p_1} \mid \bar{p}_1(x_2,p_2)\colon \cdots \bar{p}_{n-1}(x_n,p_n)\colon && \text{(Lemma 3.1)} \\
&\qquad\qquad (!x_2(r_2).\,\mathcal{O}[\![M_2]\!]_{p_2} \mid \cdots \mid !x_n(r_1).\,\mathcal{O}[\![M_n]\!]_{r_n} \mid p \leftrightarrow \bar{p}_n)) \\
&\gtrsim \boldsymbol{\nu} p_1\ (\mathcal{O}[\![M_0\{^{M_1}\!/_x\}]\!]_{p_1} \mid \bar{p}_1(x_2,p_2)\colon \cdots \bar{p}_{n-1}(x_n,p_n)\colon && \text{(Lemma B.4)} \\
&\qquad (!x_2(r_2).\,\mathcal{O}[\![M_2]\!]_{r_2} \mid \cdots \mid !x_n(r_1).\,\mathcal{O}[\![M_n]\!]_{r_n} \mid p \leftrightarrow \bar{p}_n))
\end{aligned}$$

If $n \geq 2$, we can apply Lemma B.5 and obtain $P \gtrsim \mathcal{O}[\![M_0\{^{M_1}\!/_x\}\, M_2 \cdots M_n]\!]_p$. If $n = 1$, the subprocess of the form $\bar{p}_1(x_2, p_2) \colon \cdots$ is simply $p \leftrightarrow \bar{p}_1$. Hence, by Lemma B.1, we have

$$\begin{aligned}\mathcal{O}[\![(\lambda x.\, M_0)\, M_1]\!]_p &\gtrsim \boldsymbol{\nu} p_1\ \left(\mathcal{O}[\![M_0\{^{M_1}\!/_x\}]\!]_{p_1} \mid p \leftrightarrow \bar{p}_1\right)\\ &\gtrsim \mathcal{O}[\![M_0\{^{M_1}\!/_x\}]\!]_{p_1}\end{aligned}$$

as desired. □

The next thing we prove is the property about input actions.

Lemma 5.4. *If $\mathcal{O}[\![M]\!]_p \xrightarrow{\mu} P$ and μ is an input action, then μ is an input at p.*

Proof. By induction on M with a case analysis on the shape of M.

Case $M = x$**:** In this case, $\mathcal{O}[\![M]\!]_p = \bar{x}(p') \colon p \leftrightarrow \bar{p}'$. Since the only free name that appears in an input occurrence is p (because of 1 of Definition 4.1), the only possible input action $\mathcal{O}[\![M]\!]_p$ can do is an input on p. (Note that whether the process can do an input on p will depend on the concrete instantiation of $p \leftrightarrow \bar{q}$.)

Case $M = \lambda x.\, M_0$**:** Since $\mathcal{O}[\![M]\!]_p = p(x, q) \colon \mathcal{O}[\![M_0]\!]_q$, if $\mathcal{O}[\![M]\!]_p \xrightarrow{\mu} P$ and μ is an input action, then this action must either be an input on p or an input that originates from $\mathcal{O}[\![M_0]\!]_q$. In the latter case, μ must be an input on q by the induction hypothesis. Since q is bound by $p(x, q) \colon$, this action cannot induce an input action of $\mathcal{O}[\![M]\!]_p$.

Case $M = x\, M_1 \cdots M_n$**:** In this case, $\mathcal{O}[\![M]\!]_p$ is

$$\begin{aligned}&\bar{x}(p_0) \colon \bar{p}_0(x_1, p_1) \colon \cdots \bar{p}_{n-1}(x_n, p_n) \colon\\ &(!x_1(r_1).\, \mathcal{O}[\![M_1]\!]_{r_1} \mid \cdots \mid !x_n(r_n).\, \mathcal{O}[\![M_n]\!]_{r_n} \mid p \leftrightarrow \bar{p}_n)\end{aligned}$$

Since the only free name that may appear in an input occurrence which is not guarded by a (non-permeable) prefixing is p, the only possible input action $\mathcal{O}[\![M]\!]_p$ can do is an input on p.

Case $M = (\lambda x.\, M_0)\, M_1 \cdots M_n$**:** By combining the argument we made in the previous two cases. □

We now consider the relationship between output actions and head normal forms. As an auxiliary definition, we introduce a special form of a context.

Definition B.6. *H-contexts* are contexts defined by the following grammar:

$$H ::= [\cdot] \mid (\lambda x.\, H)\, M_1 \cdots M_n \quad (n \geq 0)$$

Lemma B.7. *Let $M \stackrel{def}{=} H[x\, \widetilde{M}]$, where $\widetilde{M}$ is a possibly empty sequence of terms, and assume that $x \in \mathsf{fv}(M)$. Then $M \Longrightarrow_{\mathtt{h}} \lambda \widetilde{y}.\, x\, \widetilde{N}$ for some possibly empty sequences of variables $\widetilde{y}$ and terms $\widetilde{N}$.*

Lemma B.8. *Let M be a λ-term and suppose that $\mathcal{O}[\![M]\!]_p \xrightarrow{\mu} P$ for an output action μ. Then the action μ must be of the form $\bar{x}(p)$ for a fresh p and a variable $x \in \mathsf{fv}(M)$, and there exists a H-context H such that $H[x\, \widetilde{M}] = M$ for some possibly empty sequence $\widetilde{M} = M_1, \ldots M_n$.*

Proof. By induction on M with a case analysis on the shape of M.

Case $M = x$**:** In this case, $\mathcal{O}[\![M]\!]_p = \bar{x}(p') \colon p \leftrightarrow \bar{p}'$, and the only output action $\mathcal{O}[\![M]\!]_p$ can do is $\overline{x}(p')$ (cf. 1 of Definition 4.1). We can take the empty context $[\cdot]$ for H and the empty sequence for $\widetilde{M}$.

Case $M = \lambda x.\, M_0$**:** Since we have $\mathcal{O}[\![M]\!]_p = p(x, q) \colon \mathcal{O}[\![M_0]\!]_q$, if $\mathcal{O}[\![M]\!]_p \xrightarrow{\overline{y}(r)} P$ then this action must originate from $\mathcal{O}[\![M_0]\!]_q$ and we must have $x \neq y$. Hence, by the induction

hypothesis, there is a H-context H' and a sequence of terms $\widetilde{M}$ such that $M_0 = H'[y\,\widetilde{M}]$ with $y \in \mathsf{fv}(M_0)$. We can take H as $\lambda x.\, H'$, and since $y \neq x$ we also have $y \in \mathsf{fv}(M)$.

Case $M = x\,M_1 \cdots M_n$**:** In this case, $\mathcal{O}[\![M]\!]_p$ is

$$\begin{aligned}&\overline{x}(p_0) : \bar{p}_0(x_1, p_1) : \cdots \bar{p}_{n-1}(x_n, p_n) : \\ &(!x_1(r_1).\,\mathcal{O}[\![M_1]\!]_{r_1} \mid \cdots \mid !x_n(r_n).\,\mathcal{O}[\![M_n]\!]_{r_n} \mid p \leftrightarrow \bar{p}_n)\,.\end{aligned}$$

Obviously, the only output $\mathcal{O}[\![M]\!]_p$ can do is $\overline{x}(p_0)$. Hence, the claim holds by taking $H = [\cdot]$ and $\widetilde{M} = M_1, \ldots, M_n$.

Case $M = (\lambda x.\, M_0)\,M_1 \cdots M_n$**:** Recall that $\mathcal{O}[\![M]\!]_p$ is

$$\begin{aligned}&\boldsymbol{\nu} p_0\,(p_0(x, q) : \mathcal{O}[\![M_0]\!]_q \mid \bar{p}_0(x_1, p_1) : \cdots \bar{p}_{n-1}(x_n, p_n) : \\ &\qquad (!x_1(r_1).\,\mathcal{O}[\![M_1]\!]_{r_1} \mid \cdots \mid !x_n(r_1).\,\mathcal{O}[\![M_n]\!]_{r_n} \mid p \leftrightarrow \bar{p}_n))\end{aligned}$$

If $\mathcal{O}[\![M]\!]_p$ does an output action, then this action must originate from $\mathcal{O}[\![M_0]\!]_q$ and the subject of the action must be different from x. Assume that $\mathcal{O}[\![M_0]\!]_q \xrightarrow{\mu} P'$ for an output action μ whose subject is not x. By the induction hypothesis, $\mu = \overline{y}(r)$ and there is a H-context H' and a sequence of terms $\widetilde{M'}$ such that $M_0 = H'[y\,\widetilde{M'}]$ with $y \in \mathsf{fv}(M_0)$. We can take $H = (\lambda x.\, H')\,M_1 \cdots M_n$. Since $M = H[y\,\widetilde{M'}]$ and $y \in \mathsf{fv}(M)$ the claim follows. □

Lemma 5.5. *Let M be a λ-term. If $\mathcal{O}[\![M]\!]_p \xrightarrow{\overline{x}(q)} P$ for some P, then M has a head normal form $\lambda\widetilde{y}.\, x\,\widetilde{M}$, for some (possibly empty) sequence of terms $\widetilde{M}$ and variables $\widetilde{y}$ with $x \notin \widetilde{y}$.*

Proof. By Lemma B.7 and Lemma B.8. □

Lemma 5.6. *Let M be an unsolvable term. Then there does not exist an output action μ such that $\mathcal{O}[\![M]\!]_p \stackrel{\mu}{\Longrightarrow} P$ for some P.*

Proof. Since M is an unsolvable term, by Lemma 5.5, $\mathcal{O}[\![M]\!]_p$ cannot do an output action. Hence, if $\mathcal{O}[\![M]\!]_p \stackrel{\mu}{\Longrightarrow} P$, where μ is an output action, we must have $\mathcal{O}[\![M]\!]_p (\xrightarrow{\tau})^n P \xrightarrow{\mu} P'$ for $n \geq 1$. Assume that such an n exist. Then, by repeatedly applying Lemma 5.3, we get $P \gtrsim \mathcal{O}[\![M']\!]_p$ for some term M' such that $M \to^n M'$. Note that M' is also an unsolvable term. However, this is a contradiction since $P \xrightarrow{\mu} P'$ for an output action μ, but $\mathcal{O}[\![M']\!]_p \not\xrightarrow{\mu}$ by Lemma 5.5. □

Appendix C. Supplementary Materials for Section 6

In this section, we prove that the three concrete wires we introduced satisfy the properties of Definition 4.1. As explained in the main text, we first show that the wires are transitive, and then the other laws are proved by algebraic reasoning exploiting the transitivity of wires.

C.1. **Proofs for transitivity.** We prove that I-O wires and O-I wires are transitive. The reasoning is similar to that of the P wires which we saw in Section 6; we use bisimulation up-to context and expansion. The proofs for the I-O wires and O-I wires are slightly simpler than that of the P wires since these wires have fewer permeable prefixes, and the two proofs are essentially the same because of their 'duality'.

Chains of I-O wires and O-I wires, denoted by $\mathrm{chain}^n_{\mathtt{IO}}(a, b)$ and $\mathrm{chain}^n_{\mathtt{OI}}(a, b)$, respectively, are defined similar to the chains of P wires. (Only for the $\mathrm{chain}^n_{\mathtt{OI}}(p, q)$, we assume that p is

an output name and q is an input name so that the 'direction of the chain' becomes from output to input.)

Lemma C.1. *The wires $p \leftrightarrows_{\mathtt{IO}} \bar{q}$ and $x \leftrightarrows_{\mathtt{IO}} \bar{y}$ are transitive. That is, we have $\boldsymbol{\nu} q\,(p \leftrightarrows_{\mathtt{IO}} \bar{q} \mid q \leftrightarrows_{\mathtt{IO}} \bar{r}) \gtrsim p \leftrightarrows_{\mathtt{IO}} \bar{r}$ and $\boldsymbol{\nu} y\,(x \leftrightarrows_{\mathtt{IO}} \bar{y} \mid y \leftrightarrows_{\mathtt{IO}} \bar{z}) \gtrsim x \leftrightarrows_{\mathtt{IO}} \bar{z}$.*

Proof. We strengthen the statement and prove the transitivity for chains of wires of any length. The relations we consider for the proof are

$$\mathcal{R}_1 \stackrel{\text{def}}{=} \left\{ (p_0 \leftrightarrows_{\mathtt{IO}} \bar{p}_n, \mathrm{chain}^n_{\mathtt{IO}}(p_0, p_n)) \mid n \geq 2 \right\}$$
$$\mathcal{R}_2 \stackrel{\text{def}}{=} \left\{ (x_0 \leftrightarrows_{\mathtt{IO}} \bar{x}_n, \mathrm{chain}^n_{\mathtt{IO}}(x_0, x_n)) \mid n \geq 2 \right\}.$$

We show that $\mathcal{R}_1 \cup \mathcal{R}_2$ is an expansion up-to $\lesssim$ and context

We first consider the case for location names. Suppose $p_0 \leftrightarrows_{\mathtt{IO}} \bar{p}_m \; \mathcal{R}_1 \; \mathrm{chain}^m_{\mathtt{IO}}(p_0, p_m)$ for some $m \geq 2$. We only consider the case where the process on the right-hand side makes the challenge; the opposite direction can be proved similarly. There is only one possible actions the process can do, namely an input at p_0.

First we prove the following auxiliary statement by induction on n.

> For any $n \geq 2$, if $\mathrm{chain}^n_{\mathtt{IO}}(p_0, p_n) \xrightarrow{p_0(x_0, q_0)} P$, then we have $P \gtrsim \bar{p}_n(x_n, q_n) \colon (\mathrm{chain}^{2n-1}_{\mathtt{IO}}(x_n, x_0) \mid \mathrm{chain}^{2n-1}_{\mathtt{IO}}(q_0, q_n))$

The base case is $n = 2$. Recall that $p_0 \leftrightarrows_{\mathtt{IO}} \bar{p}_1$ and $p_1 \leftrightarrows_{\mathtt{IO}} \bar{p}_0$ are of the form

$$p_0(x_0, q_0).\,(\boldsymbol{\nu} x_1', q_1')(\overline{p_1}(x_1, q_1).\,(x_1 \leftrightarrows_{\mathtt{IO}} \bar{x}_1' \mid q_1' \leftrightarrows_{\mathtt{IO}} \bar{q}_1) \mid x_1 \leftrightarrows_{\mathtt{IO}} \bar{x}_0 \mid q_0 \leftrightarrows_{\mathtt{IO}} \bar{q}_1')$$

and

$$p_1(x_1, q_1).\,(\boldsymbol{\nu} x_2', q_2')(\overline{p_2}(x_2, q_2).\,(x_2 \leftrightarrows_{\mathtt{IO}} \bar{x}_2' \mid q_2' \leftrightarrows_{\mathtt{IO}} \bar{q}_2) \mid x_2' \leftrightarrows_{\mathtt{IO}} \bar{x}_1 \mid q_1 \leftrightarrows_{\mathtt{IO}} \bar{q}_2')).$$

Hence the derivative of the transition $\xrightarrow{p_0(x_0, q_0)}$ is

$$\begin{aligned}
&(\boldsymbol{\nu} x_1', q_1', p_1) \\
&(\overline{p_1}(x_1, q_1).\,(x_1 \leftrightarrows_{\mathtt{IO}} \bar{x}_1' \mid q_1' \leftrightarrows_{\mathtt{IO}} \bar{q}_1) \mid x_1' \leftrightarrows_{\mathtt{IO}} \bar{x}_0 \mid q_0 \leftrightarrows_{\mathtt{IO}} \bar{q}_1' \\
&\mid p_1(x_1, q_1).\,(\boldsymbol{\nu} x_2', q_2')(\overline{p_2}(x_2, q_2).\,(x_2 \leftrightarrows_{\mathtt{IO}} \bar{x}_2' \mid q_2' \leftrightarrows_{\mathtt{IO}} \bar{q}_2) \mid x_2' \leftrightarrows_{\mathtt{IO}} \bar{x}_1 \mid q_1 \leftrightarrows_{\mathtt{IO}} \bar{q}_2')) \\
&\gtrsim (\boldsymbol{\nu} x_1', x_1, x_2', q_1', q_1, q_2') \qquad \text{(interaction at } p_1\text{)} \\
&\quad (x_2' \leftrightarrows_{\mathtt{IO}} \bar{x}_1 \mid x_1 \leftrightarrows_{\mathtt{IO}} \bar{x}_1' \mid x_1' \leftrightarrows_{\mathtt{IO}} \bar{x}_0 \\
&\quad \mid q_0 \leftrightarrows_{\mathtt{IO}} \bar{q}_1' \mid q_1' \leftrightarrows_{\mathtt{IO}} \bar{q}_1 \mid q_1 \leftrightarrows_{\mathtt{IO}} \bar{q}_2' \\
&\quad \mid \overline{p_2}(x_2, q_2).\,(x_2 \leftrightarrows_{\mathtt{IO}} \bar{x}_2' \mid q_2' \leftrightarrows_{\mathtt{IO}} \bar{q}_2)) \\
&\equiv (\boldsymbol{\nu} x_2', q_2')\,(\mathrm{chain}^3_{\mathtt{IO}}(x_2', x_0) \mid \mathrm{chain}^3_{\mathtt{IO}}(q_0, q_2') \mid \overline{p_2}(x_2, q_2).\,(x_2 \leftrightarrows_{\mathtt{IO}} \bar{x}_2' \mid q_2' \leftrightarrows_{\mathtt{IO}} \bar{q}_2)) \\
&= \bar{p}_2(x_2, q_{,2}) \colon (\mathrm{chain}^3_{\mathtt{IO}}(x_2, x_0) \mid \mathrm{chain}^3_{\mathtt{IO}}(q_0, q_2))
\end{aligned}$$

as desired. The inductive case can be proved similarly.

Hence if $\mathrm{chain}^m_{\mathtt{IO}}(p_0, p_m) \xrightarrow{p_0(x_0,q_0)} P$, we have $P \gtrsim \bar{p}_m(x_m, q_m) : (\mathrm{chain}^{2m-1}_{\mathtt{IO}}(x_m, x_0) \mid \mathrm{chain}^{2m-1}_{\mathtt{IO}}(q_0, q_m))$. For the matching transition, we pick

$$p_0 \leftrightarrows_{\mathtt{IO}} \bar{p}_m \xrightarrow{p_0(x_0,q_0)} \bar{p}_m(x_m, q_m) : (x_m \leftrightarrows_{\mathtt{IO}} \bar{x}_0 \mid q_0 \leftrightarrows_{\mathtt{IO}} \bar{q}_m).$$

We can take $C \stackrel{\mathrm{def}}{=} \bar{p}_m(x_m, q_m) : ([\cdot] \mid [\cdot])$ as the common context and conclude this case because $q_0 \leftrightarrows_{\mathtt{IO}} \bar{q}_m \; \mathcal{R}_1 \; \mathrm{chain}^{2m-1}_{\mathtt{IO}}(q_0, q_m)$ and $x_m \leftrightarrows_{\mathtt{IO}} \bar{x}_0 \; \mathcal{R}_2 \; \mathrm{chain}^{2m-1}_{\mathtt{IO}}(x_m, x_0)$.

The case for the variable name is proved similarly. Suppose $x_0 \leftrightarrows_{\mathtt{IO}} \bar{x}_m \; \mathcal{R}_2 \; \mathrm{chain}^m_{\mathtt{IO}}(x_0, x_m)$ for some $m \geq 2$. The only action the two processes can do is the input at x_0. As in the case for location names, we can show that

> For any $n \geq 2$, if $\mathrm{chain}^n_{\mathtt{IO}}(x_0, x_n) \xrightarrow{x_0(p_0)} P$, then $P \gtrsim \mathrm{chain}^n_{\mathtt{IO}}(x_0, x_n) \mid \bar{x}_n(p_n) : \mathrm{chain}^{2n-1}_{\mathtt{IO}}(p_0, p_n)$

by induction on n. We omit the proof as it is similar to the case for location names; instead of the expansion relation for interactions among linear names, the proof uses replication theorems (the laws (1), (4) and (5) of Lemma 2.7). So if $\mathrm{chain}^m_{\mathtt{IO}}(x_0, x_m) \xrightarrow{x_0(p_0)} P$, we can take $x_0 \leftrightarrows_{\mathtt{IO}} \bar{x}_m \xrightarrow{x_0(p_0)} x_0 \leftrightarrows_{\mathtt{IO}} \bar{x}_m \mid \bar{x}_m(p_m) : p_0 \leftrightarrows_{\mathtt{IO}} \bar{p}_m$ as the matching transition. We have $P \gtrsim \mathrm{chain}^n_{\mathtt{IO}}(x_0, x_m) \mid \bar{x}_m(p_m) : \mathrm{chain}^{2n-1}_{\mathtt{IO}}(p_0, p_m)$; $x_0 \leftrightarrows_{\mathtt{IO}} \bar{x}_m \; \mathcal{R}_2 \; \mathrm{chain}^m_{\mathtt{IO}}(x_0,)$; and $p_0 \leftrightarrows_{\mathtt{IO}} \bar{p}_m \; \mathcal{R}_1 \; \mathrm{chain}^{2m-1}_{\mathtt{IO}}(p_0, p_m)$. We can apply the up-to context technique with the context being $[\cdot] \mid \bar{x}_m(p_m) : [\cdot]$ to conclude the case. □

Now we prove the transitivity for the O-I wires. Since the proof is almost identical to that of the I-O wires, we omit the details and only present the key points.

Lemma C.2. *The wires $q \leftrightarrows_{\mathtt{OI}} \overline{p}$ and $x \leftrightarrows_{\mathtt{OI}} \overline{y}$ are transitive. That is, we have $\boldsymbol{\nu} q \, (q \leftrightarrows_{\mathtt{OI}} \overline{p} \mid r \leftrightarrows_{\mathtt{OI}} \overline{q}) \gtrsim r \leftrightarrows_{\mathtt{OI}} \overline{p}$ and $\boldsymbol{\nu} y \, (x \leftrightarrows_{\mathtt{OI}} \overline{y} \mid y \leftrightarrows_{\mathtt{OI}} \overline{z}) \gtrsim x \leftrightarrows_{\mathtt{OI}} \overline{z}$.*

Proof. As in the case of the I-O wires, we consider the following relations.

$$\mathcal{R}_1 \stackrel{\mathrm{def}}{=} \left\{ (p_n \leftrightarrows_{\mathtt{OI}} \overline{p_0}, \mathrm{chain}^n_{\mathtt{OI}}(p_0, p_n)) \mid n \geq 2 \right\}$$
$$\mathcal{R}_2 \stackrel{\mathrm{def}}{=} \left\{ (x_0 \leftrightarrows_{\mathtt{OI}} \overline{x_n}, \mathrm{chain}^n_{\mathtt{OI}}(x_0, x_n)) \mid n \geq 2 \right\}.$$

We show that $\mathcal{R}_1 \cup \mathcal{R}_2$ is an expansion up-to $\lesssim$ and context.

Observe that $\mathrm{chain}^n_{\mathtt{OI}}(p_0, p_n)$ and $\mathrm{chain}^n_{\mathtt{OI}}(x_0, x_n)$ can only do an output at p_0 and input at x_0, respectively. We can show that, for any $n \geq 2$,

(1) if $\mathrm{chain}^n_{\mathtt{OI}}(p_0, p_n) \xrightarrow{\overline{p_0}(x_0,q_0)} P$, $P \gtrsim p_n(x_n, q_n) : (\mathrm{chain}^{2n-1}_{\mathtt{OI}}(x_0, x_n) \mid \mathrm{chain}^{2n-1}_{\mathtt{OI}}(q_n, q_0))$
(2) if $\mathrm{chain}^n_{\mathtt{OI}}(x_0, x_n) \xrightarrow{x_0(p_0)} P$, then $P \gtrsim \mathrm{chain}^n_{\mathtt{OI}}(x_0, x_n) \mid \bar{x}_n(p_n) : \mathrm{chain}^{2n-1}_{\mathtt{OI}}(p_n, p_0)$

by induction on n.

The rest of the proof follows that of Lemma C.1. □

C.2. **Proofs for laws other than transitivity.** We now show the remaining properties holds for all the three concrete wires. Again, the reasoning is similar in all the three cases, though not identical.

Lemma C.3. *The I-O wires $p \leftrightarrows_{\mathtt{IO}} \bar{q}$ and $x \leftrightarrows_{\mathtt{IO}} \bar{y}$ satisfy the laws of Definition 4.1.*

Proof. Requirements 1, 2, and 8 hold by definition. Transitivity of the wires has already been proved (Lemma C.1). Hence, we only check the remaining laws.

We start by checking laws 4 and 5. Law 4 holds because

$$
\begin{aligned}
&\boldsymbol{\nu} q\,(p \leftrightarrows_{\mathtt{IO}} \bar{q} \mid q(x,r)\colon P)\\
&\equiv (\boldsymbol{\nu} q, x, r)\,(p(x'',r'').\,\bar{q}(x',r')\colon (x' \leftrightarrows_{\mathtt{IO}} \bar{x}'' \mid r'' \leftrightarrows_{\mathtt{IO}} \bar{r}')\\
&\qquad\qquad \mid q(x',r').\,(x \leftrightarrows_{\mathtt{IO}} \bar{x}' \mid r' \leftrightarrows_{\mathtt{IO}} \bar{r}) \mid P)\\
&\sim (\boldsymbol{\nu} x, r)\,(p(x'',r'').\,\boldsymbol{\nu} q\,(\bar{q}(x',r')\colon (x' \leftrightarrows_{\mathtt{IO}} \bar{x}'' \mid r'' \leftrightarrows_{\mathtt{IO}} \bar{r}')\\
&\qquad\qquad \mid q(x',r').\,(x \leftrightarrows_{\mathtt{IO}} \bar{x}' \mid r' \leftrightarrows_{\mathtt{IO}} \bar{r})) \mid P)\\
&\gtrsim (\boldsymbol{\nu} x, r)\,(p(x'',r'').\,(\boldsymbol{\nu} x', r')\,(x' \leftrightarrows_{\mathtt{IO}} \bar{x}'' \mid x \leftrightarrows_{\mathtt{IO}} \bar{x}' && \text{(Lemma 3.1)}\\
&\qquad\qquad \mid r'' \leftrightarrows_{\mathtt{IO}} \bar{r}' \mid r' \leftrightarrows_{\mathtt{IO}} \bar{r}) \mid P)\\
&\gtrsim (\boldsymbol{\nu} x, r)\,(p(x'',r'').\,(x \leftrightarrows_{\mathtt{IO}} \bar{x}'' \mid r'' \leftrightarrows_{\mathtt{IO}} \bar{r}) \mid P) && \text{(transitivity of wires)}\\
&= p(x,r)\colon P
\end{aligned}
$$

Note that the assumption of Lemma 3.1 (interaction of permeable prefixes) is fulfilled by the transitivity of wires. Next we consider law 5:

$$
\begin{aligned}
&\boldsymbol{\nu} p\,(p \leftrightarrows_{\mathtt{IO}} \bar{q} \mid \bar{p}(x,r)\colon P)\\
&\equiv (\boldsymbol{\nu} p, x, r)\,(p(x',r').\,\bar{q}(x'',r'')\colon (x'' \leftrightarrows_{\mathtt{IO}} \bar{x}' \mid r' \leftrightarrows_{\mathtt{IO}} \bar{r}'')\\
&\qquad\qquad \mid \bar{p}(x',r').\,(x' \leftrightarrows_{\mathtt{IO}} \bar{x} \mid r \leftrightarrows_{\mathtt{IO}} \bar{r}') \mid P)\\
&\gtrsim (\boldsymbol{\nu} x, r, x', r')\,(\bar{q}(x'',r'')\colon (x'' \leftrightarrows_{\mathtt{IO}} \bar{x}' \mid r' \leftrightarrows_{\mathtt{IO}} \bar{r}'') && \text{(Lemma 3.1)}\\
&\qquad\qquad \mid x' \leftrightarrows_{\mathtt{IO}} \bar{x} \mid r \leftrightarrows_{\mathtt{IO}} \bar{r}' \mid P)\\
&\gtrsim (\boldsymbol{\nu} x, r, x', r', x'', r'')\,(\overline{q}(x''',r''').\,(x''' \leftrightarrows_{\mathtt{IO}} \bar{x}'' \mid r'' \leftrightarrows_{\mathtt{IO}} \bar{r}''') && \text{(def. of } \bar{q}(x'',r'')\text{:)}\\
&\qquad\qquad \mid x'' \leftrightarrows_{\mathtt{IO}} \bar{x}' \mid x' \leftrightarrows_{\mathtt{IO}} \bar{x}\\
&\qquad\qquad \mid r' \leftrightarrows_{\mathtt{IO}} \bar{r}'' \mid r \leftrightarrows_{\mathtt{IO}} \bar{r}' \mid P)\\
&\gtrsim (\boldsymbol{\nu} x, r, x'', r'')\,(\overline{q}(x''',r''').\,(x''' \leftrightarrows_{\mathtt{IO}} \bar{x}'' \mid r'' \leftrightarrows_{\mathtt{IO}} \bar{r}''') && \text{(transitivity of wires)}\\
&\qquad\qquad \mid x'' \leftrightarrows_{\mathtt{IO}} \bar{x} \mid r \leftrightarrows_{\mathtt{IO}} \bar{r}'' \mid P)\\
&\gtrsim (\boldsymbol{\nu} x'', r'')\,(\overline{q}(x''',r''').\,(x''' \leftrightarrows_{\mathtt{IO}} \bar{x}'' \mid r'' \leftrightarrows_{\mathtt{IO}} \bar{r}''') \mid P\{x'',r''/x,r\}) && \text{(assumption on } P\text{)}\\
&\equiv_{\alpha} (\boldsymbol{\nu} x, r)\,(\overline{q}(x',r').\,(x' \leftrightarrows_{\mathtt{IO}} \bar{x} \mid r \leftrightarrows_{\mathtt{IO}} \bar{r}') \mid P)\\
&= \bar{q}(x,r)\colon P
\end{aligned}
$$

We conclude by checking laws 6 and 7. We have

$$\begin{aligned}
\boldsymbol{\nu} y\,(x \underset{\mathtt{I0}}{\leftrightarrow} \bar{y} \mid !y(p).\,P) &= \boldsymbol{\nu} y\,(!x(p').\,\bar{y}(p')\colon p' \underset{\mathtt{I0}}{\leftrightarrow} \bar{p} \mid !y(p).\,P) \\
&\sim !x(p').\,\boldsymbol{\nu} y\,(\bar{y}(p')\colon p' \underset{\mathtt{I0}}{\leftrightarrow} \bar{p} \mid !y(p).\,P) && \text{(replication theorem)} \\
&\gtrsim !x(p').\,\boldsymbol{\nu} p\,(p \underset{\mathtt{I0}}{\leftrightarrow} \bar{p}' \mid P) && \text{(Lemma 3.1 and garbage collection on } y) \\
&\gtrsim !x(p').\,P\{p/p'\} && \text{(assumption on } P) \\
&\equiv_\alpha !x(p).\,P
\end{aligned}$$

and

$$\begin{aligned}
&\boldsymbol{\nu} x\,(x \underset{\mathtt{I0}}{\leftrightarrow} \bar{y} \mid \bar{x}(p)\colon P) \\
&\equiv (\boldsymbol{\nu} x, p)\,(!x(p').\,\bar{y}(p'')\colon p' \underset{\mathtt{I0}}{\leftrightarrow} \bar{p}'' \mid \overline{x}(p').\,p \underset{\mathtt{I0}}{\leftrightarrow} \bar{p}' \mid P) \\
&\gtrsim (\boldsymbol{\nu} p, p')\,(\bar{y}(p'')\colon p' \underset{\mathtt{I0}}{\leftrightarrow} \bar{p}'' \mid p \underset{\mathtt{I0}}{\leftrightarrow} \bar{p}' \mid P) && \text{(interaction at } x \text{ and garbage collection on)} \\
&\gtrsim (\boldsymbol{\nu} p, p', p'')\,(\bar{y}(p''').\,p'' \underset{\mathtt{I0}}{\leftrightarrow} \bar{p}''' \mid p' \underset{\mathtt{I0}}{\leftrightarrow} \bar{p}'' \mid p \underset{\mathtt{I0}}{\leftrightarrow} \bar{p}' \mid P) && \text{(def. of } \bar{y}(p'')\text{:)} \\
&\gtrsim (\boldsymbol{\nu} p, p'')\,(\bar{y}(p''').\,p'' \underset{\mathtt{I0}}{\leftrightarrow} \bar{p}''' \mid p \underset{\mathtt{I0}}{\leftrightarrow} \bar{p}'' \mid P) && \text{(transitivity of wires)} \\
&\gtrsim \boldsymbol{\nu} p''\,(\bar{y}(p''').\,p'' \underset{\mathtt{I0}}{\leftrightarrow} \bar{p}''' \mid P\{p''/p\}) && \text{(assumption on } P) \\
&\equiv_\alpha \boldsymbol{\nu} p\,(\bar{y}(p').\,p \underset{\mathtt{I0}}{\leftrightarrow} \bar{p}' \mid P) \\
&= \bar{y}(p)\colon P. && \square
\end{aligned}$$

The proof for the O-I wires is 'symmetric' to that of the I-O wires.

Lemma C.4. *The O-I wires $p \underset{\mathtt{0I}}{\leftrightarrow} \overline{q}$ and $x \underset{\mathtt{0I}}{\leftrightarrow} \overline{y}$ satisfy the laws of Definition 4.1.*

Proof. We already proved transitivity in Lemma C.2. The requirement 1 and 8 immediately follow from the definition. Law 4 holds because

$$\begin{aligned}
&\boldsymbol{\nu} q\,(p \underset{\mathtt{0I}}{\leftrightarrow} \overline{q} \mid q(x, r)\colon P) \\
&= \boldsymbol{\nu} q\,(p \underset{\mathtt{0I}}{\leftrightarrow} \overline{q} \mid (\boldsymbol{\nu} x, r\,)(q(x', r').\,(x \underset{\mathtt{0I}}{\leftrightarrow} \overline{x'} \mid r' \underset{\mathtt{0I}}{\leftrightarrow} \overline{r}) \mid P)) \\
&\equiv (\boldsymbol{\nu} q, x, r)\,(\overline{q}(x', r').\,p(x'', r'')\colon (x' \underset{\mathtt{0I}}{\leftrightarrow} \overline{x''} \mid r'' \underset{\mathtt{0I}}{\leftrightarrow} \overline{r'}) \\
&\qquad\qquad \mid q(x', r').\,(x \underset{\mathtt{0I}}{\leftrightarrow} \overline{x'} \mid r' \underset{\mathtt{0I}}{\leftrightarrow} \overline{r}) \mid P) \\
&\gtrsim (\boldsymbol{\nu} x, x', r, r')\,(p(x'', r'')\colon (x' \underset{\mathtt{0I}}{\leftrightarrow} \overline{x''} \mid r'' \underset{\mathtt{0I}}{\leftrightarrow} \overline{r'}) && \text{(communication on a linear name)} \\
&\qquad\qquad \mid x \underset{\mathtt{0I}}{\leftrightarrow} \overline{x'} \mid r' \underset{\mathtt{0I}}{\leftrightarrow} \overline{r} \mid P) \\
&\gtrsim (\boldsymbol{\nu} x, x', x'', r, r', r'') && \text{(def. of } p(x'', r'')\colon\text{)} \\
&\quad (p(x''', r''').\,(x'' \underset{\mathtt{0I}}{\leftrightarrow} \overline{x'''} \mid r''' \underset{\mathtt{0I}}{\leftrightarrow} \overline{r''}) \\
&\quad \mid x \underset{\mathtt{0I}}{\leftrightarrow} \overline{x'} \mid x' \underset{\mathtt{0I}}{\leftrightarrow} \overline{x''} \mid r' \underset{\mathtt{0I}}{\leftrightarrow} \overline{r} \mid r'' \underset{\mathtt{0I}}{\leftrightarrow} \overline{r'} \mid P) \\
&\gtrsim (\boldsymbol{\nu} x, x'', r, r'')\,(p(x''', r''').\,(x'' \underset{\mathtt{0I}}{\leftrightarrow} \overline{x'''} \mid r''' \underset{\mathtt{0I}}{\leftrightarrow} \overline{r''}) && \text{(transitivity of wires)} \\
&\qquad\qquad \mid x \underset{\mathtt{0I}}{\leftrightarrow} \overline{x''} \mid r'' \underset{\mathtt{0I}}{\leftrightarrow} \overline{r} \mid P)
\end{aligned}$$

$$
\begin{aligned}
&\gtrsim (\boldsymbol{\nu} x'', r'')\,(p(x''', r'''). (x'' \leftrightarrows_{\mathtt{OI}} \overline{x'''} \mid r''' \leftrightarrows_{\mathtt{OI}} \overline{r''}) && \text{(assumption on } P\text{)}\\
&\qquad\qquad \mid P\{x'', r''/x, r\})\\
&\equiv_\alpha (\boldsymbol{\nu} x, r)\,(p(x', r'). (x \leftrightarrows_{\mathtt{OI}} \overline{x'} \mid r' \leftrightarrows_{\mathtt{OI}} \overline{r}) \mid P)\\
&= p(x, r) \colon P
\end{aligned}
$$

The proof of law 5 is similar to that of 4, but does not use the respectfulness of P :

$$
\begin{aligned}
&\boldsymbol{\nu} p\,(p \leftrightarrows_{\mathtt{OI}} \overline{q} \mid \bar{p}(x, r) \colon P)\\
&\equiv (\boldsymbol{\nu} p, x, r\,)(\overline{q}(x'', r''). p(x', r') \colon (x'' \leftrightarrows_{\mathtt{OI}} \overline{x'} \mid r' \leftrightarrows_{\mathtt{OI}} \overline{r''})\\
&\qquad\qquad \mid \overline{p}(x', r'). (x' \leftrightarrows_{\mathtt{OI}} \overline{x} \mid r \leftrightarrows_{\mathtt{OI}} \overline{r'}) \mid P)\\
&\sim (\boldsymbol{\nu} x, r\,)(\overline{q}(x'', r''). \boldsymbol{\nu} p\,(p(x', r') \colon (x'' \leftrightarrows_{\mathtt{OI}} \overline{x'} \mid r' \leftrightarrows_{\mathtt{OI}} \overline{r''})\\
&\qquad\qquad\qquad\qquad \mid \overline{p}(x', r'). (x' \leftrightarrows_{\mathtt{OI}} \overline{x} \mid r \leftrightarrows_{\mathtt{OI}} \overline{r'})) \mid P)\\
&\gtrsim (\boldsymbol{\nu} x, r\,)(\overline{q}(x'', r''). (\boldsymbol{\nu} x', r'\,)(x'' \leftrightarrows_{\mathtt{OI}} \overline{x'} \mid x' \leftrightarrows_{\mathtt{OI}} \overline{x} \mid r' \leftrightarrows_{\mathtt{OI}} \overline{r''} \mid r \leftrightarrows_{\mathtt{OI}} \overline{r'}) \mid P) && \text{(Lemma 3.1)}\\
&\gtrsim (\boldsymbol{\nu} x, r\,)(\overline{q}(x'', r''). (x'' \leftrightarrows_{\mathtt{OI}} \overline{x} \mid r \leftrightarrows_{\mathtt{OI}} \overline{r''}) \mid P) && \text{(transitivity of wires)}\\
&= \bar{q}(x, r) \colon P
\end{aligned}
$$

The proofs for laws 6 and 7 are the same as those for I-O wires. Law 6 holds because:

$$
\begin{aligned}
\boldsymbol{\nu} y\,(x \leftrightarrows_{\mathtt{OI}} \overline{y} \mid !y(p). P) &= \boldsymbol{\nu} y\,(!x(p'). \bar{y}(p) \colon p' \leftrightarrows_{\mathtt{OI}} \overline{p} \mid !y(p). P)\\
&\sim !x(p'). \boldsymbol{\nu} y\,(\bar{y}(p) \colon p' \leftrightarrows_{\mathtt{OI}} \overline{p} \mid !y(p). P) && \text{(replication theorem)}\\
&\gtrsim !x(p'). \boldsymbol{\nu} p\,(p' \leftrightarrows_{\mathtt{OI}} \overline{p} \mid P) && \text{(Lemma 3.1 and garbage collection)}\\
&\gtrsim !x(p'). P\{p'/p\}. && \text{(assumption on } P\text{)}
\end{aligned}
$$

Finally, law 7 holds because:

$$
\begin{aligned}
&\boldsymbol{\nu} x\,(x \leftrightarrows_{\mathtt{OI}} \overline{y} \mid \bar{x}(p) \colon P)\\
&\equiv (\boldsymbol{\nu} x, p\,)(!x(p'). \bar{y}(p'') \colon p' \leftrightarrows_{\mathtt{OI}} \overline{p''} \mid \overline{x}(p'). p \leftrightarrows_{\mathtt{OI}} \overline{p'} \mid P)\\
&\gtrsim (\boldsymbol{\nu} p\,, p')(\bar{y}(p'') \colon p' \leftrightarrows_{\mathtt{OI}} \overline{p''} \mid p \leftrightarrows_{\mathtt{OI}} \overline{p'} \mid P) && \text{(reduction and garbage collection on } x\text{)}\\
&\gtrsim (\boldsymbol{\nu} p, p', p'')\,(\overline{y}(p'''). p'' \leftrightarrows_{\mathtt{OI}} \overline{p'''} \mid p' \leftrightarrows_{\mathtt{OI}} \overline{p''} \mid p \leftrightarrows_{\mathtt{OI}} \overline{p'} \mid P) && \text{(def. of } \bar{y}(p'')\text{:)}\\
&\gtrsim (\boldsymbol{\nu} p, p'')\,(\overline{y}(p'''). p'' \leftrightarrows_{\mathtt{OI}} \overline{p'''} \mid p \leftrightarrows_{\mathtt{OI}} \overline{p''} \mid P) && \text{(transitivity)}\\
&\gtrsim \boldsymbol{\nu} p''\,(\overline{y}(p'''). p'' \leftrightarrows_{\mathtt{OI}} \overline{p'''} \mid P\{p''/p\}) && \text{(assumption on } P\text{)}\\
&\equiv_\alpha \boldsymbol{\nu} p\,(\overline{y}(p'). p \leftrightarrows_{\mathtt{OI}} \overline{p'} \mid P)\\
&= \bar{y}(p) \colon P
\end{aligned}
$$

□

We conclude the section by checking that also P wires satisfy the desired properties.

Lemma C.5. *The P wires $q \leftrightarrows_{\mathtt{P}} \bar{p}$ and $x \leftrightarrows_{\mathtt{P}} \bar{y}$ satisfy the laws of Definition 4.1.*

Proof. Transitivity has already been proved in Lemma 6.3, and the requirements on free names and the shape of $x \leftrightarrow \bar{y}$ follow from the definition. So we only check the remaining laws.

We now consider law 4. Although the definition of $p \underset{\mathrm{P}}{\leftrightarrow} \bar{q}$ is more complex than that for the other wires, the proof is similar; we can remove the wires using transitivity and the assumption on P.

$$
\begin{aligned}
&\boldsymbol{\nu} q\,(p \underset{\mathrm{P}}{\leftrightarrow} \bar{q} \mid q(x,r)\colon P)\\
&= (\boldsymbol{\nu} x,y,q,r,s\,)\\
&\quad (p(x',r').\,(x \underset{\mathrm{P}}{\leftrightarrow} \bar{x}' \mid r' \underset{\mathrm{P}}{\leftrightarrow} \bar{r})\\
&\quad \mid \bar{q}(y',s').\,(y' \underset{\mathrm{P}}{\leftrightarrow} \bar{y} \mid s \underset{\mathrm{P}}{\leftrightarrow} \bar{s}')\\
&\quad \mid y \underset{\mathrm{P}}{\leftrightarrow} \bar{x} \mid r \underset{\mathrm{P}}{\leftrightarrow} \bar{s}\\
&\quad \mid (\boldsymbol{\nu} x\,,r)(q(x',r').\,(x \underset{\mathrm{P}}{\leftrightarrow} \bar{x}' \mid r' \underset{\mathrm{P}}{\leftrightarrow} \bar{r}) \mid P))\\
&\gtrsim (\boldsymbol{\nu} x,y,y',r,s,s'\,)(p(x',r').\,(x \underset{\mathrm{P}}{\leftrightarrow} \bar{x}' \mid r' \underset{\mathrm{P}}{\leftrightarrow} \bar{r}) && \text{(communication on } q\text{)}\\
&\qquad\qquad \mid y' \underset{\mathrm{P}}{\leftrightarrow} \bar{y} \mid s \underset{\mathrm{P}}{\leftrightarrow} \bar{s}' \mid y \underset{\mathrm{P}}{\leftrightarrow} \bar{x} \mid r \underset{\mathrm{P}}{\leftrightarrow} \bar{s}\\
&\qquad\qquad \mid (\boldsymbol{\nu} x\,,r)(x \underset{\mathrm{P}}{\leftrightarrow} \bar{y}' \mid s' \underset{\mathrm{P}}{\leftrightarrow} \bar{r} \mid P))\\
&\gtrsim (\boldsymbol{\nu} x,y',r,s'\,)(p(x',r').\,(x \underset{\mathrm{P}}{\leftrightarrow} \bar{x}' \mid r' \underset{\mathrm{P}}{\leftrightarrow} \bar{r}) && \text{(transitivity of wires)}\\
&\qquad\qquad \mid y' \underset{\mathrm{P}}{\leftrightarrow} \bar{x} \mid r \underset{\mathrm{P}}{\leftrightarrow} \bar{s}'\\
&\qquad\qquad \mid (\boldsymbol{\nu} x\,,r)(x \underset{\mathrm{P}}{\leftrightarrow} \bar{y}' \mid s' \underset{\mathrm{P}}{\leftrightarrow} \bar{r} \mid P))\\
&\equiv_\alpha (\boldsymbol{\nu} x'',y',r'',s'\,)(p(x',r').\,(x'' \underset{\mathrm{P}}{\leftrightarrow} \bar{x}' \mid r' \underset{\mathrm{P}}{\leftrightarrow} \bar{r}'')\\
&\qquad\qquad \mid y' \underset{\mathrm{P}}{\leftrightarrow} \bar{x}'' \mid r'' \underset{\mathrm{P}}{\leftrightarrow} \bar{s}'\\
&\qquad\qquad \mid (\boldsymbol{\nu} x\,,r)(x \underset{\mathrm{P}}{\leftrightarrow} \bar{y}' \mid s' \underset{\mathrm{P}}{\leftrightarrow} \bar{r} \mid P))\\
&\gtrsim (\boldsymbol{\nu} x,x'',r,r''\,)(p(x',r').\,(x'' \underset{\mathrm{P}}{\leftrightarrow} \bar{x}' \mid r' \underset{\mathrm{P}}{\leftrightarrow} \bar{r}'') && \text{(transitivity of wires)}\\
&\qquad\qquad \mid x \underset{\mathrm{P}}{\leftrightarrow} \bar{x}'' \mid r'' \underset{\mathrm{P}}{\leftrightarrow} \bar{r} \mid P)\\
&\gtrsim (\boldsymbol{\nu} x'',r''\,)(p(x',r').\,(x'' \underset{\mathrm{P}}{\leftrightarrow} \bar{x}' \mid r' \underset{\mathrm{P}}{\leftrightarrow} \bar{r}'') \mid P\{x'',r''/x,r\}) && \text{(assumption on } P\text{)}\\
&\equiv_\alpha (\boldsymbol{\nu} x,r\,)(p(x',r').\,(x \underset{\mathrm{P}}{\leftrightarrow} \bar{x}' \mid r' \underset{\mathrm{P}}{\leftrightarrow} \bar{r}) \mid P)\\
&= p(x,r)\colon P
\end{aligned}
$$

The proof for law 5 is the dual of the previous case.

Now we check law 6. The reasoning is identical to the case for I-O wires and O-I wires.

$$
\begin{aligned}
\boldsymbol{\nu} y\,(x \underset{\mathrm{P}}{\leftrightarrow} \bar{y} \mid !y(p).\,P) &= \boldsymbol{\nu} y\,(!x(p').\,\bar{y}(p)\colon p' \underset{\mathrm{P}}{\leftrightarrow} \bar{p} \mid !y(p).\,P)\\
&\sim !x(p').\,\boldsymbol{\nu} y\,(\bar{y}(p)\colon p' \underset{\mathrm{P}}{\leftrightarrow} \bar{p} \mid !y(p).\,P) && \text{(replication theorem)}\\
&\gtrsim !x(p').\,\boldsymbol{\nu} p\,(p' \underset{\mathrm{P}}{\leftrightarrow} \bar{p} \mid P) && \text{(Lemma 3.1 and garbage collection)}\\
&\gtrsim !x(p').\,P\{p'/p\} && \text{(assumption on } P\text{)}
\end{aligned}
$$

$$\equiv_\alpha !x(p).\, P$$

The last thing to check is law 7. Again, the reasoning is identical to the case for I-O wires and O-I wires.

$$\begin{aligned}
&\boldsymbol{\nu} x\, (x \leftrightarrows_{\mathsf{P}} \bar{y} \mid \bar{x}(p) \colon P) \\
&= \boldsymbol{\nu} x\, (!x(p).\, \bar{y}(q) \colon p \leftrightarrows_{\mathsf{P}} \bar{q} \mid \boldsymbol{\nu} p\, (\overline{x}(p').\, p \leftrightarrows_{\mathsf{P}} \bar{p}' \mid P)) \\
&\gtrsim (\boldsymbol{\nu} p, p' q\,)(\overline{y}(q').\, q \leftrightarrows_{\mathsf{P}} \bar{q}' \mid p' \leftrightarrows_{\mathsf{P}} \bar{q} \mid p \leftrightarrows_{\mathsf{P}} \bar{p}' \mid P) \quad \text{(interaction at } x \text{ and garbage collection)} \\
&\gtrsim (\boldsymbol{\nu} p, q\,)(\overline{y}(q').\, q \leftrightarrows_{\mathsf{P}} \bar{q}' \mid p \leftrightarrows_{\mathsf{P}} \bar{q} \mid P) \quad \text{(transitivity)} \\
&\gtrsim \boldsymbol{\nu} q\, (\overline{y}(q').\, q \leftrightarrows_{\mathsf{P}} \bar{q}' \mid P\{q\!/\!p\}) \quad \text{(assumption on } P\text{)} \\
&\equiv_\alpha \boldsymbol{\nu} p\, (\overline{y}(p').\, p \leftrightarrows_{\mathsf{P}} \bar{p}' \mid P) \\
&= \bar{y}(p) \colon P
\end{aligned}$$

□

Appendix D. Supplementary Materials for Section 7

We present the proofs that were omitted from the main text.

D.1. **Proofs for the properties of encoding of unsolvable terms.** In the main text, we have seen what kind of transition $\mathcal{O}_{\mathtt{OI}}[\![M]\!]_p$ and $\mathcal{O}_{\mathtt{IO}}[\![M]\!]_p$ can do when M is an unsolvable term. Notably, the only visible action these processes can do is an input at p, and the behaviour of $\mathcal{O}_{\mathtt{OI}}$ differs from that of $\mathcal{O}_{\mathtt{IO}}$. We give the proofs for these results.

Lemma 7.2. *Let M be an unsolvable term of order 0. Then the only action $\mathcal{O}_{\mathtt{OI}}[\![M]\!]_p$ can do is a τ-action.*

Proof. If M is unsolvable of order 0 then it must be of the form $(\lambda x.\, M_0)\, \widetilde{M}$, with $\widetilde{M}$ non-empty. Therefore, by definition of $\mathcal{O}_{\mathtt{OI}}$, if M is unsolvable of order 0, $\mathcal{O}_{\mathtt{OI}}[\![M]\!]_p$ cannot do an input at p. This implies that the only transition $\mathcal{O}_{\mathtt{OI}}[\![M]\!]_p$ can do is a τ-transition. □

Lemma 7.1. *Let M be an unsolvable term of order n, where $0 < n \leq \omega$. Then $\mathcal{O}_{\mathtt{OI}}[\![M]\!]_p$ can do a weak input transition at p. Moreover, if $\mathcal{O}_{\mathtt{OI}}[\![M]\!]_p \xRightarrow{p(x,q)} P$, then there exists N such that $P \gtrsim \mathcal{O}_{\mathtt{OI}}[\![N]\!]_q$ and N is an unsolvable of order $n-1$ (under the assumption $\omega - 1 = \omega$).*

Proof. By definition of the order of unsolvables, we have $M \Longrightarrow_{\mathtt{h}} \lambda x.\, M'$ for some M' whose order is $n-1$. By validity of β-reduction (Lemma 5.1), we have $\mathcal{O}_{\mathtt{OI}}[\![M]\!]_p \gtrsim \mathcal{O}_{\mathtt{OI}}[\![\lambda x.\, M']\!]_p$.

First, we show that M can do a weak input transition at p. Since $\mathcal{O}_{\mathtt{OI}}[\![\lambda x.\, M']\!]_p \xrightarrow{p(x,q)} \mathcal{O}_{\mathtt{OI}}[\![M']\!]_q$, we must have a matching transition $\mathcal{O}_{\mathtt{OI}}[\![M]\!]_p \xRightarrow{p(x,q)} P$ for some P.

We remain to show that for every $\mathcal{O}_{\mathtt{OI}}[\![M]\!]_p \xRightarrow{p(x,q)} P$ there is a suitable N with $P \gtrsim \mathcal{O}_{\mathtt{OI}}[\![N]\!]_q$. Assume that $\mathcal{O}_{\mathtt{OI}}[\![M]\!]_p (\xrightarrow{\tau})^n Q \xrightarrow{p(x,q)} P$. Then we have $\mathcal{O}_{\mathtt{OI}}[\![\lambda x.\, M']\!]_p (\xrightarrow{\hat{\tau}})^n Q'$ such that $Q \gtrsim Q'$. By Lemma 5.3, we have $Q' \gtrsim \mathcal{O}_{\mathtt{OI}}[\![\lambda x.\, M'']\!]_p$ for a λ-term $\lambda x.\, M''$ such that $\lambda x.\, M' \Longrightarrow \lambda x.\, M''$. Since M' is unsolvable of order $n-1$, so is M''. We also have $P \gtrsim \mathcal{O}[\![M'']\!]_q$, because $Q \gtrsim \mathcal{O}_{\mathtt{OI}}[\![\lambda x.\, M'']\!]_p$ and $\mathcal{O}_{\mathtt{OI}}[\![\lambda x.\, M'']\!]_p \xrightarrow{p(x,q)} \mathcal{O}_{\mathtt{OI}}[\![M'']\!]_q$ is the only input transition that $\mathcal{O}_{\mathtt{OI}}[\![\lambda x.\, M'']\!]_p$ can do. □

Lemma 7.4.

(1) *If M is an unsolvable of order 0, then $\mathcal{O}_{\mathtt{IO}}[\![M]\!]_p$ can do an input at p. Moreover, if $\mathcal{O}_{\mathtt{IO}}[\![M]\!]_p \xrightarrow{p(x,q)} P$, then $P \gtrsim \mathcal{O}_{\mathtt{IO}}[\![M\,x]\!]_q$.*

(2) *If M is unsolvable, then $\mathcal{O}_{\mathtt{IO}}[\![M]\!]_p$ can do an input at p. Moreover, if $\mathcal{O}_{\mathtt{IO}}[\![M]\!]_p \xrightarrow{p(x,q)} P$, then there exists an unsolvable term M' such that $P \gtrsim \mathcal{O}_{\mathtt{IO}}[\![M']\!]_q$.*

Proof. To prove 1, first observe that M, an unsolvable term of order 0, must be of the form $(\lambda x.\, M_0)\, M_1 \cdots M_n$ for $n \geq 0$. Hence, we have

$$
\begin{aligned}
&\mathcal{O}_{\mathtt{IO}}[\![M]\!]_p \\
&= \boldsymbol{\nu} p_0\, (p_0(x,r) \colon \mathcal{O}_{\mathtt{IO}}[\![M_0]\!]_r \mid \bar{p}_0(x_1,p_1) \colon \cdots \bar{p}_{n-1}(x_n,p_n) \colon \\
&\qquad (!x_1(r_1).\, \mathcal{O}_{\mathtt{IO}}[\![M_1]\!]_{r_1} \mid \cdots \mid !x_n(r_n).\, \mathcal{O}_{\mathtt{IO}}[\![M_n]\!]_{r_n} \mid p \underset{\mathtt{IO}}{\leftrightarrow} \bar{p}_n)) \\
&\xrightarrow{p(x,q)} \equiv \boldsymbol{\nu} p_0\, (p_0(x,r) \colon \mathcal{O}_{\mathtt{IO}}[\![M_0]\!]_r \\
&\qquad\qquad \mid \bar{p}_0(x_1,p_1) \colon \cdots \bar{p}_{n-1}(x_n,p_n) \colon \bar{p}_n(x_{n+1},p_{n+1}) \colon \\
&\qquad\qquad (!x_1(r_1).\, \mathcal{O}_{\mathtt{IO}}[\![M_1]\!]_{r_1} \mid \cdots \mid !x_n(r_n).\, \mathcal{O}_{\mathtt{IO}}[\![M_n]\!]_{r_n} \mid x_{n+1} \underset{\mathtt{IO}}{\leftrightarrow} \bar{x} \mid q \underset{\mathtt{IO}}{\leftrightarrow} \bar{p}_{n+1})) \\
&= \boldsymbol{\nu} p_0\, (p_0(x,r) \colon \mathcal{O}_{\mathtt{IO}}[\![M_0]\!]_r \mid \bar{p}_0(x_1,p_1) \colon \cdots \bar{p}_{n-1}(x_n,p_n) \colon \bar{p}_n(x_{n+1},p_{n+1}) \colon \\
&\qquad ((!x_1(r_1).\, \mathcal{O}_{\mathtt{IO}}[\![M_1]\!]_{r_1} \mid \cdots \mid !x_n(r_n).\, \mathcal{O}_{\mathtt{IO}}[\![M_n]\!]_{r_n} \mid !x_{n+1}(r_{n+1}).\, \mathcal{O}_{\mathtt{IO}}[\![x]\!]_{r_{n+1}} \mid q \underset{\mathtt{IO}}{\leftrightarrow} \bar{p}_{n+1})) \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad (\text{since } x \underset{\mathtt{IO}}{\leftrightarrow} \bar{y} = !x(p).\, \mathcal{O}_{\mathtt{IO}}[\![y]\!]_p) \\
&= \mathcal{O}_{\mathtt{IO}}[\![(\lambda x.\, M_0)\, M_1 \cdots M_n\; x]\!]_p
\end{aligned}
$$

as desired.

We prove (2) by a case analysis on the order of M. If M is an unsolvable of order 0, the claim follows from (1) because $M\,x$ is also an unsolvable of order 0.

Now assume that M is an unsolvable of order $n > 0$ (n may be ω). The fact that $\mathcal{O}_{\mathtt{IO}}[\![M]\!]_p$ can do an input at p follows by the definition of $\mathcal{O}_{\mathtt{IO}}$. We remain to prove the latter claim. Observe that we have $M \Longrightarrow_{\mathtt{h}} \lambda x.\, M'$, and M' must be an unsolvable. By Lemma 5.1, we have $\mathcal{O}_{\mathtt{IO}}[\![M]\!]_p \gtrsim \mathcal{O}_{\mathtt{IO}}[\![\lambda x.\, M']\!]_p$. Thus, if $\mathcal{O}_{\mathtt{IO}}[\![M]\!]_p \xrightarrow{p(x,q)} P$, we have a matching transition $\mathcal{O}_{\mathtt{IO}}[\![\lambda x.\, M']\!]_p \xrightarrow{p(x,q)} Q$ such that $P \gtrsim Q$. By definition of $\mathcal{O}_{\mathtt{IO}}$ and of permeable inputs, and by Lemma B.1, we have $Q \gtrsim \mathcal{O}_{\mathtt{IO}}[\![M']\!]_q$. Hence, $P \gtrsim \mathcal{O}_{\mathtt{IO}}[\![M']\!]_q$ as desired. □

D.2. **Proof for the inverse context lemma.**

Lemma 7.10. *The abstraction and variable contexts of $\mathcal{O}$ have inverse with respect to $\lesssim$,* under the assumption that the every abstraction F that fills the context satisfies $F = \mathcal{O}[\![M]\!]$ for some λ-term M.

Proof. **Abstraction context:** For our encoding, the abstraction context is defined by

$$C_\lambda^x \stackrel{\text{def}}{=} (p)\ (\boldsymbol{\nu} x, q)\ (p(x',q').\, (q' \leftrightarrow \bar{q} \mid x \leftrightarrow \bar{x}') \mid [\cdot]\langle q \rangle).$$

We define the inverse context by

$$D \stackrel{\text{def}}{=} \overline{a}(b,x).\, b(r).\, \boldsymbol{\nu} p\, ([\cdot]\langle p \rangle \mid \overline{p}(x',q').\, (x' \leftrightarrow \bar{x} \mid r \leftrightarrow \bar{q}')).$$

Then

$$
\begin{aligned}
&D[C[F]]\\
&= \overline{a}(b,x).\, b(r).\, \boldsymbol{\nu} p\, ((\boldsymbol{\nu} x, q)\, (p(x',q').\, (q' \leftrightarrow \bar{q} \mid x \leftrightarrow \bar{x}')\\
&\quad \mid F\langle q\rangle) \mid \overline{p}(x',q').\, (x' \leftrightarrow \bar{x} \mid r \leftrightarrow \bar{q}'))\\
&\equiv_\alpha \overline{a}(b,x).\, b(r).\, \boldsymbol{\nu} p\, ((\boldsymbol{\nu} z, q)\, (p(x',q').\, (q' \leftrightarrow \bar{q} \mid z \leftrightarrow \bar{x}') && (z \text{ fresh})\\
&\quad \mid (F\langle q\rangle)\{z/x\}) \mid \overline{p}(x',q').\, (x' \leftrightarrow \bar{x} \mid r \leftrightarrow \bar{q}))\\
&\gtrsim \overline{a}(b,x).\, b(r).\, (\boldsymbol{\nu} z, x', q, q')\, (q \leftrightarrow \bar{q}' \mid z \leftrightarrow \bar{x}' && (\text{communication on } p)\\
&\quad \mid (F\langle q\rangle)\{z/x\} \mid x' \leftrightarrow \bar{x} \mid r \leftrightarrow \bar{q})\\
&\gtrsim \overline{a}(b,x).\, b(r).\, (\boldsymbol{\nu} z, q')(r \leftrightarrow \bar{q}' \mid z \leftrightarrow \bar{x} \mid (F\langle q'\rangle)\{z/x\}) && (\text{transitivity of wires})\\
&\gtrsim \overline{a}(b,x).\, b(r).\, F\langle r\rangle && (\text{Lemma B.1 and } F = \mathcal{O}[\![M]\!])
\end{aligned}
$$

as desired.

Variable context: For a variable context obtained by translating $x\,[\cdot]_1 \cdots [\cdot]_n$ the inverse context D for the i-th hole can be defined by

$$
\begin{aligned}
&\overline{a}(x',b).\, b(r).\, (\boldsymbol{\nu} x, p\,)([\cdot]\langle p\rangle &&.\\
&\mid x(p_0).\, p_0(x_1,p_1).\ldots p_{n-1}(x_n,p_n).\, (x' \leftrightarrow \bar{x} \mid \overline{x_i}(r').\, r \leftrightarrow \bar{r}'))
\end{aligned}
$$

Now let us consider the process $D[C[\widetilde{F}]]$. Since the communication on p_i is a communication on linear names, we can safely execute theses communications. (Note that since permeable input prefixes are encoded using wires, there will be unguarded wires after the reductions) With this in mind, we get

$$
\begin{aligned}
&D[C[\widetilde{F}]]\\
&\gtrsim\ \overline{a}(x',b).\, b(r).\, (\boldsymbol{\nu} x, x_1, \ldots, x_n, x_1', \ldots, x_n', p_n\,)\\
&\quad (!x_1(r_1).\, F_1\langle r_1\rangle \mid \cdots \mid !x_n(r_n).\, F_n\langle r_n\rangle \mid\\
&\quad x_1' \leftrightarrow \bar{x}_1 \mid \cdots \mid x_n' \leftrightarrow \bar{x}_n \mid\\
&\quad p \leftrightarrow \bar{p}_n \mid x' \leftrightarrow \bar{x} \mid \overline{x_i'}(r').\, r \leftrightarrow \bar{r}')\\
&\gtrsim\ \overline{a}(x',b).\, b(r).\, (\boldsymbol{\nu} x, x_1', \ldots, x_n', p_n\,) && (6 \text{ of Definition 4.1 and } F_j = \mathcal{O}[\![M_j]\!])\\
&\quad (!x_1'(r_1).\, F_1\langle r_1\rangle \mid \cdots \mid !x_n'(r_n).\, F_n\langle r_n\rangle \mid\\
&\quad p \leftrightarrow \bar{p}_n \mid x' \leftrightarrow \bar{x} \mid \overline{x_i'}(r').\, r \leftrightarrow \bar{r}')\\
&\sim \overline{a}(x',b).\, b(r).\, (\boldsymbol{\nu} x, x_i')(!x_i'(r_i).\, F_i\langle r_i\rangle \mid x' \leftrightarrow \bar{x} && (\text{garbage collection})\\
&\qquad\qquad \mid \overline{x_i'}(r').\, r \leftrightarrow \bar{r}')\\
&\gtrsim \overline{a}(x',b).\, b(r).\, (\boldsymbol{\nu} x, r'\,)\, \left(F_i\langle r'\rangle \mid x' \leftrightarrow \bar{x} \mid r \leftrightarrow \bar{r}'\right)\\
&\qquad\qquad (\text{communication on } x_i' \text{ and garbage collection})\\
&\gtrsim \overline{a}(x',b).\, b(r).\, (F_i\langle r\rangle)\{x'/x\} && (\text{Lemma B.1 and } F_i = \mathcal{O}[\![M_i]\!])\\
&\equiv_\alpha \overline{a}(x,b).\, b(r).\, F_i\langle r\rangle && \square
\end{aligned}
$$