Remark on Some π Variants

Jianxin Xue[1,2(✉)], Huan Long[2], and Yuxi Fu[2]

[1] College of Computer and Information Engineering, Shanghai Polytechnic University, Shanghai, China
jxxue@sspu.edu.cn

[2] BASICS, MOE-MS Key Laboratory for Intelligent Computing and Intelligent Systems, Department of Computer Science, Shanghai Jiao Tong University, Shanghai, China
longhuan@sjtu.edu.cn, fu-yx@cs.sjtu.edu.cn

Abstract. Two π variants that restrict the use of received names are studied. For either variant the external characterization of the absolute equality is given using a family of bisimulations; the expressive completeness of the calculus is established; and a complete equational proof system is constructed. The relative expressiveness between the two variants and their relationship to π are revealed in terms of subbisimilarity.

1 Introduction

The π-calculus of Milner, Parrow and Walker [MPW92] has proved to be a versatile and robust programming language. Being a prime model of interaction, it accommodates the λ-calculus [Mil92,CF11] and is capable of explaining a wide range of phenomena where dynamic reconfiguration is a fundamental property [Wal95]. The name-passing communication mechanism of π is so strong that most of its variations are able to simulate each other both operationally and observationally to a considerable degree [San93,Tho95,San96a]. It has been an interesting topic to investigate different π-variants from different viewpoints [Pal03,Gor08,FL10]. The results obtained so far are important in that they help improve our understanding of the interaction models at both technical and conceptual levels [Fu16].

There are a number of ways to restrain the power of the π-calculus. Different variants are obtained by considering different forms of the choice operator and the recursion operator. Based on the results on CCS [BGZ03,BGZ04,GSV04], the relative expressive power of these variants have been examined [Pal03, Gor08], the most recent results being reported in [FL10]. Less close relatives are obtained by restricting the usage of the output prefix operator. In the private π-calculus [San96a], denoted by π^P, the exported names are always local names. Apart from its ability to code up higher order processes, the expressive power of π^P is almost unknown. The difficulty is partly attributed to the fact that it has a different set of action labels than the other π-variants. In the asynchronous π-calculus studied in [HT91a,HT91b,Bou92,ACS96] an output prefix is detached from any continuation. The reason why this simple syntactic manipulation provides a semantic modeling of asynchrony is explained

K.G. Larsen et al. (Eds.): SETTA 2017, LNCS 10606, pp. 183–199, 2017.
https://doi.org/10.1007/978-3-319-69483-2_11

in [Fu10] using the theory by process. It is our personal view that the theory of the asynchronous π-calculus is best seen as an example of the theories defined by the processes [Fu10]. More distant cousins of the π-calculus are the process-passing calculi [Tho95, San93]. It is shown in [Fu16] however that these models are too weak from the viewpoint of computation, despite of the result proved in [LPSS08]. To achieve completeness we need to turn the higher order model into abstraction passing calculi [San93, XYL15, Fu17].

Instead of restraining the power of the operators, the calculi that impose conditions on the usage of the received names are the truly proper variants of the name-passing calculus [FZ15]. In the local π-calculus introduced in [Mer00, MS04] a received name can never be used as an input channel. A piece of program that defines a local subroutine with name say f can be rest assured that no other programs or subroutines share the name f. This is certainly a useful safety property and the π^L-calculus is defined in this way. Symmetrically the π^R-calculus imposes the condition that a received name cannot appear as an output channel. A host that intends to send a piece of information to another site may send a local channel name to the site and upload the information using the private channel. In this way the host makes sure that it is not at the receiving end of anything from any aliens. In another variant, called π^S-calculus, a process is not allowed to pass a received name to a third party. This appears as a more fundamental restriction on the name-passing mechanism. In this model the dynamic reconfiguration of the local communication topology is regional. One can never know a secret about you from a third party. Despite of their practical significance these three important variants have not been systematically studied.

Which aspects of π^L, π^R, π^S should we look into? We cannot claim to understand the calculi if we do not know the relative expressiveness between them and the π-calculus. This brings up the issue of model independence since the expressiveness relationship must be defined irrespectively of any particular model. As it turned out, the majority results in process theory are about particular models [Hoa78, Mil89, SW01]. The lack of the emphasis on model independence has been a blocking factor for the development of the process theory. Theory of Interaction proposed in [Fu16] is an attempt to provide a theory for all interaction models. The fundamental part of Theory of Interaction, the theory of equality, the theory of expressiveness, and the theory of completeness have been outlined, and a number of foundational results have been revealed. The applications of this general approach to the value-passing calculus and the name-passing calculus are reported in [Fu13, FZ15]. We will apply the general methodology of Theory of Interaction to π^L, π^R and π^S. The observational theory developed in this manner will help to construct an equational proof system for each of the variants. As a fallout, we will be able to say more about the theory of π^P.

Section 2 defines the semantics of the π-calculus and the three variants. Section 3 characterizes for π^L and π^R the absolute equality in terms of external bisimulation. Section 4 discusses the relative expressiveness of π^L, π^R and π-calculus. Section 5 confirms that all the three variants are legitimate models of interaction. Section 6 describes the equational proof systems for π^L and π^R. Section 7 takes a look at the private π-calculus. Section 8 concludes.

2 Pi and the Variants

Our definition of the π-calculus follows the presentation given in [FZ15]. Throughout the paper, we adopt the following notational conventions.

- The small letters a, b, c, d, e, f, g, h from the beginning of the alphabet range over the infinite set $\mathcal{N}$ of the *names*.
- The lowercase letters u, v, w, x, y, z towards the end of the alphabet range over the infinite set $\mathcal{N}_v$ of the *name variables*.
- The letters l, m, n, o, p, q in the middle of the alphabet range over $\mathcal{N} \cup \mathcal{N}_v$.

We often write $\widetilde{c}$ for a name sequence $c_1, \ldots, c_n$ and similarly $\widetilde{x}$ for $x_1, \ldots, x_n$.

In the π-calculus a name received in a communication can be used as either an output channel, or an input channel, or the content of a further communication. The π-terms are defined by the following grammar:

$$T := \mathbf{0} \mid \sum_{i \in I} n(x).T_i \mid \sum_{i \in I} \overline{n}m_i.T_i \mid T \,|\, T' \mid (c)T \mid [p{=}q]T \mid [p{\neq}q]T \mid !n(x).T \mid !\overline{n}m.T.$$

In the π^L-calculus a received name cannot be used as an input channel. The π^L-terms are inductively generated by following grammar:

$$T := \mathbf{0} \mid \sum_{i \in I} a(x).T_i \mid \sum_{i \in I} \overline{n}m_i.T_i \mid T \,|\, T' \mid (c)T \mid [p{=}q]T \mid [p{\neq}q]T \mid !a(x).T \mid !\overline{n}m.T.$$

In the π^R-calculus a received name cannot appear as an output channel. Its terms are produced by the following grammar:

$$T := \mathbf{0} \mid \sum_{i \in I} n(x).T_i \mid \sum_{i \in I} \overline{a}m_i.T_i \mid T \,|\, T' \mid (c)T \mid [p{=}q]T \mid [p{\neq}q]T \mid !n(x).T \mid !\overline{a}m.T.$$

Finally in the π^S-calculus a received name is not allowed to be transmitted to another process. The grammar is

$$T := \mathbf{0} \mid \sum_{i \in I} n(x).T_i \mid \sum_{i \in I} \overline{n}c_i.T_i \mid T \,|\, T' \mid (c)T \mid [p{=}q]T \mid [p{\neq}q]T \mid !n(x).T \mid !\overline{n}c.T.$$

$\sum_{i \in I} n(x).T_i$ is an input choice term and $\sum_{i \in I} \overline{n}m_i.T_i$ an output choice term. The binder $n(x)$ is an input prefix and $\overline{n}m_i$ an output prefix. The components $n(x).T_i$ and $\overline{n}m_i.T_i$ are called summands. In $n(x).T_i$ the name variable x is bound. A name variable is free if it is not bound. We use guarded replications $!n(x).T$ and $!\overline{n}m.T$. The conditional operator $[p{=}q]$ is a match and $[p{\neq}q]$ a mismatch. The restriction term $(c)T$ is in localization form, where the name c is local. A name is global if it is not local. We will write $gn(\_)$ for the function that returns the set of the global names. The derived prefix operator $\overline{a}(c).T$ abbreviates $(c)\overline{a}c.T$. We assume α-convention, meaning that no misuse of names/name variables ever occurs. The application of a substitution $\sigma = \{n_1/x_1, \ldots, n_i/x_i\}$ to a term is denoted by $T\sigma$. A term is open if it contains free name variables; it is closed otherwise. A process is a closed term. For each π-variant π', the set of the π'-processes is denoted by $\mathcal{P}_{\pi'}$ and is ranged over by L, M, N, O, P, Q.

The semantics of π, π^L, π^R, π^S is defined by the same labeled transition system. The observable action set is $\mathcal{L} = \{ab, \overline{a}b, \overline{a}(c) \mid a, b, c \in \mathcal{N}\}$. The action set $\mathcal{L} \cup \{\tau\}$ is ranged over by λ. The semantic rules are given below.

Action

$$\frac{}{\sum_{i\in I} a(x).T_i \xrightarrow{ac} T_i\{c/x\}} \qquad \frac{}{\sum_{i\in I} \overline{a}c_i.T_i \xrightarrow{\overline{a}c_i} T_i}$$

Composition

$$\frac{T \xrightarrow{\lambda} T'}{S \mid T \xrightarrow{\lambda} S \mid T'} \qquad \frac{S \xrightarrow{ac} S' \quad T \xrightarrow{\overline{a}c} T'}{S \mid T \xrightarrow{\tau} S' \mid T'} \qquad \frac{S \xrightarrow{ac} S' \quad T \xrightarrow{\overline{a}(c)} T'}{S \mid T \xrightarrow{\tau} (c)(S' \mid T')}$$

Localization

$$\frac{T \xrightarrow{\overline{a}c} T'}{(c)T \xrightarrow{\overline{a}(c)} T'} \qquad \frac{T \xrightarrow{\lambda} T'}{(c)T \xrightarrow{\lambda} (c)T'}\, c \notin gn(\lambda)$$

Condition

$$\frac{T \xrightarrow{\lambda} T'}{[a{=}a]T \xrightarrow{\lambda} T'} \qquad \frac{T \xrightarrow{\lambda} T'}{[a{\neq}b]T \xrightarrow{\lambda} T'}$$

Replication

$$\frac{}{!\overline{a}c.T \xrightarrow{\overline{a}c} T \mid !\overline{a}c.T} \qquad \frac{}{!a(x).T \xrightarrow{ac} T\{c/x\} \mid !a(x).T}$$

The notation $\Longrightarrow$ denotes the reflexive and transitive closure of $\xrightarrow{\tau}$.

3 Observational Theory

The first fundamental relationship in process theory is the equality relationship. It is argued in [Fu16] that from the point of view of both computation and interaction, there is only one equality that satisfies the following conditions:

- it is model independent;
- it is an equality for self-evolving and interactive objects.

Self-evolution is the feature of computation and interaction is what a process is supposed to do. We refer the reader to [Fu16] for a detailed argument and convincing technical support for the above remarks and the principles behind the argument. In this short paper we simply repeat the relevant definitions.

Definition 1. *A symmetric relation $\mathcal{R}$ on processes is a* bisimulation *if it validates the following bisimulation property:*

– *If $Q\mathcal{R}P \xrightarrow{\tau} P'$ then one of the following statements is valid:*
  (i) $Q \Longrightarrow Q'$ for some Q' such that $Q'\mathcal{R}P$ and $Q'\mathcal{R}P'$.
  (ii) $Q \Longrightarrow Q''\mathcal{R}P$ for some Q'' such that $\exists Q'.Q'' \xrightarrow{\tau} Q'\mathcal{R}P'$.

It is codivergent *if the following codivergence property is satisfied:*

– *If $Q\mathcal{R}P \xrightarrow{\tau} P_1 \xrightarrow{\tau} \ldots \xrightarrow{\tau} P_i \ldots$ is an infinite computation, then $\exists Q'.\exists i \geqslant 1.Q \overset{\tau}{\Longrightarrow} Q'\mathcal{R}P_i$.*

It is extensional *if the following extensionality property holds:*

1. *if $M\mathcal{R}N$ and $P\mathcal{R}Q$ then $(M \mid P)\mathcal{R}(N \mid Q)$;*
2. *if $P\mathcal{R}Q$ then $(a)P\mathcal{R}(a)Q$ for every $a \in \mathcal{N}$.*

It is equipollent *if $P \Downarrow \Leftrightarrow Q \Downarrow$ whenever $P\mathcal{R}Q$, where $P \Downarrow$, meaning that P is observable, if and only if $P \Longrightarrow\xrightarrow{\lambda} P'$ for some P' and some $\lambda \neq \tau$.*

The bisimulation of the above definition is what van Glabbeek and Weijland called branching bisimulation [vGW89,Bae96]. Codivergence is Priese's eventually progressing property [Pri78]. Equipollence is the most abstract form of Milner and Sangiorgi's barbness condition [MS92]. All the properties introduced in Definition 1 are model independent. Their combination imposes a minimal requirement from the point of computation as well as interaction and a maximal condition from the point of model independence.

Definition 2. *The absolute equality $=$ is the largest relation on processes validating the following statements:*

1. *The relation is reflexive;*
2. *The relation is equipollent, extensional, codivergent and bisimilar.*

The approach to extend the absolute equality from the processes to the terms is standard. The model independence necessarily means that the absolute equality is difficult to work with. If there is a single technical lemma that helps reason about $=$, it must be the Bisimulation Lemma stated next.

Lemma 1. *If $P \Longrightarrow P' = Q$ and $Q \Longrightarrow Q' = P$, then $P = Q$.*

The property stated in Lemma 1 is called X-property by De Nicola, Montanari and Vaandrager [DNMV90].

Once we have defined the absolute equality, we can distinguish two kinds of internal actions. We will write $T \xrightarrow{\iota} T'$ if $T \xrightarrow{\tau} T' \neq T$, and $T \rightarrow T'$ if $T \xrightarrow{\tau} T' = T$.

The observational theory discusses model specific characterizations of the absolute equality. A model dependent counterpart of $=$ is often far more tractable. An external bisimilarity is a Milner-Park style bisimulation [Mil89, Par81] in which every action is explicitly bisimulated. The external characterization of $=$ for the π-calculus is given in [Fu16]. For the π^L-calculus we can give an external counterpart in terms of a family of relations in the style of Sangiorgi's open bisimulations [San96b]. In the following definition $\subseteq_f$ stands for the finite subset relationship.

Definition 3. *A π^L-bisimulation is a family $\{\mathcal{R}_\mathcal{F}\}_{\mathcal{F}\subseteq_f\mathcal{N}}$ of codivergent symmetric bisimulations on $\mathcal{P}_{\pi^L}$ if the following statements are valid whenever $P\mathcal{R}_\mathcal{F}Q$:*

1. *If $P \xrightarrow{ab} P'$ then $Q \Longrightarrow Q'' \xrightarrow{ab} Q'\mathcal{R}_\mathcal{F}P'$ and $P\mathcal{R}_\mathcal{F}Q''$ for some Q', Q''.*
2. *If $P \xrightarrow{\overline{a}b} P'$ and $a \notin \mathcal{F}$ then $Q \Longrightarrow Q'' \xrightarrow{\overline{a}b} Q'\mathcal{R}_\mathcal{F}P'$ and $P\mathcal{R}_\mathcal{F}Q''$ for some Q', Q''.*
3. *If $P \xrightarrow{\overline{a}(c)} P'$ and $a \notin \mathcal{F}$ then $Q \Longrightarrow Q'' \xrightarrow{\overline{a}(c)} Q'\mathcal{R}_{\mathcal{F}\cup\{c\}}P'$ and $P\mathcal{R}_\mathcal{F}Q''$ for some Q', Q''.*

We write $\left\{\simeq_\mathcal{F}^{\pi^L}\right\}_{\mathcal{F}\subseteq_f\mathcal{N}}$ for the largest π^L-bisimulation, each $\simeq_\mathcal{F}^{\pi^L}$ is called the $\mathcal{F}$-π^L-bisimilarity. The $\simeq^{\pi^L}$-bisimilarity $\simeq^{\pi^L}$ is the $\emptyset$-π^L-bisimilarity.

The idea of Definition 3 is that an indexing set $\mathcal{F}$ records all the local names that have been opened up as it were by bound output actions. If $P \simeq_\mathcal{F}^{\pi^L} Q$ and $a \in \mathcal{F}$ then the action $\overline{a}b$ say need not be bisimulated for the reason that no environments that have received the local name a will ever do any input actions at a. A similar idea motivates the following definition.

Definition 4. *A π^R-bisimulation is a family $\{\mathcal{R}_\mathcal{F}\}_{\mathcal{F}\subseteq_f\mathcal{N}}$ of codivergent symmetric bisimulations on $\mathcal{P}_{\pi^R}$ if the followings are valid whenever $P\mathcal{R}_\mathcal{F}Q$:*

1. *If $P \xrightarrow{ab} P'$ and $a \notin \mathcal{F}$ then $Q \Longrightarrow Q'' \xrightarrow{ab} Q'\mathcal{R}_\mathcal{F}P'$ and $P\mathcal{R}_\mathcal{F}Q''$ for some Q', Q''.*
2. *If $P \xrightarrow{\overline{a}b} P'$ then $Q \Longrightarrow Q'' \xrightarrow{\overline{a}b} Q'\mathcal{R}_\mathcal{F}P'$ and $P\mathcal{R}_\mathcal{F}Q''$ for some Q', Q''.*
3. *If $P \xrightarrow{\overline{a}(c)} P'$ then $Q \Longrightarrow Q'' \xrightarrow{\overline{a}(c)} Q'\mathcal{R}_{\mathcal{F}\cup\{c\}}P'$ and $P\mathcal{R}_\mathcal{F}Q''$ for some Q', Q''.*

We write $\left\{\simeq_\mathcal{F}^{\pi^R}\right\}_{\mathcal{F}\subseteq_f\mathcal{N}}$ for the largest π^R-bisimulation, each $\simeq_\mathcal{F}^{\pi^R}$ is called the $\mathcal{F}$-π^R-bisimilarity. The $\simeq^{\pi^R}$-bisimilarity $\simeq^{\pi^R}$ is the $\emptyset$-π^R-bisimilarity.

The proof of the next lemma is routine.

Lemma 2. *Both $\simeq^{\pi^L}$ and $\simeq^{\pi^R}$ are equivalence and congruence relations.*

The next is another useful technical lemma.

Lemma 3. *The following statements are valid:*

1. *If $P \simeq_{\mathcal{F}\cup\{c\}}^{\pi^L} Q$ and c is not a global output channel in $P \mid Q$ then $P \simeq_\mathcal{F}^{\pi^L} Q$;*
2. *If $P \simeq_{\mathcal{F}\cup\{c\}}^{\pi^R} Q$ and c is not a global input channel in $P \mid Q$ then $P \simeq_\mathcal{F}^{\pi^R} Q$;*
3. *If $P \simeq_{\mathcal{F}\cup\{c\}}^{\pi'} Q$ for $\pi' \in \{\pi^L, \pi^R\}$ then $(c)(P \mid A) \simeq_\mathcal{F}^{\pi'} (c)(Q \mid A)$ for each A.*

Without further ado, we come to the main result of this section.

Theorem 1. *The following are valid:*

1. *The π^L-bisimilarity $\simeq^{\pi^L}$ coincides with $=^{\pi^L}$.*
2. *The π^R-bisimilarity $\simeq^{\pi^R}$ coincides with $=^{\pi^R}$.*

Proof. Lemma 2 implies both $\simeq^{\pi^L}\subseteq=^{\pi^L}$ and $\simeq^{\pi^R}\subseteq=^{\pi^R}$. The proof of the reverse inclusions is a modification of a proof in [Fu05]. For the present proof we only have to mention the part that differs from the previous proofs. (2) Let $\{\mathcal{R}_{\mathcal{F}}\}_{\mathcal{F}\subseteq_f\mathcal{N}}$ be defined in the following manner.

$$\mathcal{R}_{\{c_1,\ldots,c_n\}} \stackrel{\mathrm{def}}{=} \left\{ (P,Q) \left| \begin{array}{l} \{a_1,\ldots,a_n\}\cap gn(P\mid Q)=\emptyset, \\ (c_1,\ldots,c_n)(\overline{a_1}c_1\mid\ldots\mid\overline{a_n}c_n\mid P) =^{\pi^R} \\ (c_1,\ldots,c_n)(\overline{a_1}c_1\mid\ldots\mid\overline{a_n}c_n\mid Q) \end{array} \right.\right\}.$$

We prove that $\{\mathcal{R}_{\mathcal{F}}\}_{\mathcal{F}\subseteq_f\mathcal{N}}$ is a π^R-bisimulation. Suppose $A=B$, where

$$A \stackrel{\mathrm{def}}{=} (c_1,\ldots,c_n)(\overline{a_1}c_1\mid\ldots\mid\overline{a_n}c_n\mid P),$$
$$B \stackrel{\mathrm{def}}{=} (c_1,\ldots,c_n)(\overline{a_1}c_1\mid\ldots\mid\overline{a_n}c_n\mid Q),$$

such that $\{a_1,\ldots,a_n\}\cap gn(P\mid Q)=\emptyset$. Consider $P\xrightarrow{\overline{a}(c)}P'$ for some $c\notin\{c_1,\ldots,c_n\}$. Let d,f,a_{n+1} be fresh and let D be defined by

$$D \stackrel{\mathrm{def}}{=} a(x).(\overline{a_{n+1}}x\mid[x\notin gn(P\mid Q)]f)+a(x).d.$$

Now

$$A\,|\,D\xrightarrow{\tau}(c)(A'\mid\overline{a_{n+1}}c\mid[c\notin gn(P\mid Q)]f)$$

must be bisimulated by

$$B\,|\,D\Longrightarrow B''\,|\,D\xrightarrow{\tau}(c)(B'\mid\overline{a_{n+1}}c\mid[c\notin gn(P\mid Q)]f).$$

Since $B''\,|\,D\xrightarrow{\tau}(c)(B'\mid\overline{a_{n+1}}c\mid[c\notin gn(P\mid Q)]f))$ must be a change-of-state, it must be the case that $A\,|\,D=^{\pi^R}B''\,|\,D$. It follows easily from Bisimulation Lemma that $B\Longrightarrow B''\xrightarrow{\overline{a}(c)}B'=^{\pi^R}A'$. Clearly,

$$A'\equiv(c_1,\ldots,c_n)(\overline{a_1}c_1\mid\ldots\mid\overline{a_n}c_n\mid P'),$$
$$B'\equiv(c_1,\ldots,c_n)(\overline{a_1}c_1\mid\ldots\mid\overline{a_n}c_n\mid Q'),$$
$$B''\equiv(c_1,\ldots,c_n)(\overline{a_1}c_1\mid\ldots\mid\overline{a_n}c_n\mid Q'')$$

for some P',Q',Q''. It follows from $B\Longrightarrow B''\xrightarrow{\overline{a}(c)}B'$ that $Q\Longrightarrow Q''\xrightarrow{\overline{a}(c)}Q'$. Moreover $P\mathcal{R}_{\{c_1,\ldots,c_n\}}Q''$ and $P'\mathcal{R}_{\{c_1,\ldots,c_n,c\}}Q'$ by definition. We can symmetrically deal with (1). □

4 Relative Expressiveness

The second fundamental relationship in process theory is the expressiveness relationship between process calculi. For this relationship to make sense at all, model independence has to be a born property. A theory of expressiveness is developed in [Fu16]. The philosophy of the theory is that the expressiveness relationship is the generalization of the absolute equality from one model to two models. Again we will simply repeat the definition here.

Definition 5. *Suppose* $\mathbb{M}, \mathbb{N}$ *are two* π*-variants. A binary relation* $\mathfrak{R} \subseteq \mathcal{P}_{\mathbb{M}} \times \mathcal{P}_{\mathbb{N}}$ *is a* subbisimilarity *if it validates the following statements.*

1. $\mathfrak{R}$ *is reflexive in the following sense:*
   (a) $\mathfrak{R}$ *is total, meaning that* $\forall M \in \mathcal{P}_{\mathbb{M}}.\exists N \in \mathcal{P}_{\mathbb{N}}.M\mathfrak{R}N$.
   (b) $\mathfrak{R}$ *is sound, meaning that* $N_1\mathfrak{R}^{-1}M_1 =_{\mathbb{M}} M_2\mathfrak{R}N_2$ *implies* $N_1 =_{\mathbb{N}} N_2$.
2. $\mathfrak{R}$ *is equipollent, extensional, codivergent and bisimilar.*

We say that $\mathbb{M}$ is subbisimilar to $\mathbb{N}$, notated by $\mathbb{M} \sqsubseteq \mathbb{N}$, if there is a subbisimilarity from $\mathbb{M}$ to $\mathbb{N}$. We write $\mathbb{M} \sqsubset \mathbb{N}$ if $\mathbb{M} \sqsubseteq \mathbb{N}$ and $\mathbb{N} \not\sqsubseteq \mathbb{M}$. Intuitively $\mathbb{M} \sqsubseteq \mathbb{N}$ means that $\mathbb{N}$ is at least as expressive as $\mathbb{M}$.

Theorem 2. *Suppose* $\mathbb{M}, \mathbb{N} \in \{\pi, \pi^L, \pi^R\}$. *If* $\mathbb{M}, \mathbb{N}$ *are distinct then* $\mathbb{M} \not\sqsubseteq \mathbb{N}$.

Proof. Suppose $\mathfrak{F}$ is a subbisimilarity from $\mathbb{M}$ to $\mathbb{N}$. The proof of Theorem 4.23 in [Fu16] essentially shows that, for all P, Q such that $P\mathfrak{F}Q$, the following *Global Bisimulation* property holds:

- If $P \xrightarrow{ac} P'$ then $Q \Longrightarrow Q'' \xrightarrow{ac} Q'\mathfrak{F}^{-1}P'$ and $P\mathfrak{F}Q''$ for some Q'', Q'.
- If $Q \xrightarrow{ac} Q'$ then $P \Longrightarrow P'' \xrightarrow{ac} P'\mathfrak{F}^{-1}Q'$ and $P''\mathfrak{F}Q$ for some Q'', Q'.
- If $P \xrightarrow{\overline{a}c} P'$ then $Q \Longrightarrow Q'' \xrightarrow{\overline{a}c} Q'\mathfrak{F}^{-1}P'$ and $P\mathfrak{F}Q''$ for some Q'', Q'.
- If $Q \xrightarrow{\overline{a}c} Q'$ then $P \Longrightarrow P'' \xrightarrow{\overline{a}c} P'\mathfrak{F}^{-1}Q'$ and $P''\mathfrak{F}Q$ for some Q'', Q'.

Using the above property the following crucial fact is established in [Fu16]:

Self Interpretation: $A\mathfrak{F}A$ whenever A contains no replication operator.

The Global Bisimulation property, the extensionality and Theorem 1 are necessary to guarantee the Self Interpretation property.

Now we can argue as follows:

- It should be clear that $a(x).x \ \mathfrak{F}\ a(x).x$ implies $\pi \not\sqsubseteq \pi^L$, $a(x).\overline{x} \ \mathfrak{F}\ a(x).\overline{x}$ implies $\pi \not\sqsubseteq \pi^R$.
- It follows from the Self Interpretation property that

$$(d)(\overline{a}d \,|\, \overline{d}.\overline{e})\, \mathfrak{F}\, (d)(\overline{a}d \,|\, \overline{d}.\overline{e}), \tag{1}$$

$$(d)(\overline{a}d \,|\, \overline{d})\, \mathfrak{F}\, (d)(\overline{a}d \,|\, \overline{d}). \tag{2}$$

The equality $(d)(\overline{a}d \,|\, \overline{d}.\overline{e}) = (d)(\overline{a}d \,|\, \overline{d})$ holds in π^L. It holds in none of π, π^R. Therefore (1) and (2) imply $\pi^L \not\sqsubseteq \pi$ and $\pi^L \not\sqsubseteq \pi^R$.

– Similarly the Self Interpretation property implies

$$(d)(\overline{a}d \mid d.\overline{e}) \mathfrak{F} (d)(\overline{a}d \mid d.\overline{e}),$$
$$(d)(\overline{a}d \mid d) \mathfrak{F} (d)(\overline{a}d \mid d),$$

from which $\pi^R \not\sqsubseteq \pi$ and $\pi^R \not\sqsubseteq \pi^L$ follow.

We are done. □

5 Expressive Completeness

The variants π^L, π^R, π^S would not be interesting if they are not complete. For interaction models completeness means that the computable functions can be encoded as interactive processes. There are many notions of Turing completeness. In this paper we adopt the definition introduced in [Fu16]. The idea is to formalize the following interactive version of the Church-Turing Thesis:

Axiom of Completeness. $\mathbb{C} \sqsubseteq \mathbb{M}$ for all models of interaction $\mathbb{M}$.

Here $\mathbb{C}$ is the Computability Model, which is basically the interaction model of computable function. The reader is referred to [Fu16] for the definition of $\mathbb{C}$ and what it means for $\mathbb{M}$ to satisfy $\mathbb{C} \sqsubseteq \mathbb{M}$. It is sufficient to say that a proof of completeness boils down to showing how the natural numbers are defined and how the recursive functions are translated into processes that can input natural numbers and output the computing results. To avoid confusion we will write $\underline{0}, \underline{1}, \underline{2}, \ldots, \underline{i}, \ldots$ for the natural numbers. We will use the following notations for the recursive functions defined in [Rog87]:

– $\mathsf{s}(x)$ is the successor function.
– $\underline{i}(x_1, \ldots, x_n)$ is the n-ary constant function with value $\underline{i}$.
– $\mathsf{p}_n^i(x_1, .., x_n)$ is the n-ary projection function at the i-th parameter.
– $\mathsf{f}(\mathsf{f}_1(\widetilde{x}), .., \mathsf{f}_i(\widetilde{x}))$ is the function composed of $\mathsf{f}(x_1, \ldots, x_i), \mathsf{f}_1(\widetilde{x}), .., \mathsf{f}_i(\widetilde{x})$.
– $\mathsf{rec}\ z.[\mathsf{f}(\widetilde{x}, x', z), \mathsf{g}(\widetilde{x})]$ is the recursion function defined by $\mathsf{f}(\widetilde{x}, x', z), \mathsf{g}(\widetilde{x})$.
– $\mu z.\mathsf{f}(\widetilde{x}, z)$ is the minimization function over $\mathsf{f}(\widetilde{x}, z)$.

Theorem 3. *$\mathbb{C} \sqsubseteq \pi^L$, $\mathbb{C} \sqsubseteq \pi^R$ and $\mathbb{C} \sqsubseteq \pi^S$.*

The encoding of the natural numbers in π^L is as $[\![\underline{0}]\!]_c^{\pi^L} \overset{\text{def}}{=} c(z).\overline{z}$ and $[\![\underline{n{+}1}]\!]_c^{\pi^L} \overset{\text{def}}{=} (d)(c(z).\overline{z}d \mid [\![\underline{n}]\!]_d^{\pi^L})$. Every number is accessible at a global name. The encoding makes use of a special name $\bot$. This is harmless because $\bot$ never appears as a channel name in our encoding. Since an input number might be used several time when computing a recursive function, a persistent form of the above encoding is necessary. This is given by $[\![!\underline{0}]\!]_c^{\pi^L} \overset{\text{def}}{=} !c(z).\overline{z}$ and $[\![!\underline{n{+}1}]\!]_c^{\pi^L} \overset{\text{def}}{=} (d)(!c(z).\overline{z}d \mid [\![!\underline{n}]\!]_d^{\pi^L})$. In sequel we shall use Milner's encoding of the polyadic π-prefixes in terms of the monadic prefixes [Mil93]. This is given by

$$\begin{aligned} a(x_1,\cdots,x_k).T &\stackrel{\text{def}}{=} a(z).\overline{z}(c).c(x_1).\cdots.c(x_k).T,\\ \overline{a}\langle n_1,\cdots,n_k\rangle.T &\stackrel{\text{def}}{=} \overline{a}(d).d(y).\overline{y}n_1.\cdots.\overline{y}n_k.T. \end{aligned}$$

It is not a very faithful translation [Fu16]. But this point should not concern us since the above encoding will only be used locally. Now for every π^L-term T, we would like to introduce a π^L-term $Rp(n,a).T$ that is capable of converting an input number to its persistent form. More specifically, it enables the following change-of-state internal action: $[\![\underline{n}]\!]_c^{\pi^L} \,|\, Rp(c,d).T \xrightarrow{\iota}= [\![!\underline{n}]\!]_d^{\pi^L} \,|\, T$. The process $Rp(n,a).T$ is defined by

$$\begin{aligned} &(f)(\overline{n}(c).c(y).([y{=}\bot](!a(z).\overline{z}\bot \,|\, T) \,|\, [y{\neq}\bot]\overline{y}(c).c(z).(d)\overline{f}\langle d,z\rangle.!d(z_1).\overline{z_1}\bot)\\ &\quad |\, !f(u,v).([v{=}\bot](!a(z).\overline{z}u \,|\, T) \,|\, [v{\neq}\bot]\overline{v}(c).c(z).(d)\overline{f}\langle d,z\rangle.!d(z_2).\overline{z_2}u)). \end{aligned}$$

Once we have the process $[\![\underline{n{+}1}]\!]_c^{\pi^L}$ and $[\![!\underline{n{+}1}]\!]_c^{\pi^L}$, we might want to make a copy of them when necessary. This is achieved by $Cp(n,a).T$ defined by

$$\begin{aligned} &(f)(\overline{n}(c).c(y).([y{=}\bot](a(z).\overline{z}\bot \,|\, T) \,|\, [y{\neq}\bot]\overline{y}(c).c(z).(d)\overline{f}\langle d,z\rangle.d(z_1).\overline{z_1}\bot)\\ &\quad |\, !f(u,v).([v{=}\bot](a(z).\overline{z}u \,|\, T) \,|\, [v{\neq}\bot]\overline{v}(c).c(z).(d)\overline{f}\langle d,z\rangle.d(z_2).\overline{z_2}u)), \end{aligned}$$

which is very much similar to the previous process. Clearly the following interactions are admissible.

$$\begin{aligned} [\![\underline{n}]\!]_c^{\pi^L} \,|\, Cp(c,d).T &\xrightarrow{\iota}= [\![\underline{n}]\!]_d^{\pi^L} \,|\, T,\\ [\![!\underline{n}]\!]_c^{\pi^L} \,|\, Cp(c,d).T &\xrightarrow{\iota}= [\![!\underline{n}]\!]_c^{\pi^L} \,|\, [\![\underline{n}]\!]_d^{\pi^L} \,|\, T. \end{aligned}$$

An n-ary function $\mathsf{f}(x_1,\cdots,x_n)$ is translated to a process that picks up n inputs consecutively before outputting the result. The input and output actions must be carried out in particular channels. We write $[\![F^b_{a_1\cdots a_n}(\mathsf{f}(x_1,\cdots,x_n))]\!]^{\pi^L}$ for the translation of $\mathsf{f}(x_1,\cdots,x_n)$ in π^L at the input channels $a_1,\ldots,a_n$ and the output channel b. The structural definition goes as follows:

– The successor, constant, projection and composition functions are defined as follows:

$$\begin{aligned} [\![F^b_{a_1}(s(x))]\!]^{\pi^L} &\stackrel{\text{def}}{=} (d_1)Rp(a_1,d_1).(c)Cp(d_1,c).b(x).\overline{x}(c),\\ [\![F^b_{a_1\cdots a_n}(i_n(x_1,\cdots,x_n))]\!]^{\pi^L} &\stackrel{\text{def}}{=} (d_1)Rp(a_1,d_1).\cdots.(d_n)Rp(a_n,d_n).[\![\underline{i}]\!]_b^{\pi^L},\\ [\![F^b_{a_1\cdots a_n}(\mathsf{p}^i_n(x_1,\cdots,x_n))]\!]^{\pi^L} &\stackrel{\text{def}}{=} (d_1)Rp(a_1,d_1).\cdots.(d_n)Rp(a_n,d_n).Cp(d_i,b),\\ [\![F^b_{a_1\cdots a_n}(\mathsf{f}(\mathsf{f}_1(\widetilde{x}),\cdots,\mathsf{f}_i(\widetilde{x})))]\!]^{\pi^L} &\stackrel{\text{def}}{=} (d_1)Rp(a_1,d_1).\cdots.(d_n)Rp(a_n,d_n).(c_1\cdots c_i)\\ &\quad\ ([\![F^b_{c_1\cdots c_i}(\mathsf{f}(\widetilde{x}))]\!]^{\pi^L} \,|\, [\![F^{c_1}_{d_1\cdots d_n}(\mathsf{f}_1(\widetilde{x}))]\!]^{\pi^L}\\ &\quad\ |\cdots|\, [\![F^{c_i}_{d_1\cdots d_n}(\mathsf{f}_i(\widetilde{x}))]\!]^{\pi^L}) \end{aligned}$$

– $[\![F^b_{a_1\cdots a_{n+1}}(\mathrm{rec}z.[\mathsf{f}(\widetilde{x},x',z),\mathsf{g}(\widetilde{x})])]\!]^{\pi^L}$ is the following process

$$\begin{aligned}&(d_1)Rp(a_1,d_1).\cdots.(d_{n+1})Rp(a_{n+1},d_{n+1}).\overline{d_{n+1}}(c).c(y).\\&\quad([y{=}\bot][\![F^b_{d_1,\cdots,d_n}(\mathsf{g}(\widetilde{x}))]\!]^{\pi^L}\\&\quad\,|\,[y{\neq}\bot](f)(Rec\,|\,(g)\overline{f}\langle g,y\rangle.g(w).[\![F^b_{d_1,\cdots,d_n,w,y}\mathsf{f}(\widetilde{x},x',z)]\!]^{\pi^L}))\end{aligned}$$

where Rec stands for

$$\begin{aligned}&!f(u,v).\overline{u}(d).([v{=}\bot][\![F^d_{d_1,\cdots,d_n}(\mathsf{g}(\widetilde{x}))]\!]^{\pi^L}\\&\quad\,|\,[v{\neq}\bot]\overline{v}(c).c(y).(g)\overline{f}\langle g,y\rangle.(g(w).[\![F^d_{d_1,\cdots,d_n,w,v}(\mathsf{f}(\widetilde{x},x',x''))]\!]^{\pi^L})).\end{aligned}$$

– $[\![F^b_{a_1\cdots a_n}(\mu z.[\mathsf{f}(\widetilde{x},z)])]\!]^{\pi^L}$ is the following process

$$(d_1)Rp(a_1,d_1).\cdots.(d_n)Rp(a_n,d_n).(f)(Mu\,|\,\overline{f}(c).[\![\underline{0}]\!]^{\pi^L}_c)$$

where Mu stands for

$$\begin{aligned}&!f(v).(g)Rp(v,g).(d)([\![F^d_{d_1,\cdots,d_n,g}(\mathsf{f}(\widetilde{x},z))]\!]^{\pi^L}\\&\quad\,|\,\overline{d}(e).e(z).([z{=}\bot]Cp(g,b).\,|\,[z{\neq}\bot]\overline{f}(c).(c')Cp(g,c').c(y).\overline{y}c')).\end{aligned}$$

This completes the definition of $[\![\_]\!]^{\pi^L}$. The reader can work out the encoding $[\![\_]\!]^{\pi^R}$ symmetrically. The proof of the completeness of π^S is subsumed by that for π^P. See Sect. 7 for more details. We only have to remark that the parametric definitions can be implemented in π^S using the replication operator.

6 Proof System

Based on Theorem 1 one may talk about complete equational proof systems for the absolute equality of the π-variants. In view of Theorem 3 no decidable proof system is possible for all processes. However the finite fragment consisting of $\mathbf{0}$, the choice operator, the match/mismatch operator and the localization operator is decidable. Equational systems for various congruence relations on the finite π-processes are well-known. The paper by Parrow and Sangiorgi [PS95] deserves particular attention. A complete equational system AS for the absolute equality on the finite π-terms is studied in [FZ15]. In this section we shall briefly explain how to construct complete systems for π^L and π^R by extending AS.

L	$\overline{n}(c).C[\sum_{i\in I}\overline{c}m_i.T_i] = \overline{n}(c).C[\mathbf{0}]$
R	$\overline{n}(c).C[\sum_{i\in I}c(x).T_i] = \overline{n}(c).C[\mathbf{0}]$

Fig. 1. Axioms for the variants.

In Fig. 1 two axioms are proposed. The law L is valid for $\simeq^{\pi^L}$ and the law R is valid for $\simeq^{\pi^R}$. Let AS_L be $AS \cup \{\mathrm{L}\}$ and AS_R be $AS \cup \{\mathrm{R}\}$. The first indication of the power of AS_L, as well as AS_R, is a normalization lemma stating that all finite terms can be converted to some normal forms. Due to the presence of mobility the normal forms for the π^L-terms are a bit involved. But the main definition is the following.

Definition 6. *Suppose $\mathcal{F}, \mathcal{G} \subseteq_f \mathcal{N} \cup \mathcal{N}_v$. The π^L-term T is a normal form on $\mathcal{F}$ and $\mathcal{G}$ if it is of the form*

$$\sum_{i \in I} \lambda_i.T_i$$

such that for each $i \in I$ one of the followings holds.

1. *If $\lambda_i = \tau$ then T_i is a normal form on $\mathcal{F}$ and $\mathcal{G}$.*
2. *If $\lambda_i = \bar{n}m$ and $n \notin \mathcal{G}$ then T_i is a normal form on $\mathcal{F}$ and $\mathcal{G}$.*
3. *If $\lambda_i = \bar{n}(c)$ and $n \notin \mathcal{G}$ then $T_i \equiv (\bigwedge_{n\in\mathcal{F}} c{\neq}n)T_i^c$ for some normal form T_i^c on $\mathcal{F} \cup \{c\}$ and $\mathcal{G} \cup \{c\}$.*
4. *If $\lambda_i = c(x)$ then T_i is of the form*
$$(\bigwedge_{n\in\mathcal{F}} x{\neq}n)T_i^{\neq} + \sum_{m\in\mathcal{F}} [x{=}m]T_i^m$$
such that $T_i^{\neq}$ is a normal form on $\mathcal{F} \cup \{x\}$ and $\mathcal{G}$, and, for each $m \in \mathcal{F}$, $x \notin fv(T_i^m)$ and T_i^m is a normal form on $\mathcal{F}$ and $\mathcal{G}$.

The reader can easily work out the definition of the normal forms for the π^R-processes from Definitions 3, 4 and 6. The rest of the arguments and proofs are almost an reiteration of corresponding arguments and proofs in [FZ15]. Without further ado, let's state the main result.

Theorem 4. *The following statements are valid:*

1. *$S \simeq^{\pi^L} T$ if and only if $AS_L \vdash \tau.S = \tau.T$ for all finite π^L-terms S, T.*
2. *$S \simeq^{\pi^R} T$ if and only if $AS_R \vdash \tau.S = \tau.T$ for all finite π^R-terms S, T.*

The above theorem is concerned with π^L-terms, from which we can easily derive that $P \simeq^{\pi^L} Q$ if and only if $AS_L \vdash P = Q$ for all finite π^L-processes P, Q and that $P \simeq^{\pi^R} Q$ if and only if $AS_R \vdash P = Q$ for all finite π^R-processes P, Q.

7 Private Pi

The private π-calculus of Sangiorgi [San96a], denoted by π^P, is interesting in that it is a nontrivial model that stays between CCS and the π-calculus. All mobility admitted in π^P is internal. It is shown in [FL10] that π^P fails to be complete if recursion is provided by the replication operator. So normally π^P comes with the parametric definition. Here is its grammar:

$$T := \mathbf{0} \mid \sum_{i\in I} n(x).T_i \mid \sum_{i\in I} \overline{n}(c).T_i \mid T \,|\, T' \mid (c)T \mid D(p_1, \ldots, p_n).$$

A parametric definition is given by

$$D(x_1,\ldots,x_n) = T, \tag{3}$$

where $\{x_1,\ldots,x_n\}$ is the set of all the name variables in T. An instantiation of the parametric definition at $p_1,\ldots,p_n$, denoted by $D(p_1,\ldots,p_n)$, is the term $T\{p_1/x_1,\ldots,p_n/x_n\}$. The match and mismatch operators are absent in π^P because they are useless in this particular model. The operational semantics of π^P can be read off from the semantics of the π-calculus. We will not repeat it here.

The Turing completeness of π^P [BGZ03] does not imply the completeness of π^P since the latter is a strictly stronger property. We still need be assured that π^P is a legitimate model according to the Axiom of Completeness. Unlike in π^L and π^R the natural numbers in π^P must be defined in prefix form.

$$[\![\underline{0}]\!]_p^{\pi^P} \stackrel{\text{def}}{=} \overline{p}(b_0,b_1).\overline{b_0}, \tag{4}$$

$$[\![\underline{n{+}1}]\!]_p^{\pi^P} \stackrel{\text{def}}{=} \overline{p}(b_0,b_1).[\![\underline{n}]\!]_{b_1}^{\pi^P}. \tag{5}$$

To get a feeling of how (4,5) work, let's see how the successor function Sc and the predecessor function Sb are defined.

$$\begin{aligned} Sc(u,v).T &\stackrel{\text{def}}{=} u(x_0,x_1).(\overline{v}(e_0,e_1).\overline{e_1}(f_0,f_1).Com(x_0,x_1,f_0,f_1) \mid T), \\ Sb(u,v).T &\stackrel{\text{def}}{=} u(x_0,x_1).(x_0.\overline{v}(e_0,e_1).\overline{e_0} \\ &\qquad\qquad \mid x_1(y_0,y_1).\overline{v}(e_0,e_1).Com(y_0,y_1,e_0,e_1) \mid T), \end{aligned}$$

where Com is introduced by the following parametric definition:

$$Com(x_0,x_1,y_0,y_1) = x_0.\overline{y_0} \mid x_1(z_0,z_1).\overline{y_1}(c_0,c_1).Com(z_0,z_1,c_0,c_1).$$

The interesting thing about Com is that it works in a lazy fashion. It is only when a produced number is being used can the copy mechanism of Com be invoked. This feature is prominent in all the following encodings. For example the persistent form of the natural number must make use of Com:

$$\begin{aligned} [\![!\underline{0}]\!]_p^{\pi^P} &= [\![!\underline{0}]\!]_p^{\pi^P} \mid [\![\underline{0}]\!]_p^{\pi^P}, \\ [\![!\underline{n{+}1}]\!]_p^{\pi^P} &\stackrel{\text{def}}{=} (c)([\![!\underline{n}]\!]_c^{\pi^P} \mid !PSc(c,p)), \end{aligned}$$

where

$$PSc(u,v).T \stackrel{\text{def}}{=} \overline{v}(e_0,e_1).(u(x_0,x_1).\overline{e_1}(f_0,f_1).Com(x_0,x_1,f_0,f_1) \mid T).$$

The copy term can be simply defined in terms of Com:

$$Cp(u,v).T \stackrel{\text{def}}{=} u(x_0,x_1).(\overline{v}(e_1,e_2).Com(x_0,x_1,e_1,e_2) \mid T).$$

It is not difficult to see that the following actions are admissible:

$$\begin{aligned} [\![\underline{n}]\!]_a^{\pi^P} \,|\, Cp(a,b).T &\xrightarrow{\iota}= [\![\underline{n}]\!]_b^{\pi^P} \,|\, T, \\ [\![!\underline{n}]\!]_a^{\pi^P} \,|\, Cp(a,b).T &\xrightarrow{\iota}= [\![!\underline{n}]\!]_a^{\pi^P} \,|\, [\![\underline{n}]\!]_b^{\pi^P} \,|\, T. \end{aligned}$$

The definition of the replication term is slightly more involved:

$$\begin{aligned} Rp(u,v).T &\overset{\text{def}}{=} u(x_0,x_1).(x_0.[\![!0]\!]_v^{\pi^P} \,|\, x_1(y_0,y_1).(c)(!PSc(c,v) \mid R(y_0,y_1,c)) \,|\, T), \\ R(y_0,y_1,w) &= y_0.[\![!0]\!]_w^{\pi^P} \,|\, y_1(z_0,z_1).(c)(!PSc(c,w) \,|\, R(z_0,z_1,c)). \end{aligned}$$

The reader is advised to verify that the following action is admissible:

$$[\![\underline{n}]\!]_a^{\pi^P} \,|\, Rp(a,b).T \xrightarrow{\iota}= [\![!\underline{n}]\!]_b^{\pi^P} \,|\, T.$$

Another process useful to the encoding is the following:

$$\begin{aligned} Eq(u,v).(T_0,T_1) &\overset{\text{def}}{=} u(x_0,x_1).v(y_0,y_1). \\ &\qquad (d_0d_1)(E(x_0,x_1,y_0,y_1,d_0,d_1) \,|\, \overline{d_0}.T_0 \,|\, \overline{d_1}.T_1), \\ E(x_0,x_1,y_0,y_1,u_0,u_1) &\overset{\text{def}}{=} x_0.(y_0.u_0 \,|\, y_1(z_0,z_1).u_1) \\ &\qquad |\, x_1(z_0,z_1).(y_0.u_1 \,|\, y_1(w_0,w_1).E(z_0,z_1,w_0,w_1,u_0,u_1)). \end{aligned}$$

The encoding $[\![\_]\!]^{\pi^P}$ of the computable functions in π^P can now be given. It should be enough to explain how the recursion functions and the minimization functions are interpreted.

– $[\![F^b_{a_1\cdots a_{n+1}}(\mathrm{rec}z.[\mathrm{f}(\widetilde{x},x',z),\mathrm{g}(\widetilde{x})])]\!]^{\pi^P}$ is defined by the following process

$$\begin{aligned} &(d_1)Rp(a_1,d_1).\cdots.(d_n)Rp(a_n,d_n).(d_{n+1})Rp(a_{n+1},d_{n+1}). \\ &\quad (d_0)([\![!\overline{d_0}(\underline{0})]\!]^{\pi^P} \,|\, Eq(d_0,d_{n+1}).([\![F^b_{\tilde{d}}(\mathrm{g}(\tilde{x}))]\!]^{\pi^P}, \\ &\qquad (e'eg)Sb(d_{n+1},e').Rp(e',e).([\![F^b_{\tilde{d}ge}(\mathrm{f}(\tilde{x},x',x''))]\!]^{\pi^P} \,|\, Rec(g,e,d_0,\widetilde{d})))), \end{aligned}$$

where

$$\begin{aligned} Rec(u,v,z_0,\widetilde{z}) = {} & Eq(z_0,v).([\![F^u_{\tilde{z}}(\mathrm{g}(\tilde{x}))]\!]^{\pi^P}, \\ &\qquad (e'eg)Sb(v,e').Rp(e',e).([\![F^u_{\tilde{z}ge}(\mathrm{f}(\tilde{x},x',x''))]\!]^{\pi^P} \,|\, Rec(g,e,z_0,\widetilde{z}))). \end{aligned}$$

– $[\![F^b_{a_1\cdots a_n}(\mu z.[\mathtt{f}(\widetilde{x},z)])]\!]^{\pi^P}$ is the following process

$$\begin{aligned} &(d_1)Rp(a_1,d_1).\cdots.(d_n)Rp(a_n,d_n). \\ &\quad (d_0d')([\![!\overline{d_0}(\underline{0})]\!]^{\pi^P} \,|\, [\![F^{d'}_{\tilde{d},d_0}(\mathtt{f}(\tilde{x},z))]\!]^{\pi^P} \,|\, Eq(d_0,d').(Cp(d_0,b), \\ &\qquad (e'ef)Sc(d_0,e').Rp(e',e).([\![F^f_{\tilde{d},e}(\mathtt{f}(\tilde{x},z))]\!]^{\pi^P} \,|\, Mu(f,e,d_0,\widetilde{d},b)))), \end{aligned}$$

where

$$\begin{aligned} Mu(u,v,z_0,\widetilde{z},w) = {} & Eq(z_0,u).(Cp(v,w), \\ &\qquad (e'ef)Sc(v,e').Rp(e',e).([\![F^f_{\tilde{d},e}(\mathrm{f}(\tilde{x},z))]\!]^{\pi^P} \,|\, Mu(f,e,z_0,\widetilde{z},w))). \end{aligned}$$

We have coded up all the computable functions. Hence the next result.

Theorem 5. *π^P, and consequently π^S as well, are complete.*

8 Remark

We have provided the observational theories for π^L and π^R. Our attempt to do the same thing for π^S, π^P has not been successful. What has stopped us from getting a similar picture for the latter is the absence of an external characterization of the absolute equality for either variant. It is easy to conceive a family of explicit bisimilarities for π^S. But we probably need a new technique to show that it gives rise to an alternative way of defining the absolute equality in π^S. The external characterization of $=$ for π^P appears more elusive. For one thing the match and the mismatch operators are not of any use in any proof since they are redundant in π^P. So π^P poses a bigger challenge.

The completeness of all the four calculi raises the following question: What kind of universal processes does each of them have? It has been shown in [Fu] that the π-calculus proper has very powerful universal processes. The situations in $\pi^L, \pi^R, \pi^S, \pi^P$ remain to be seen.

Acknowledgments. This work has been supported by National Natural Science Foundation of China (61502296, 61472239, 61261130589) and Natural Science Foundation of Shanghai (15ZR1417000).

References

[ACS96] Amadio, R.M., Castellani, I., Sangiorgi, D.: On bisimulations for the asynchronous π-calculus. In: Montanari, U., Sassone, V. (eds.) CONCUR 1996. LNCS, vol. 1119, pp. 147–162. Springer, Heidelberg (1996). doi:10.1007/3-540-61604-7_53

[Bae96] Baeten, J.: Branching bisimilarity is an equivalence indeed. Inform. Process. Lett. **58**, 141–147 (1996)

[BGZ03] Busi, N., Gabbrielli, M., Zavattaro, G.: Replication vs. recursive definitions in channel based calculi. In: Baeten, J.C.M., Lenstra, J.K., Parrow, J., Woeginger, G.J. (eds.) ICALP 2003. LNCS, vol. 2719, pp. 133–144. Springer, Heidelberg (2003). doi:10.1007/3-540-45061-0_12

[BGZ04] Busi, N., Gabbrielli, M., Zavattaro, G.: Comparing recursion, replication, and iteration in process calculi. In: Díaz, J., Karhumäki, J., Lepistö, A., Sannella, D. (eds.) ICALP 2004. LNCS, vol. 3142, pp. 307–319. Springer, Heidelberg (2004). doi:10.1007/978-3-540-27836-8_28

[Bou92] Boudol, G.: Asynchrony and the π-calculus. Technical report RR-1702, INRIA Sophia-Antipolis (1992)

[CF11] Cai, X., Fu, Y.: The λ-calculus in the π-calculus. Math. Struct. Comput. Sci. **21**, 943–996 (2011)

[DNMV90] Nicola, R., Montanari, U., Vaandrager, F.: Back and forth bisimulations. In: Baeten, J.C.M., Klop, J.W. (eds.) CONCUR 1990. LNCS, vol. 458, pp. 152–165. Springer, Heidelberg (1990). doi:10.1007/BFb0039058

[FL10] Fu, Y., Lu, H.: On the expressiveness of interaction. Theoret. Comput. Sci. **411**, 1387–1451 (2010)

[Fu] Fu, Y.: The universal process. In: Logical Methods in Computer Science (to appear)

[Fu05] Fu, Y.: On quasi open bisimulation. Theoret. Comput. Sci. **338**, 96–126 (2005)

[Fu10] Fu, Y.: Theory by process. In: Gastin, P., Laroussinie, F. (eds.) CONCUR 2010. LNCS, vol. 6269, pp. 403–416. Springer, Heidelberg (2010). doi:10.1007/978-3-642-15375-4_28

[Fu13] Fu, Y.: The Value-Passing Calculus. In: Liu, Z., Woodcock, J., Zhu, H. (eds.) Theories of Programming and Formal Methods. LNCS, vol. 8051, pp. 166–195. Springer, Heidelberg (2013). doi:10.1007/978-3-642-39698-4_11

[Fu16] Fu, Y.: Theory of interaction. Theoret. Comput. Sci. **611**, 1–49 (2016)

[Fu17] Fu, Y.: On the expressive power of name-passing communication. In: CONCUR 2017 (2017)

[FZ15] Fu, Y., Zhu, H.: The name-passing calculus. arXiv:1508.00093 (2015)

[Gor08] Gorla, D.: Comparing communication primitives via their relative expressive power. Inf. Comput. **206**, 931–952 (2008)

[GSV04] Giambiagi, P., Schneider, G., Valencia, F.D.: On the expressiveness of infinite behavior and name scoping in process calculi. In: Walukiewicz, I. (ed.) FoSSaCS 2004. LNCS, vol. 2987, pp. 226–240. Springer, Heidelberg (2004). doi:10.1007/978-3-540-24727-2_17

[Hoa78] Hoare, C.: Communicating sequential processes. Commun. ACM **21**, 666–677 (1978)

[HT91a] Honda, K., Tokoro, M.: An object calculus for asynchronous communication. In: America, P. (ed.) ECOOP 1991. LNCS, vol. 512, pp. 133–147. Springer, Heidelberg (1991). doi:10.1007/BFb0057019

[HT91b] Honda, K., Tokoro, M.: On asynchronous communication semantics. In: Tokoro, M., Nierstrasz, O., Wegner, P. (eds.) ECOOP 1991. LNCS, vol. 612, pp. 21–51. Springer, Heidelberg (1992). doi:10.1007/3-540-55613-3_2

[LPSS08] Lanese, I., Perez, J., Sangiorgi, D., Schmitt, A.: On the expressiveness and decidability of higher-order process calculi. In: Proceedings of LICS 2008, pp. 145–155 (2008)

[Mer00] Merro, M.: Locality in the π-calculus and applications to object-oriented languages. PhD thesis, Ecole des Mines de Paris (2000)

[Mil89] Milner, R.: Communication and Concurrency. Prentice Hall, Upper Saddle River (1989)

[Mil92] Milner, R.: Functions as processes. Math. Struct. Comput. Sci. **2**, 119–146 (1992)

[Mil93] Milner, R.: The polyadic π-calculus: a tutorial. In: Bauer, F.L., Brauer, W., Schwichtenberg, H. (eds.) Logic and Algebra of Specification. NATO ASI Series (Series F: Computer & Systems Sciences), vol. 94, pp. 203–246. Springer, Heidelberg (1993). doi:10.1007/978-3-642-58041-3_6

[MPW92] Milner, R., Parrow, J., Walker, D.: A calculus of mobile processes. Inform. Comput. **100**, 1–40 (Part I), 41–77 (Part II) (1992)

[MS92] Milner, R., Sangiorgi, D.: Barbed bisimulation. In: Kuich, W. (ed.) ICALP 1992. LNCS, vol. 623, pp. 685–695. Springer, Heidelberg (1992). doi:10.1007/3-540-55719-9_114

[MS04] Merro, M., Sangiorgi, D.: On asynchrony in name-passing calculi. Math. Struct. Comput. Sci. **14**, 715–767 (2004)

[Pal03] Palamidessi, C.: Comparing the expressive power of the synchronous and the asynchronous π-calculus. Math. Struct. Comput. Sci. **13**, 685–719 (2003)

[Par81] Park, D.: Concurrency and automata on infinite sequences. In: Deussen, P. (ed.) GI-TCS 1981. LNCS, vol. 104, pp. 167–183. Springer, Heidelberg (1981). doi:10.1007/BFb0017309

[Pri78] Priese, L.: On the concept of simulation in asynchronous, concurrent systems. Progress Cybern. Syst. Res. **7**, 85–92 (1978)

[PS95] Parrow, J., Sangiorgi, D.: Algebraic theories for name-passing calculi. Inf. Comput. **120**, 174–197 (1995)

[Rog87] Rogers, H.: Theory of Recursive Functions and Effective Computability. MIT Press, Cambridge (1987)

[San93] Sangiorgi, D.: From π-calculus to higher-order π-calculus – and back. In: Gaudel, M.-C., Jouannaud, J.-P. (eds.) CAAP 1993. LNCS, vol. 668, pp. 151–166. Springer, Heidelberg (1993). doi:10.1007/3-540-56610-4_62

[San96a] Sangiorgi, D.: π-calculus, internal mobility and agent-passing calculi. Theoret. Comput. Sci. **167**, 235–274 (1996)

[San96b] Sangiorgi, D.: A theory of bisimulation for π-calculus. Acta Informatica **3**, 69–97 (1996)

[SW01] Sangiorgi, D., Walker, D.: The π Calculus: A Theory of Mobile Processes. Cambridge University Press, Cambridge (2001)

[Tho95] Thomsen, B.: A theory of higher order communicating systems. Inf. Comput. **116**, 38–57 (1995)

[vGW89] van Glabbeek, R., Weijland, W.: Branching time and abstraction in bisimulation semantics. In: Information Processing 1989, North-Holland, pp. 613–618 (1989)

[Wal95] Walker, D.: Objects in the π-calculus. Inf. Comput. **116**, 253–271 (1995)

[XYL15] Xu, X., Yin, Q., Long, H.: On the computation power of name parameterization in higher-order processes. In: ICE 2015 (2015)