On the Validity of Encodings of the Synchronous in the Asynchronous π-calculus

Rob J. van Glabbeek[a,b]

[a] *Data61, CSIRO, Sydney, Australia*
[b] *School of Computer Science and Engineering, University of New South Wales, Sydney, Australia*

Abstract

Process calculi may be compared in their expressive power by means of encodings between them. A widely accepted definition of what constitutes a valid encoding for (dis)proving relative expressiveness results between process calculi was proposed by Gorla. Prior to this work, diverse encodability and separation results were generally obtained using distinct, and often incompatible, quality criteria on encodings.

Textbook examples of valid encoding are the encodings proposed by Boudol and by Honda & Tokoro of the synchronous choice-free π-calculus into its asynchronous fragment, illustrating that the latter is no less expressive than the former. Here I formally establish that these encodings indeed satisfy Gorla's criteria.

Keywords: process calculi, expressiveness, quality criteria for encodings, valid encoding, π-calculus

1. Introduction

Since the late 1970s, a large number of process calculi have been proposed, such as CCS [29], CSP [8], ACP [3], SCCS [30], MEIJE [1], LOTOS [4], the π-calculus [32], mobile ambients [14] and mCRL2 [24]. To cater to specific applications, moreover many variants of these calculi were created, including versions incorporating notions of time, and probabilistic choice.

To order these calculi w.r.t. expressiveness, encodings between them have been studied [50, 48, 51, 20, 5, 39, 35, 36, 34, 14, 47, 9, 13, 2, 38, 37, 45, 12, 53, 11, 25, 46, 52, 26, 42]. Process calculus $\mathcal{L}_1$ is said to be *at least as expressive as* process calculus $\mathcal{L}_2$ iff there exists a valid encoding from $\mathcal{L}_1$ into $\mathcal{L}_2$. However, in proving that one languages is—or is not—at least as expressive as another, different authors have used different, and often incomparable, criteria.

Gorla [23] collected some essential features of the above approaches and integrated them in a proposal for a valid encoding that justifies many encodings and separation results from the literature. Since then, many authors have used Gorla's framework as a basis for establishing new valid encodings and separation results [22, 28, 44, 41, 43, 16, 17, 18, 19].

Often quoted token examples of valid encodings [36, 34, 10, 13, 12] are the encodings proposed by Boudol [6] and by Honda & Tokoro [27] of the synchronous choice-free π-calculus into its asynchronous fragment, illustrating that the latter is as expressive as the former. Gorla mentions these encodings among his first three examples of encodings that satisfy his criteria for validity [23], thereby giving evidence in support of his combination of criteria, more than in support of these encodings. Nevertheless, I have not found a proof in the literature that these encodings satisfy Gorla's notion of validity, nor is the matter trivial. The goal of this paper is fill this gap and formally establish that the encodings of [6, 27] indeed are valid à la Gorla.

Section 2 recalls Gorla's proposal for validity of an encoding; for their motivation see [23]. Section 3 presents the encodings of [6] and [27], again suppressing motivation, and Sections 4–5 establish their validity. Section 6 reflects back on Gorla's criteria in the light of the present application, and compares with the notion of a valid encoding from [21].

2. Valid encodings

In [23] a *process calculus* is given as a triple $\mathcal{L}=(\mathcal{P},\longmapsto,\asymp)$, where

- $\mathcal{P}$ is the set of language terms (called *processes*), built up from k-ary composition operators op.
- $\longmapsto$ is a binary *reduction* relation between processes.
- $\asymp$ is a semantic equivalence on processes.

The operators themselves may be constructed from a set $\mathcal{N}$ of names. In the π-calculus, for instance, there is a unary operator $\bar{x}y.\_$ for each pair of names $x,y\in\mathcal{N}$. This way names occur in processes; the occurrences of names in processes are distinguished in *free* and *bound* ones; $\mathrm{fn}(\vec{P})$ denotes the set of names occurring free in the k-tuple of processes $\vec{P}=(P_1,\ldots,P_k)\in\mathcal{P}^k$. A *renaming* is a function $\sigma:\mathcal{N}\to\mathcal{N}$; it extends componentwise to k-tuples of names. If $P\in\mathcal{P}$ and σ is a renaming, then $P\sigma$ denotes the term P in which each free occurrence of a name x is replaced by $\sigma(x)$, while renaming bound names to avoid name capture.

A k-ary $\mathcal{L}$-*context* $C[\_{1};\ldots;\_{k}]$ is a term build by the composition operators of $\mathcal{L}$ from *holes* $\_{1},\ldots,\_{k}$; each of these holes must occur exactly once in the context. If $C[\_1;\ldots;\_k]$ is a k-ary $\mathcal{L}$-*context* and $P_1,\ldots,P_k \in \mathcal{P}$ then $C[P_1;\ldots;P_k]$ denotes the result of substituting P_i for $\_i$ for each $i=1,\ldots,k$, while renaming bound names to avoid capture.

Let $\Longmapsto$ denote the reflexive-transitive closure of $\longmapsto$. One writes $P \longmapsto^{\omega}$ if P *diverges*, that is, if there are P_i for $i \in \mathbb{N}$ such that $P = P_0$ and $P_i \longmapsto P_{i+1}$ for all $i \in \mathbb{N}$. Finally, write $P \longmapsto$ if $P \longmapsto Q$ for some term Q.

For the purpose of comparing the expressiveness of languages, a constant $\surd$ is added to each of them [23]. A term P in the upgraded language is said to *report success*, written $P{\downarrow}$, if it has an *top-level unguarded* occurrence of $\surd$.[1] Write $P{\Downarrow}$ if $P \Longmapsto P'$ for a process P' with $P'{\downarrow}$.

Definition 1 ([23])**.** An *encoding* of $\mathcal{L}_1 = (\mathcal{P}_1, \longmapsto_1, \asymp_1)$ into $\mathcal{L}_2 = (\mathcal{P}_2, \longmapsto_2, \asymp_2)$ is a pair $(\llbracket\cdot\rrbracket, \varphi_{\llbracket\,\rrbracket})$ where $\llbracket\cdot\rrbracket : \mathcal{P}_1 \to \mathcal{P}_2$ is called *translation* and $\varphi_{\llbracket\,\rrbracket} : \mathcal{N} \to \mathcal{N}^k$ for some $k \in \mathbb{N}$ is called *renaming policy* and is such that for $u \neq v$ the k-tuples $\varphi_{\llbracket\,\rrbracket}(u)$ and $\varphi_{\llbracket\,\rrbracket}(v)$ have no name in common.

The terms of the source and target languages $\mathcal{L}_1$ and $\mathcal{L}_2$ are often called S and T, respectively.

Definition 2 ([23])**.** An encoding is *valid* if it satisfies the following five criteria.

1. *Compositionality:* for every k-ary operator op of $\mathcal{L}_1$ and for every set of names $N \subseteq \mathcal{N}$, there exists a k-ary context $C^N_{\mathsf{op}}[\_1;\ldots;\_k]$ such that

$$\llbracket \mathsf{op}(S_1,\ldots,S_k)\rrbracket = C^N_{\mathsf{op}}(\llbracket S_1\rrbracket;\ldots;\llbracket S_k\rrbracket)$$

for all $S_1,\ldots,S_k \in \mathcal{P}_1$ with $\mathrm{fn}(S_1,\ldots,S_n) = N$.

2. *Name invariance:* for every $S \in \mathcal{P}_1$ and $\sigma : \mathcal{N} \to \mathcal{N}$

$$\begin{array}{lcll}\llbracket S\sigma\rrbracket & = & \llbracket S\rrbracket\sigma' & \text{if } \sigma \text{ is injective}\\ \llbracket S\sigma\rrbracket & \asymp_2 & \llbracket S\rrbracket\sigma' & \text{otherwise}\end{array}$$

with σ' such that $\varphi_{\llbracket\,\rrbracket}(\sigma(a)) = \sigma'(\varphi_{\llbracket\,\rrbracket}(a))$ for all $a \in \mathcal{N}$.

3. *Operational correspondence:*
Completeness if $S \Longmapsto_1 S'$ then $\llbracket S\rrbracket \Longmapsto_2 \asymp_2 \llbracket S'\rrbracket$
Soundness and if $\llbracket S\rrbracket \Longmapsto_2 T$ then $\exists S'$: $S \Longmapsto_1 S'$ and $T \Longmapsto_2 \asymp_2 \llbracket S'\rrbracket$.

4. *Divergence reflection:* if $\llbracket S\rrbracket \longmapsto_2^{\omega}$ then $S \longmapsto_1^{\omega}$.

5. *Success sensitiveness:* $S{\Downarrow}$ iff $\llbracket S\rrbracket{\Downarrow}$.
For this purpose $\llbracket\cdot\rrbracket$ is extended to deal with the added constant $\surd$ by taking $\llbracket\surd\rrbracket = \surd$.

[1]Gorla defines the latter concept only for languages that are equipped with a notion of *structural congruence* $\equiv$ as well as a parallel composition $|$. In that case P has a top-level unguarded occurrence of $\surd$ iff $P \equiv Q|\surd$, for some Q [23]. Specialised to the π-calculus, a *(top-level) unguarded* occurrence is one that not lays strictly within a subterm $\alpha.Q$, where α is τ, $\bar{x}y$ or $x(z)$. For De Simone languages [50], even when not equipped with $\equiv$ and $|$, a suitable notion of an unguarded occurrence is defined in [51].

3. Encoding synchronous into asynchronous π

Consider the π-calculus as presented by Milner in [31], i.e., the one of Sangiorgi and Walker [49] without matching, τ-prefixing, and choice.

Given a set of *names* $\mathcal{N}$, the set $\mathcal{P}_\pi$ of *processes* or *terms* P of the calculus is given by

$$P ::= \mathbf{0} \quad|\quad \bar{x}y.P \quad|\quad x(z).P \quad|\quad P|P' \quad|\quad (z)P \quad|\quad !P$$

with x,y,z,u,v,w ranging over $\mathcal{N}$.

Definition 3. An occurrence of a name z in π-calculus process $P \in \mathcal{P}_\pi$ is *bound* if it lays within a subexpression $x(z).P'$ or $(z)P'$ of P; otherwise it is *free*. Let $\mathrm{n}(P)$ be the set of names occurring in $P \in \mathcal{P}_\pi$, and $\mathrm{fn}(P)$ (resp. $\mathrm{bn}(P)$) be the set of names occurring free (resp. bound) in P.

Structural congruence, $\equiv$, is the smallest congruence relation on processes satisfying

$$\begin{array}{llcl|lcll}
(1) & P_1|(P_2|P_3) & \equiv & (P_1|P_2)|P_3 & (z)\mathbf{0} & \equiv & \mathbf{0} & (5)\\
(2) & P_1|P_2 & \equiv & P_2|P_1 & (z)(u)P & \equiv & (u)(z)P & (6)\\
(3) & P|\mathbf{0} & \equiv & P & (w)(P|Q) & \equiv & P|(w)Q & (7)\\
 & & & & (z)P & \equiv & (w)P\{w/z\} & (8)\\
(4) & !P & \equiv & P|!P & x(z).P & \equiv & x(w).P\{w/z\}. & (9)
\end{array}$$

Here $w \notin \mathrm{n}(P)$, and $P\{w/z\}$ denotes the process obtained by replacing each free occurrence of z in P by w.

Definition 4. The *reduction relation*, $\longmapsto \subseteq \mathcal{P}_\pi \times \mathcal{P}_\pi$, is generated by the following rules.

$$\frac{y \notin \mathrm{bn}(Q)}{\bar{x}y.P|x(z).Q \longmapsto P|Q\{y/z\}} \qquad \frac{P \longmapsto P'}{P|Q \longmapsto P'|Q}$$

$$\frac{P \longmapsto P'}{(z)P \longmapsto (z)P'} \qquad \frac{Q \equiv P \quad P \longmapsto P' \quad P' \equiv Q'}{Q \longmapsto Q'}$$

The asynchronous π-calculus, as introduced by Honda & Tokoro in [27] and by Boudol in [6], is the sublanguage aπ of the fragment π of the π-calculus presented above where all subexpressions $\bar{x}y.P$ have the form $\bar{x}y.\mathbf{0}$, and are written $\bar{x}y$.

Boudol [6] defined an encoding $\llbracket\cdot\rrbracket_{\mathrm{B}}$ from π to aπ inductively as follows:

$$\begin{array}{rcl}
\llbracket\mathbf{0}\rrbracket_{\mathrm{B}} & = & \mathbf{0}\\
\llbracket\bar{x}y.P\rrbracket_{\mathrm{B}} & = & (u)(\bar{x}u|u(v).(\bar{v}y|\llbracket P\rrbracket_{\mathrm{B}}))\\
\llbracket x(z).P\rrbracket_{\mathrm{B}} & = & x(u).(v)(\bar{u}v|v(z).\llbracket P\rrbracket_{\mathrm{B}})\\
\llbracket P|Q\rrbracket_{\mathrm{B}} & = & (\llbracket P\rrbracket_{\mathrm{B}}|\llbracket Q\rrbracket_{\mathrm{B}})\\
\llbracket !P\rrbracket_{\mathrm{B}} & = & !\llbracket P\rrbracket_{\mathrm{B}}\\
\llbracket (x)P\rrbracket_{\mathrm{B}} & = & (x)\llbracket P\rrbracket_{\mathrm{B}}
\end{array}$$

always choosing $u,v \notin \mathrm{fn}(P)\cup\{x,y\}$, $u \neq v$. The encoding $\llbracket\cdot\rrbracket_{\mathrm{HT}}$ of Honda & Tokoro [27] differs only in the clauses for the input and output prefix:

$$\begin{array}{rcl}
\llbracket\bar{x}y.P\rrbracket_{\mathrm{HT}} & = & x(u).(\bar{u}y|\llbracket P\rrbracket_{\mathrm{HT}})\\
\llbracket x(z).P\rrbracket_{\mathrm{HT}} & = & (u)(\bar{x}u|u(z).\llbracket P\rrbracket_{\mathrm{HT}})
\end{array}$$

again choosing $u \notin \mathrm{fn}(P)\cup\{x,y\}$.

4. Validity of Boudol's encoding

In this section I show that Boudol's encoding satisfies all five criteria of Gorla [23]. I will drop the subscript $_{\mathrm{B}}$.

4.1. Compositionality

Boudol's encoding is compositional by construction, for it is *defined* in terms of the contexts C^N_{op} that are required to exist by Definition 2. Note that, for the cases of input and output prefixing, these contexts *do* depend on N, namely through the requirement that the fresh names u and v are chosen to lay outside N.

4.2. Name invariance

An encoding according to Gorla is a pair $(\llbracket\cdot\rrbracket, \varphi_{\llbracket\,\rrbracket})$, of which the second component, the renaming policy, is relevant only for satisfying the criterion of name invariance. Here I take $k=1$ and $\varphi_{\llbracket\,\rrbracket} : \mathcal{N} \rightarrow \mathcal{N}$ the identity mapping.

Lemma 1. *Let $S \in \mathcal{P}_\pi$. Then* $\mathrm{fn}(\llbracket S \rrbracket) = \mathrm{fn}(S)$.
Moreover, $\llbracket S \rrbracket\{y/z\} = \llbracket S\{y/z\}\rrbracket$ for any $y, z \in \mathcal{N}$.

Proof. A straightforward structural induction on S. ☐

This implies that $\llbracket S\sigma \rrbracket = \llbracket S \rrbracket\sigma$ for any renaming $\sigma : \mathcal{N} \rightarrow \mathcal{N}$, injective or otherwise. So the criterion of name invariance is satisfied.

4.3. Operational correspondence

A process calculus à la Gorla is a triple $\mathcal{L} = (\mathcal{P}, \longmapsto, \asymp)$; so far I defined $\mathcal{P}$ and $\longmapsto$ only. The semantic equivalence $\asymp$ of the source language plays no rôle in assessing whether an encoding is valid; the one of the target language is used only for satisfying the criteria of name invariance and operational correspondence. Here I take $\asymp_\pi$ and $\asymp_{\mathrm{a}\pi}$ the identity relations.

If $S \equiv S'$ for $S, S' \in \mathcal{P}_\pi$ then there exists a sequence $S_0 \equiv S_1 \equiv \cdots \equiv S_n$ for some $n \geq 0$, with $S = S_0$ and $S' = S_n$, such that each step $S_i \equiv S_{i+1}$ for $0 \leq i < n$ is an application of one of the rules (1)–(9) of Definition 4 or their symmetric counterparts $\overleftarrow{(1)}$–$\overleftarrow{(9)}$. (In fact, there is no need for rules $\overleftarrow{(2)}$,$\overleftarrow{(6)}$,$\overleftarrow{(8)}$ and $\overleftarrow{(9)}$ as rules (2),(6),(8) and (9) are their own symmetric counterparts.) Being an application of a rule $L \equiv R$ here means that $S_i = C[L]$ and $S_{i+1} = C[R]$ for some unary context $C[\_{}_1]$.

Operational completeness.

Lemma 2. *If $S \equiv S'$ for $S, S' \in \mathcal{P}_\pi$ then $\llbracket S \rrbracket \equiv \llbracket S' \rrbracket$.*

Proof. Using the reflexivity, symmetry and transitivity of $\equiv$ one may restrict attention to the case that $S \equiv S'$ is a single application of a rule (1)–(9) of Definition 4. The proof proceeds by structural induction on the context $C[\_{}_1]$. The case that $C[\_{}_1] = \_{}_1$, the *trivial context*, is straightforward for each of the rules (1)–(9), applying Lemma 1 in the cases of rules (8),(9). The induction step is a straightforward consequence of the compositionality of $\llbracket\cdot\rrbracket$. ☐

Lemma 3. *Let $S, S' \in \mathcal{P}_\pi$. If $S \longmapsto S'$ then $\llbracket S \rrbracket \Longmapsto \llbracket S' \rrbracket$.*

Proof. By induction on the derivation of $S \longmapsto S'$.

- Let $S = \bar{x}y.P|x(z).Q$, $y \notin \mathrm{bn}(Q)$ and $S' = P|Q\{y/z\}$. Pick $u, v \notin \mathrm{fn}(P) \cup \mathrm{fn}(Q)$, with $u \neq v$. Write $P^* := \bar{v}y|\llbracket P \rrbracket$ and $Q^* := v(z).\llbracket Q \rrbracket$. Then

$$\begin{array}{rcl} \llbracket S \rrbracket & = & (u)(\bar{x}u|u(v).P^*) \mid x(u).(v)(\bar{u}v|Q^*) \\ & \longmapsto & (u)\big(u(v).P^* \mid (v)(\bar{u}v|Q^*)\big) \\ & \longmapsto & (v)(P^* \mid Q^*) \\ & \longmapsto & \llbracket P \rrbracket \mid (\llbracket Q \rrbracket\{y/z\}) \\ & = & \llbracket P \rrbracket \mid \llbracket Q\{y/z\} \rrbracket \qquad \text{(using Lemma 1)} \\ & = & \llbracket P \mid Q\{y/z\} \rrbracket = \llbracket S' \rrbracket. \end{array}$$

  Here structural congruence is applied in omitting parallel components $\mathbf{0}$ and empty binders (u), (v).

- Let $S = (z)P$ and $S' = (z)P'$, with $P \longmapsto P'$. By the induction hypothesis, $\llbracket P \rrbracket \Longmapsto \llbracket P' \rrbracket$. Therefore, $\llbracket S \rrbracket \Longmapsto \llbracket S' \rrbracket$, as $\llbracket S \rrbracket = (z)\llbracket P \rrbracket$ and $\llbracket S' \rrbracket = (z)\llbracket P' \rrbracket$.

- The case that $S = P|Q$ and $S' = P'|Q$ with $P \longmapsto P'$ proceeds likewise.

- Let $S \equiv P$ and $P' \equiv S'$ with $P \longmapsto P'$. By the induction hypothesis, $\llbracket P \rrbracket \Longmapsto \llbracket P' \rrbracket$. By Lemma 2, $\llbracket S \rrbracket \equiv \llbracket P \rrbracket$ and $\llbracket P' \rrbracket \equiv \llbracket S' \rrbracket$. So $\llbracket S \rrbracket \Longmapsto \llbracket S' \rrbracket$. ☐

The above yields that $S \Longmapsto S'$ implies $\llbracket S \rrbracket \Longmapsto \llbracket S' \rrbracket$. So the criterion of operational completeness is satisfied.

Remark 1. The above proof shows that $\Longmapsto$ in Lemma 3 may be replaced by $\longmapsto\!\longmapsto\!\longmapsto$. As a direct consequence $S \longmapsto^\omega$ implies $\llbracket S \rrbracket \longmapsto^\omega$ (*divergence preservation*).

Operational soundness. The following result provides a normal form up to structural congruence for reduction steps in the asynchronous π-calculus. Here a term is *plain* if it is a parallel composition $P_1|\ldots|P_n$ of subterms P_i of the form $\bar{x}y.R$ or $x(z).R$ or $\surd$ or $\mathbf{0}$ or $!R$. Moreover, $(\tilde{w})P$ for $\tilde{w} = \{w_1, \ldots, w_n\} \subseteq \mathcal{N}$ with $n \in \mathbb{N}$ denotes $(w_1)\cdots(w_n)P$ for some arbitrary order of the (w_i). Without the statements that U is plain and $\tilde{w} \subseteq \mathrm{fn}((\bar{x}y|x(z).R)|U)$, this lemma is a simplification, by restricting attention to the syntax of aπ, of Lemma 1.2.20 in [49], established for the full π-calculus.

Lemma 4. *If $T \longmapsto T'$ with $T, T' \in \mathcal{P}_{\mathrm{a}\pi}$ then there are $\tilde{w} \subseteq \mathcal{N}$, $x, y, z \in \mathcal{N}$ and terms $R, U \in \mathcal{P}_{\mathrm{a}\pi}$ with U plain, such that $T \equiv (\tilde{w})((\bar{x}y|x(z).R)|U) \longmapsto (\tilde{w})((\mathbf{0}|R\{y/z\})|U) \equiv T'$ and $\tilde{w} \subseteq \mathrm{fn}((\bar{x}y|x(z).R)|U)$.*

Proof. The reduction $T \longmapsto T'$ is provable from the reduction rules of Definition 4. Since $\equiv$ is a congruence, applications of the last rule can always be commuted until they appear at the end of such a proof. Hence there are terms T^{pre} and T^{post} such that $T \equiv T^{\mathrm{pre}} \longmapsto T^{\mathrm{post}} \equiv T'$, and the reduction $T^{\mathrm{pre}} \longmapsto T^{\mathrm{post}}$ is generated by the first three rules of Definition 4. Applying rules (8), (9), (2) and $\overleftarrow{(7)}$ of structural congruence, the terms T^{pre} and T^{post} can be brought in the forms $(\tilde{w})P^{\rm pre}$ and $(\tilde{w})P^{\rm post}$, with $P^{\rm pre}$ and $P^{\rm post}$ plain, at the same time moving all applications of the reduction rule for restriction $(z)P$ after all applications of the rule for parallel composition. Applying rules $\overleftarrow{(1)},\overleftarrow{(3)}$, all applications of the reduction rule for parallel composition can be merged into a single application. After this proof normalisation, the reduction $T^{\rm pre} \longmapsto T^{\rm post}$ is generated by one application of the first reduction rule of Definition 4, followed by one application of the rule for $|$, followed by applications of the rule for restriction. Now $T^{\rm pre}$ has the form $(\tilde{w})((\bar{x}y|x(z).R)|U)$ and $T^{\rm post} = (\tilde{w})((\mathbf{0}|R\{y/z\})|U)$ with U plain.

Rules $\overleftarrow{(3)}$, (7), (5) and (3) of structural congruence, in combination with α-conversion (rules (8) and (9)), allow all names w with $w \notin {\rm fn}((\bar{x}y|x(z).R)|U)$ to be dropped from $\tilde{w}$, while preserving the validity of $T^{\rm pre} \longmapsto T^{\rm post}$. ☐

Write $P \equiv_{\rm S} Q$ if P can be converted into Q using applications of rules (1)–(3),(5)–(9) only, in either direction, possibly within a context, and $P \Rrightarrow_! Q$ if this can be done with rule (4), from left to right.

Lemma 5. *Lemma 4 can be strengthened by replacing $T \equiv (\tilde{w})((\bar{x}y|x(z).R)|U)$ by $T \Rrightarrow_! \equiv_{\rm S} (\tilde{w})((\bar{x}y|x(z).R)|U)$.*

Proof. Define rule $\overleftarrow{(4)}$ to *commute over* rule (1) if for each sequence $P \equiv Q \equiv R$ with $P \equiv Q$ an application of rule $\overleftarrow{(4)}$ and $Q \equiv R$ an application of rule (1), there exists a term Q' such that $P \equiv Q'$ holds by (possibly multiple) applications of rule (1) and $Q' \equiv R$ by applications of rule $\overleftarrow{(4)}$. As indicated in the table below, rule $\overleftarrow{(4)}$ commutes over all other rules of structural congruence, except for rule (4). The proof of this is trivial: in each case the two rules act on disjoint part of the syntax tree of Q. Moreover, rule $\overleftarrow{(4)}$ commutes over rule (4) too, except in the special case that the two applications annihilate each other precisely, meaning that $P = R$; this situation is indicated by the $\star$.

	(1)	$\overleftarrow{(1)}$	(2)	(3)	$\overleftarrow{(3)}$	(4)	$\overleftarrow{(4)}$	(5)	$\overleftarrow{(5)}$	(6)	(7)	$\overleftarrow{(7)}$	(8)	(9)
$\overleftarrow{(4)}$	✓	✓	✓	✓	✓	$\star$	·	✓	✓	✓	✓	✓	✓	✓

As a consequence of this, in a sequence $P_0 \equiv P_1 \equiv \ldots \equiv P_n$, all applications of rule $\overleftarrow{(4)}$ can be moved to the right. Moreover, when $P_{n-1} \equiv P_n := (\tilde{w})((\bar{x}y|x(z).R)|U) \longmapsto (\tilde{w})((\mathbf{0}|R\{y/z\})|U) \equiv T'$ and $P_{n-1} \equiv P_n$ is an application of rule $\overleftarrow{(4)}$, then this application must take place within the term R or U, and thus can be postponed until after the reduction step, so that $P_{n-1} = (\tilde{w})((\bar{x}y|x(z).R')|U') \longmapsto (\tilde{w})((\mathbf{0}|R'\{y/z\})|U') \equiv T'$ with U' plain. Thus, one may assume that in the sequence $T = P_0 \equiv P_1 \equiv \cdots \equiv P_n$ none of the steps is an application of rule $\overleftarrow{(4)}$.

Since applications of rule $\overleftarrow{(4)}$ could be shifted to the right in this sequence, all applications of rule (4) can be shifted to the left. Hence $T \Rrightarrow_! \equiv_{\rm S} (\tilde{w})((\bar{x}y|x(z).R)|U)$. ☐

Lemma 6. *If $[\![S]\!] \Rrightarrow_! T_0$ for $S \in \mathcal{P}_\pi$ and $T_0 \in \mathcal{P}_{\rm a\pi}$ then there is an $S_0 \in \mathcal{P}_\pi$ with $S \Rrightarrow_! S_0$ and $[\![S_0]\!] = T_0$.*

Proof. Similar to the proof of Lemma 2. ☐

Note that a variant of Lemma 6 with (2), (3), $\overleftarrow{(3)}$, or $\overleftarrow{(7)}$ in the rôle of (4) would not be valid.

Up to $\equiv_{\rm S}$ each term $P \in \mathcal{P}_\pi$ can be brought in the form $(\tilde{w})P$ with P plain and $\tilde{w} \subseteq {\rm fn}(P)$. Moreover, such a normal form has a degree of uniqueness:

Observation 1. If $(\tilde{w})P \equiv_{\rm S} (\tilde{v})Q$ with P,Q plain, $\tilde{w} \subseteq {\rm fn}(P)$ and $\tilde{v} \subseteq {\rm fn}(Q)$, then there is an injective renaming $\sigma{:}\mathcal{N} \to \mathcal{N}$ such that $\sigma(\tilde{v}) = \tilde{w}$ and $P \equiv_{\rm S} Q\sigma$. Thus, for each parallel component P' of P of the form $\bar{x}y.R$ or $x(z).R$ or $\surd$ or $!R$ there is a parallel component Q' of $Q\sigma$ with $P' \equiv_{\rm S} Q'$.

Below, $\equiv_{(8),(9)}$ denotes convertibility by applications of rules (8) and (9) only, and similarly for other rules.

Lemma 7. *If $[\![S]\!] \equiv_{\rm S} (\tilde{w})(U|\bar{x}u|x(r).R)$ with $S \in \mathcal{P}_\pi$, U plain and $\tilde{w} \subseteq {\rm fn}(U|\bar{x}u|x(r).R)$, then there are $V, R_1, R_2 \in \mathcal{P}_\pi$, $W \in \mathcal{P}_{\rm a\pi}$, $y, z, v_1, v_2 \in \mathcal{N}$ and $\tilde{s}, \tilde{t} \subseteq \mathcal{N}$ such that $S \equiv_{\rm S} (\tilde{s})(V \mid \bar{x}y.R_1 \mid x(z).R_2)$, $v_1 \neq y \neq u$, $\tilde{w} = \tilde{s} \uplus \tilde{t} \uplus \{u\}$, $U \equiv_{\rm S} W|u(v_1).(\bar{v}_1 y|[\![R_1]\!])$, $u, v_1 \notin {\rm fn}([\![R_1]\!])$, $[\![V]\!] \equiv_{\rm S} (\tilde{t})W$, $R \equiv_{\rm S} (v_2)(\bar{r}v_2|v_2(z).[\![R_2]\!])$, $r \neq v_2$ and $r, v_2 \notin {\rm fn}([\![R_2]\!]) \backslash \{z\}$.*

Proof. By applying rules (8), (9), (2) and $\overleftarrow{(7)}$ only, S can be brought into the form $S' := (\tilde{p})(P_1|\ldots|P_n)$ for some $n > 0$, where each P_i is of the form $\bar{s}y.R$ or $s(z).R$ or $\surd$ or $\mathbf{0}$ or $!R$. By means of $\overleftarrow{(3)}$, (7), (5) and (3) one can moreover assure that $\tilde{p} \subseteq {\rm fn}(P_1|\ldots|P_n)$. By the proof of Lemma 2 $[\![S]\!] \equiv_{\rm S} [\![S']\!]$. Furthermore, $[\![S']\!] = (\tilde{p})([\![P_1]\!]|\ldots|[\![P_n]\!])$.

By applying rules (8), (9), (2) and $\overleftarrow{(7)}$ only, the term $[\![P_1]\!]|\ldots|[\![P_n]\!]$ can be brought into the form $(\tilde{q})P$ with P plain; moreover, the set $\tilde{q}$ can be chosen disjoint from $\tilde{p}$.

Each $q \in \tilde{q}$ is a renaming of the name u in a term $[\![P_i]\!] = [\![\bar{x}y.Q]\!] = (u)(\bar{x}u|u(v).(\bar{v}y|[\![Q]\!]))$, so that $q \in {\rm fn}(P)$.

So $(\tilde{w})(U|\bar{x}u|x(r).R) \equiv_{\rm S} (\tilde{p})(\tilde{q})P$. Let σ be the renaming that exists by Observation 1, so that $\sigma(\tilde{p}) \uplus \sigma(\tilde{q}) = \tilde{w}$. Then $U|\bar{x}u|x(r).R \equiv_{\rm S} P\sigma$. So $\bar{x}u$ and $x(r).R$ (up to $\equiv_{\rm S}$) must be parallel components of $P\sigma$.

Let σ' be the restriction of σ to $\tilde{p}$ and take $\tilde{s} := \sigma'(\tilde{p})$. Let $S'' := (\tilde{s})(P_1\sigma'|\ldots|P_n\sigma')$. Then $S' \equiv_{\rm S} S''$ and $[\![S']\!] \equiv_{\rm S} [\![S'']\!] = (\tilde{s})([\![P_1\sigma']\!]|\ldots|[\![P_n\sigma']\!]) = (\tilde{s})([\![P_1]\!]\sigma'|\ldots|[\![P_n]\!]\sigma')$. Since $[\![P_1]\!]|\ldots|[\![P_n]\!]$ can be converted into $(\tilde{q})(P)$ by applications of rules (8), (9), (2) and $\overleftarrow{(7)}$, $[\![P_1]\!]\sigma'|\ldots|[\![P_n]\!]\sigma'$ can be converted into $(\tilde{q})(P\sigma')$ and even into $(\sigma(\tilde{q}))(P\sigma)$ by applications of these rules. One can apply (8),(9) first, so that each $[\![P_i]\!]\sigma'$ is converted into some term Q_i by applications of (8),(9), and $Q_1|\ldots|Q_n$ is converted into $(\sigma(\tilde{q}))(P\sigma)$ by applications of (2) and $\overleftarrow{(7)}$ only.

One of the $P_i\sigma'$ must be of the form $\bar{x}y.R_1$, so that $[\![P_i]\!]\sigma' = [\![P_i\sigma']\!] = [\![\bar{x}y.R_1]\!] = (u')(\bar{x}u'|u'(v_1).(\bar{v}_1 y|[\![R_1]\!]))$ with $u', v_1 \notin {\rm fn}(R_1) \cup \{x,y\}$, while u' is renamed into u in $Q_i = (u)(\bar{x}u|u(v_1).(\bar{v}_1 y|[\![R_1]\!]))$. For this is the only way $\bar{x}u$ can end up as a parallel component of $P\sigma$. It follows that $u, v_1 \notin {\rm fn}(R_1)$ and $v_1 \neq y \neq u \in \sigma(\tilde{q})$. Let $\tilde{t} := \sigma(\tilde{q}) \setminus u$. One obtains $\tilde{w} = \tilde{s} \uplus \tilde{t} \uplus \{u\}$.

Searching for an explanation of the parallel component $x(r).R$ (up to $\equiv_{\rm S}$) of $P\sigma$, the existence of the component $\bar{x}u$ of $P\sigma$ excludes the possibility that one of the $P_i\sigma'$ is of the form $\bar{t}'y'.R_2$ so that $[\![P_i]\!]\sigma' = (u')(\bar{t}'u'|u'(r').(\bar{r}'y'|[\![R_2]\!]))$, while u' is renamed into x and r' into r in the expression $Q_i = (x)(\bar{t}'x|x(r).(\bar{r}y'|[\![R_2]\!]))$.

Hence one of the $P_i\sigma'$ is of the form $x(z).R_2$, so that $[\![P_i]\!]\sigma' = [\![P_i\sigma']\!] = [\![x(z).R_2]\!] = x(r').(v')(\bar{r}'v'|v'(z).[\![R_2]\!])$ with $r' \neq v'$ and $r', v' \notin \mathrm{fn}(R_2)\setminus\{z\}$, while r', v' and z are renamed into r, v_2 and z' in $Q_i = x(r).(v_2)(\bar{r}v_2|v_2(z').R_2')$, where $(z')R_2' \equiv_{(8),(9)} (z)[\![R_2]\!]$. Thus $r, v_2 \notin \mathrm{fn}(R_2)\setminus\{z\}$ and $r{\neq}v_2$. Further, $x(z).R \equiv_{\mathrm{S}} Q_i$, so $R \equiv_{\mathrm{S}} (v_2)(\bar{r}v_2|v_2(z).[\![R_2]\!])$.

Let V collect all parallel components $P_i\sigma'$ other than the above discussed components $\bar{x}y.R_1$ and $x(z).R_2$. Then $S \equiv_{\mathrm{S}} S'' \equiv_{\mathrm{S}} (\tilde{s})(V \mid \bar{x}y.R_1 \mid x(z).R_2)$.

One has $(\tilde{w})(U|\bar{x}u|x(r).R) \equiv_{\mathrm{S}} [\![S]\!] \equiv_{\mathrm{S}} [\![S'']\!] \equiv_{\mathrm{S}}$
$[\![(\tilde{s})(V \mid \bar{x}y.R_1 \mid x(z).R_2)]\!] =$
$(\tilde{s})([\![V]\!] \mid (u')(\bar{x}u'|u'(v_1).(\bar{v_1}y|[\![R_1]\!])) \mid x(r')(v')\cdots) \equiv_{(8),(9)}$
$(\tilde{s})(T \mid (u)(\bar{x}u|u(v_1).(\bar{v_1}y|[\![R_1]\!])) \mid x(r).R) \equiv_{(1),(2),(7)}$
$(\tilde{s})(u)(T \mid u(v_1).(\bar{v_1}y|[\![R_1]\!]) \mid \bar{x}u \mid x(r).R) \equiv_{(2),(7)}$
$(\tilde{s})(u)(\tilde{t})(W \mid u(v_1).(\bar{v_1}y|[\![R_1]\!]) \mid \bar{x}u \mid x(r).R)$.
Here T is the parallel composition of all components Q_i obtained by renaming of the parallel components $P_i\sigma'$ of $[\![V]\!]$, and $(\tilde{t})W$ with W plain is obtained from T by rules (2),(7). So $U \mid \bar{x}u \mid x(r).R \equiv_{\mathrm{S}} W \mid u(v_1).(\bar{v_1}y|[\![R_1]\!]) \mid \bar{x}u \mid x(r).R$ by Observation 1. It follows that $U \equiv_{\mathrm{S}} W \mid u(v_1).(\bar{v_1}y|[\![R_1]\!])$. □

A straightforward case distinction shows that the set of names occurring free in a term is invariant under structural congruence:

Observation 2. If $P \equiv Q$ then $\mathrm{fn}(P) = \mathrm{fn}(Q)$.

The above results can be combined to establish the special case of operational soundness where the sequence of reductions $[\![S]\!] \Longmapsto T$ consists of one reduction step only.

Lemma 8. *Let $S \in \mathcal{P}_\pi$ and $T \in \mathcal{P}_{\mathrm{a}\pi}$. If $[\![S]\!] \longmapsto T$ then there is a S' with $S \longmapsto S'$ and $T \Longmapsto [\![S']\!]$.*

Proof. Suppose $[\![S]\!] \longmapsto T$. Then, by Lemma 5, there are $\tilde{w}{\subseteq}\mathcal{N}$, $x, u, r{\in}\mathcal{N}$ and $T_0, R, U{\in}\mathcal{P}_{\mathrm{a}\pi}$ with U plain, such that $[\![S]\!] \Rrightarrow_! T_0 \equiv_{\mathrm{S}} (\tilde{w})(U|\bar{x}u|x(r).R) \longmapsto (\tilde{w})(U|R\{u/r\}) \equiv T$ and $\tilde{w} \subseteq \mathrm{fn}(U|\bar{x}u|x(r).R)$. By Lemma 6, there is an $S_0 \in \mathcal{P}_\pi$ with $S \Rrightarrow_! S_0$ and $[\![S_0]\!] = T_0$. So, by Lemma 7, there are $V, R_1, R_2 \in \mathcal{P}_\pi$, $W \in \mathcal{P}_{\mathrm{a}\pi}$, $y, z, v_1, v_2 \in \mathcal{N}$ and $\tilde{s}, \tilde{t} \subseteq \mathcal{N}$ such that $S \equiv_{\mathrm{S}} (\tilde{s})(V \mid \bar{x}y.R_1 \mid x(z).R_2)$, $v_1 \neq y \neq u$, $\tilde{w} = \tilde{s} \uplus \tilde{t} \uplus \{u\}$, $U \equiv_{\mathrm{S}} W|u(v_1).(\bar{v_1}y|[\![R_1]\!])$, $u, v_1 \notin \mathrm{fn}([\![R_1]\!])$, $[\![V]\!] \equiv_{\mathrm{S}} (\tilde{t})W$, $R \equiv_{\mathrm{S}} (v_2)(\bar{r}v_2|v_2(z).[\![R_2]\!])$, $r \neq v_2$ and $r, v_2 \notin \mathrm{fn}([\![R_2]\!])\setminus\{z\}$.

As $(\mathrm{fn}(V) \cup \{x, y\} \cup \mathrm{fn}(R_1) \cup (\mathrm{fn}(R_2)\setminus\{z\}))\setminus\tilde{s} = \mathrm{fn}(S_0)$, using Observation 2, and $\tilde{w} \cap \mathrm{fn}(S_0) = \tilde{w} \cap \mathrm{fn}(T_0) = \emptyset$, using Lemma 1, $t, u \notin \mathrm{fn}(V) \cup \{x, y\} \cup \mathrm{fn}(R_1) \cup (\mathrm{fn}(R_2)\setminus\{z\})$ for all $t{\in}\tilde{t}$. Let $v{\in}\mathcal{N}$ satisfy $u, r, y{\neq}v \notin \mathrm{fn}(R_1) \cup (\mathrm{fn}(R_2)\setminus\{z\})$.

Take $S' := (\tilde{s})(V \mid R_1 \mid R_2\{y/z\})$. Then $S \longmapsto S'$ and
$T \equiv (\tilde{w})(U \mid R\{u/r\})$
$\equiv (\tilde{w})\big(W|u(v_1).(\bar{v_1}y|[\![R_1]\!]) \mid (v_2)(\bar{r}v_2|v_2(z).[\![R_2]\!])\{u/r\}\big)$
$\equiv (\tilde{w})\big(W|u(v).(\bar{v}y|[\![R_1]\!]) \mid (v)(\bar{r}v|v(z).[\![R_2]\!])\{u/r\}\big)$
(*as $y \neq v_1$, $v_1 \notin \mathrm{fn}([\![R_1]\!])$, $r \neq v_2$ and $v_2 \notin \mathrm{fn}([\![R_2]\!])\setminus\{z\}$*)
$\equiv (\tilde{w})\big(W|u(v).(\bar{v}y|[\![R_1]\!]) \mid (v)(\bar{u}v|v(z).[\![R_2]\!])\big)$
(*since $r \neq v \neq u$ and $r \notin \mathrm{fn}([\![R_2]\!])\setminus\{z\}$*)
$\equiv (\tilde{s})(u)(\tilde{t})\big(W \mid u(v).(\bar{v}y|[\![R_1]\!]) \mid (v)(\bar{u}v|v(z).[\![R_2]\!])\big)$
$\equiv (\tilde{s})(u)\big((\tilde{t})W \mid u(v).(\bar{v}y|[\![R_1]\!]) \mid (v)(\bar{u}v|v(z).[\![R_2]\!])\big)$
(*since $t \notin \{u, y\} \cup \mathrm{fn}([\![R_1]\!]) \cup (\mathrm{fn}([\![R_2]\!]) \setminus \{z\})$ for $t \in \tilde{t}$*)
$\equiv (\tilde{s})\,\big([\![V]\!] \mid (u)(u(v).(\bar{v}y|[\![R_1]\!]) \mid (v)(\bar{u}v|v(z).[\![R_2]\!]))\big)$
(*since $u \notin \mathrm{fn}([\![V]\!])$*)
$\longmapsto (\tilde{s})\big([\![V]\!] \mid (v).((\bar{v}y|[\![R_1]\!]) \mid v(z).[\![R_2]\!])\big)$
(*since $u \neq v$ and $u \notin \{y\} \cup \mathrm{fn}([\![R_1]\!]) \cup \mathrm{fn}([\![R_2]\!])\setminus\{z\}$*)
$\longmapsto (\tilde{s})\big([\![V]\!] \mid [\![R_1]\!] \mid [\![R_2]\!]\{y/z\}\big)$
(*since $v \notin \{y\} \cup \mathrm{fn}([\![R_1]\!]) \cup \mathrm{fn}([\![R_2]\!])\setminus\{z\}$*)
$= (\tilde{s})\big([\![V]\!] \mid [\![R_1]\!] \mid [\![R_2\{y/z\}]\!]\big)$ (*by Lemma 1*)
$= [\![S']\!]$. □

To obtain general operational soundness, I introduce an *inert* reduction relation with a confluence property, stated in Lemma 9 below: any other reduction that can occur as an alternative to an inert reduction is still possible after the occurrence of the inert reduction.

Definition 5. Let $\Rrightarrow$ be the smallest relation on $\mathcal{P}_{\mathrm{a}\pi}$ such that

1. $(v)(\bar{v}y|P|v(z).Q) \Rrightarrow P|(Q\{y/z\})$,
2. if $P \Rrightarrow Q$ then $P|R \Rrightarrow Q|R$,
3. if $P \Rrightarrow Q$ then $(w)P \Rrightarrow (w)Q$,
4. if $P \equiv P' \Rrightarrow Q' \equiv Q$ then $P \Rrightarrow Q$,

where $v \notin \mathrm{fn}(P) \cup \mathrm{fn}(Q\{y/z\})$.

First of all observe that whenever two processes are related by $\Rrightarrow$, an actual reduction takes place.

Observation 3. If $P \Rrightarrow Q$ then $P \longmapsto Q$.

As its proof shows, the conclusion of Lemma 3 can be restated as $[\![S]\!] \longmapsto\Rrightarrow\Rrightarrow [\![S']\!]$. Likewise, the two occurrences of $\longmapsto$ at the end of the proof of Lemma 8 can be replaced by $\Rrightarrow$:

Observation 4. In the conclusion of Lemma 8, $T \Longmapsto [\![S']\!]$ can be restated as $T \Rrightarrow\Rrightarrow [\![S']\!]$.

Lemma 9. *If $P \Rrightarrow Q$ and $P \longmapsto P'$ with $P' \not\equiv Q$ then there is a Q' with $Q \longmapsto Q'$ and $P' \Rrightarrow Q'$.*

Proof. By Lemma 4 there are $\tilde{w} \subseteq \mathcal{N}$, $x, y, z \in \mathcal{N}$ and $R, U \in \mathcal{P}_{\mathrm{a}\pi}$ such that U plain, $\tilde{w} \subseteq \mathrm{fn}((\bar{x}y|x(z).R)|U)$ and $P \equiv P_0 := (\tilde{w})((\bar{x}y|x(z).R)|U) \longmapsto (\tilde{w})((\mathbf{0}|R\{y/z\})|U) \equiv P'$. Likewise, there are $\tilde{q} \subseteq \mathcal{N}$, $x', y', z' \in \mathcal{N}$ and $R', U' \in \mathcal{P}_{\mathrm{a}\pi}$ such that U' plain, $\tilde{q} \subseteq \mathrm{fn}((\bar{x}'y'|x'(z').R')|U')$ and $P \equiv P_1 := (\tilde{q})((\bar{x}'y'|x'(z').R')|U') \longmapsto (\tilde{q})((\mathbf{0}|R'\{y'/z'\})|U') \equiv Q$. So $(\tilde{w})((\mathbf{0}|R\{y/z\})|U) \longleftarrow P_0 \equiv P_1 \longmapsto (\tilde{q})(R'\{y'/z'\}|U')$. As in the proof of Lemma 5, all applications of rule (4) in the sequence of reductions $P_0 \equiv P_1$ can be moved to the right and shifted over the $\longmapsto$. Likewise, all applications of rule (4) can be moved to the left and shifted over the $\longleftarrow$. Therefore, I may assume that $P_0 \equiv_{\mathrm{S}} P_1$. Let σ be the injective renaming that exists by Observation 1. Then $(\bar{x}y|x(z).R)|U \equiv_{\mathrm{S}} ((\bar{x}'y'|x'(z').R')|U')\sigma$. Let $u := \sigma(x')$, $v := \sigma(y')$, $r := \sigma(z')$, $R'' := R\sigma$ and $U'' := U\sigma$. Then $(\bar{x}y|x(z).R)|U \equiv_{\mathrm{S}} (\bar{u}v|u(r).R'')|U''$.

In a reduction step of the form $P \Longrightarrow Q$, the reacting prefixes $\bar{a}b$ and $a(c).V$ are always found in the scope of a restriction operator (a), and without a ! between (a) and $\bar{a}b$ or $a(c).V$, such that in this scope there are no other unguarded occurrences of prefixes $\bar{a}d$ or $a(e).W$. This follows by a trivial induction on the definition of $\Longrightarrow$. In particular, this property is preserved when applying structural congruence to P. Consequently, the plain term U'' has no parallel components of the form $\bar{u}y''$ or $u(z'').R'''$.

Case 1: $x = u$. Then, by the above, $(z)R \equiv_{\rm S} (r)R''$, $y = v$ and $U \equiv_{\rm S} U''$. Consequently, $P' \equiv Q$.

Case 2: $x \neq u$. Then $\bar{u}v$ and $u(r).R''$ (up to $\equiv_{\rm S}$) must be parallel components of U, and $\bar{x}y$ and $x(z).R$ (up to $\equiv_{\rm S}$) must be parallel components of U'', so that $P \equiv P_0 = (\tilde{w})((\bar{x}y|x(z).R)|(\bar{u}v|u(r).R'')|V)$, where $U \equiv_{\rm S} (\bar{u}v|u(r).R'')|V$ and $U'' \equiv_{\rm S} (\bar{x}y|x(z).R)|V$. This shows that the reductions $P \Longrightarrow Q$ and $P \longmapsto P'$ are concurrent, so that there is a Q' with $Q \longmapsto Q'$ and $P' \Longrightarrow Q'$. □

Corollary 1. *If $P \Longrightarrow Q$ and $P \Longmapsto P'$ then either $Q \Longmapsto P'$ or there is a Q' with $Q \Longmapsto Q'$ and $P' \Longrightarrow Q'$. Moreover, the sequence $Q \Longmapsto P'$ or $Q \Longmapsto Q'$ contains no more reduction steps than the sequence $P \Longmapsto Q'$.*

Proof. By repeated application of Lemma 9. □

Corollary 2. *If $P \Longrightarrow^* Q$ and $P \Longmapsto P'$ then there is a Q' with $Q \Longmapsto Q'$ and $P' \Longrightarrow^* Q'$. Moreover, the sequence $Q \Longmapsto Q'$ contains no more reduction steps than the sequence $P \Longmapsto Q'$.*

By combining Corollary 2 with Observations 3 and 4 one finds that the criterion of operational soundness is met.

Theorem 1. *Let $S \in \mathcal{P}_\pi$ and $T \in \mathcal{P}_{\rm a\pi}$. If $[\![S]\!] \Longmapsto T$ then there is a S' with $S \Longmapsto S'$ and $T \Longmapsto [\![S']\!]$.*

Proof. By induction on the length n of the sequence $[\![S]\!] \Longmapsto T$. The base case $n = 0$ is trivial: take $S' := S$. So let $[\![S]\!] \longmapsto T_1 \Longmapsto T$, where $T_1 \Longmapsto T$ has length n. By Lemma 8 with Observation 4 $\exists S_1$ with $S \longmapsto S_1$ and $T_1 \Longrightarrow^* [\![S_1]\!]$. By Corollary 2 $\exists T'$ with $[\![S_1]\!] \Longmapsto T'$ and $T \Longrightarrow^* T'$. Furthermore, the sequence $[\![S_1]\!] \Longmapsto T'$ has length $\leq n$. By induction, there is a S' with $S_1 \Longmapsto S'$ and $T' \Longmapsto [\![S']\!]$. Hence $S \Longmapsto S'$ and $T \Longmapsto [\![S']\!]$, using Observation 3. □

4.4. Divergence reflection

Corollary 3. *If $P \Longrightarrow Q$ and $P \longmapsto^\omega$ then $Q \longmapsto^\omega$.*

Proof. By repeated application of Lemma 9. □

Together with Observation 4 this implies that the criterion of divergence reflection is met.

Theorem 2. *Let $S \in \mathcal{P}_\pi$. If $[\![S]\!] \longmapsto^\omega$ then $S \longmapsto^\omega$.*

Proof. Suppose $[\![S]\!] \longmapsto^\omega$. Then $[\![S]\!] \longmapsto T_1 \longmapsto^\omega$. By Lemma 8 with Observation 4 there is an S_1 with $S \longmapsto S_1$ and $T_1 \Longrightarrow^* [\![S_1]\!]$. By Corollary 3 $[\![S_1]\!] \longmapsto^\omega$. In the same way there is an S_2 with $S_1 \longmapsto S_2$ and $[\![S_2]\!] \longmapsto^\omega$, and so on. Thus $S \longmapsto^\omega$. □

4.5. Success sensitiveness

The success predicate $\downarrow$ can also be defined inductively:

$$\surd\!\downarrow \qquad \frac{P\!\downarrow}{(P|Q)\!\downarrow} \qquad \frac{Q\!\downarrow}{(P|Q)\!\downarrow} \qquad \frac{P\!\downarrow}{((z)P)\!\downarrow} \qquad \frac{P\!\downarrow}{(!P)\!\downarrow}$$

Note that if $P \equiv Q$ and $P\!\downarrow$ then also $Q\!\downarrow$.

Lemma 10. *Let $S \in \mathcal{P}_\pi$. Then $[\![S]\!]\!\downarrow$ iff $S\!\downarrow$.*

Proof. A trivial structural induction. □

Lemma 11. *If $T \longmapsto T'$ and $T\!\downarrow$ then $T'\!\downarrow$.*

Proof. By Lemma 4 there are $\tilde{w} \subseteq \mathcal{N}$, $x, u, r \in \mathcal{N}$ and $R, U \in \mathcal{P}_{\rm a\pi}$ with U plain, such that $\tilde{w} \subseteq {\rm fn}((\bar{x}u|x(r).R)|U)$ and $T \equiv (\tilde{w})((\bar{x}u|x(r).R)|U) \longmapsto (\tilde{w})((\mathbf{0}|R\{u/r\})|U) \equiv T'$. Since $T\!\downarrow$, it must be that $U\!\downarrow$ and hence $T'\!\downarrow$. □

By combining Lemmata 10 and 11 with Lemma 3 and Theorem 1 one finds that also the criterion of success sensitiveness is met.

Theorem 3. *Let $S \in \mathcal{P}_\pi$. Then $S\!\Downarrow$ iff $[\![S]\!]\!\Downarrow$.*

Proof. Suppose that $S\!\Downarrow$. Then $S \Longmapsto S'$ for a process S' with $S'\!\downarrow$. By Lemma 3 $[\![S]\!] \Longmapsto [\![S']\!]$. By Lemma 10 $[\![S']\!]\!\downarrow$. Hence $[\![S]\!]\!\Downarrow$.

Now suppose $[\![S]\!]\!\Downarrow$. Then $[\![S]\!] \Longmapsto T$ for a process T with $T\!\downarrow$. By Theorem 1 there is a S' with $S \Longmapsto S'$ and $T \Longmapsto [\![S']\!]$. By Lemma 11 $[\![S']\!]\!\downarrow$. By Lemma 10 $S'\!\downarrow$. Hence $S\!\Downarrow$. □

5. Validity of Honda & Tokoro's encoding

That the encoding of Honda & Tokoro also satisfies all five criteria of Gorla follows in the same way. I will only show the steps where a difference with the previous sections occurs. In this section $[\![\,\cdot\,]\!]$ stands for $[\![\,\cdot\,]\!]_{\rm HT}$.

Lemma 12. *Let $S, S' \in \mathcal{P}_\pi$. If $S \longmapsto S'$ then $[\![S]\!] \Longmapsto [\![S']\!]$.*

Proof. By induction on the derivation of $S \longmapsto S'$.

- Let $S = \bar{x}y.P|x(z).Q$, $y \notin {\rm bn}(Q)$ and $S' = P|Q\{y/z\}$. Pick $u \notin {\rm fn}(P) \cup {\rm fn}(Q) \cup \{x, y\}$. Then

  $$\begin{array}{rcl} [\![S]\!] & = & x(u).(\bar{u}y|[\![P]\!]) \mid (u)(\bar{x}u|u(z).[\![Q]\!]) \\ & \longmapsto & (u)\big(\bar{u}y|[\![P]\!] \mid u(z).[\![Q]\!]\big) \\ & \longmapsto & [\![P]\!] \mid ([\![Q]\!]\{y/z\}) \\ & = & [\![P]\!] \mid [\![Q\{y/z\}]\!] \qquad \text{(using Lemma 1)} \\ & = & [\![P \mid Q\{y/z\}]\!] = [\![S']\!]. \end{array}$$

  Here structural congruence is applied in omitting parallel components $\mathbf{0}$ and the empty binders (u).

- The other three cases proceed as in the proof of Lemma 3. □

Lemma 13. *If* $[\![S]\!] \equiv_S (\tilde{w})(U|\bar{x}u|x(r).R)$ *with* $S \in \mathcal{P}_\pi$, *U plain and* $\tilde{w} \subseteq \text{fn}(U|\bar{x}u|x(r).R)$, *then there are terms* $V, R_1, R_2 \in \mathcal{P}_\pi$, $W \in \mathcal{P}_{a\pi}$, *and names* $y, z \in \mathcal{N}$ *and* $\tilde{s}, \tilde{t} \subseteq \mathcal{N}$ *such that* $S \equiv_S (\tilde{s})(V \mid \bar{x}y.R_1 \mid x(z).R_2)$, $\tilde{w} = \tilde{s} \uplus \tilde{t} \uplus \{u\}$, $U \equiv_S W|u(z).[\![R_2]\!]$, $u \notin \text{fn}([\![R_2]\!]) \backslash \{z\}$, $[\![V]\!] \equiv_S (\tilde{t})W$, $r \neq y$, $R \equiv_S \bar{r}y|[\![R_1]\!]$ *and* $r \notin \text{fn}([\![R_1]\!])$.

Proof. The first two paragraphs proceed exactly as in the proof of Lemma 7.

Each $q \in \tilde{q}$ is a renaming of the name u in a term $[\![P_i]\!] = [\![x(z).Q]\!] = (u)(\bar{x}u|u(z).[\![Q]\!])$, so that $q \in \text{fn}(P)$.

The next two paragraphs proceed exactly as in the proof of Lemma 7.

One of the $P_i\sigma'$ must be of the form $x(z).R_2$, so that $[\![P_i]\!]\sigma' = [\![P_i\sigma']\!] = [\![x(z).R_2]\!] = (u')(\bar{x}u'|u'(z).[\![R_2]\!])$ with $u' \notin \text{fn}(R_2)\backslash\{z\} \cup \{x\}$, while u' and z are renamed into u and z' in $Q_i = (u)(\bar{x}u|u(z').R_2')$, where $(z')R_2' \equiv_{(8),(9)} (z)[\![R_2]\!]$. For this is the only way $\bar{x}u$ can end up as a parallel component of $P\sigma$. It follows that $u \notin \text{fn}(R_2)\backslash\{z\}$ and $u \in \sigma(\tilde{q})$. Let $\tilde{t} := \sigma(\tilde{q})\backslash u$. One obtains $\tilde{w} = \tilde{s} \uplus \tilde{t} \uplus \{u\}$.

Searching for an explanation of the parallel component $x(r).R$ (up to $\equiv_S$) of $P\sigma$, the existence of the component $\bar{x}u$ of $P\sigma$ excludes the possibility that one of the $P_i\sigma'$ is of the form $t'(r').R_1$ so that $[\![P_i]\!]\sigma' = (u')(\bar{t}'u'|u'(r').[\![R_1]\!])$, while u' is renamed into x and r' into r in the expression $Q_i = (u')(\bar{t}'x|x(r).R_1')$.

Hence one of the $P_i\sigma'$ is of the form $\bar{x}y.R_1$, so that $[\![P_i]\!]\sigma' = [\![P_i\sigma']\!] = [\![\bar{x}y.R_1]\!] = x(r').(\bar{r}'y|[\![R_1]\!])$ with $r' \notin \text{fn}(R_1) \cup \{x, y\}$, while r' is renamed into r in $Q_i = x(r).(\bar{r}y|[\![R_1]\!])$. Thus $r \notin \text{fn}(R_1)$ and $r \neq y$. Further, $R \equiv_S (\bar{r}y|[\![R_1]\!])$.

Let V collect all parallel components $P_i\sigma'$ other than the above discussed components $\bar{x}y.R_1$ and $x(z).R_2$. Then $S \equiv_S S'' \equiv_S (\tilde{s})(V \mid \bar{x}y.R_1 \mid x(z).R_2)$.

One has $(\tilde{w})(U|\bar{x}u|x(r).R) \equiv_S [\![S]\!] \equiv_S [\![S'']\!] \equiv_S$
$[\![(\tilde{s})(V \mid \bar{x}y.R_1 \mid x(z).R_2)]\!] =$
$(\tilde{s})([\![V]\!] \mid x(r').(\bar{r}'y|[\![R_1]\!]) \mid (u')(\bar{x}u'|u'(z).[\![R_2]\!])) \equiv_{(8),(9)}$
$(\tilde{s})(T \mid x(r).R \mid (u)(\bar{x}u|u(z).[\![R_2]\!])) \equiv_{(1),(2),(7)}$
$(\tilde{s})(u)(T \mid \bar{x}u|u(z).[\![R_2]\!] \mid x(r).R) \equiv_{(2),(7)}$
$(\tilde{s})(u)(\tilde{t})(W \mid u(z).[\![R_2]\!] \mid \bar{x}u \mid x(r).R)$.
Here T is the parallel composition of all components Q_i obtained by renaming of the parallel components $P_i\sigma'$ of $[\![V]\!]$, and $(\tilde{t})W$ with W plain is obtained from T by rules (2),(7). So $U \mid \bar{x}u \mid x(r).R \equiv_S W \mid u(z).[\![R_2]\!] \mid \bar{x}u \mid x(r).R$ by Observation 1. It follows that $U \equiv_S W \mid u(z).[\![R_2]\!]$. ☐

Lemma 14. *Let* $S \in \mathcal{P}_\pi$ *and* $T \in \mathcal{P}_{a\pi}$. *If* $[\![S]\!] \longmapsto T$ *then there is a* S' *with* $S \longmapsto S'$ *and* $T \Longmapsto [\![S']\!]$.

Proof. The first paragraph proceeds as in the proof of Lemma 8, but incorporating the conclusion of Lemma 13 instead of Lemma 7. Again, one finds, for all $t \in \tilde{t}$, that $t, u \notin \text{fn}(V) \cup \{x, y\} \cup \text{fn}(R_1) \cup (\text{fn}(R_2) \setminus \{z\})$.

Take $S' := (\tilde{s})(V \mid R_1 \mid R_2\{y/z\})$. Then $S \longmapsto S'$ and
$T \equiv (\tilde{w})(U \mid R\{u/r\})$
$\equiv (\tilde{w})\big(W|u(z).[\![R_2]\!] \mid \bar{r}y|[\![R_1]\!]\{u/r\}\big)$
$\equiv (\tilde{w})\big(W|u(z).[\![R_2]\!] \mid \bar{u}y|[\![R_1]\!]\big)$
(*since* $r \neq y$ *and* $r \notin \text{fn}([\![R_1]\!])$)
$\equiv (\tilde{s})(u)(\tilde{t})\big(W \mid u(z).[\![R_2]\!] \mid \bar{u}y|[\![R_1]\!]\big)$
$\equiv (\tilde{s})(u)\big((\tilde{t})W \mid \bar{u}y|[\![R_1]\!] \mid u(z).[\![R_2]\!]\big)$
(*since* $t \notin \{u, y\} \cup \text{fn}([\![R_1]\!]) \cup (\text{fn}([\![R_2]\!]) \setminus \{z\})$ *for* $t \in \tilde{t}$)
$\equiv (\tilde{s})\big([\![V]\!] \mid (u)(\bar{u}y|[\![R_1]\!] \mid u(z).[\![R_2]\!])\big)$ (*as* $u \notin \text{fn}([\![V]\!])$)
$\longmapsto (\tilde{s})\big([\![V]\!] \mid [\![R_1]\!] \mid [\![R_2]\!]\{y/z\}\big)$
(*since* $u \notin \{y\} \cup \text{fn}([\![R_1]\!]) \cup \text{fn}([\![R_2]\!]) \backslash \{z\}$)
$= (\tilde{s})\big([\![V]\!] \mid [\![R_1]\!] \mid [\![R_2\{y/z\}]\!]\big)$ (*by Lemma 1*)
$= [\![S']\!]$. ☐

As its proof shows, the conclusion of Lemma 12 can be restated as $[\![S]\!] \longmapsto\Longrightarrow [\![S']\!]$. The occurrence of $\longmapsto$ at the end of the proof of Lemma 14 can be replaced likewise:

Observation 5. In the conclusion of Lemma 8, $T \Longmapsto [\![S']\!]$ can be restated as $T \Longrightarrow [\![S']\!]$.

6. Conclusion

This paper proved the validity according to Gorla of the encodings proposed by Boudol and by Honda & Tokoro of the synchronous choice-free π-calculus into its asynchronous fragment; that is, both encodings enjoy the five correctness criteria of [23]. For such a result, easily believed to be "obvious", the proofs are surprisingly complicated,[2] and involve the new concept of an inert reduction. Yet, I conjecture that it is not possible to simplify the proofs in any meaningful way.

Below I reflect on three of Gorla's criteria in the light of the lessons learned from this case study.

Compositionality. Compositionality demands that for every k-ary operator $\texttt{op}$ of the source language there is a k-ary context $C^N_{\texttt{op}}[\_1; \ldots; \_k]$ in the target such that

$$[\![\texttt{op}(S_1, \ldots, S_k)]\!] = C^N_{\texttt{op}}([\![S_1]\!]; \ldots; [\![S_k]\!])$$

for all $S_1, \ldots, S_k \in \mathcal{P}_1$. A drawback of this criterion is that this context may depend on the set of names N that occur free in the arguments $S_1, \ldots, S_k$. The present application shows that we cannot simply strengthen the criterion of compositionality by dropping the dependence on N. For then the present encodings would fail to be compositional. However, in [21] a form of compositionality is proposed where $C_{\texttt{op}}$ does not depend on N, but the main requirement is weakened to

$$[\![\texttt{op}(S_1, \ldots, S_k)]\!] \stackrel{\alpha}{=} C_{\texttt{op}}([\![S_1]\!]; \ldots; [\![S_k]\!]).$$

Here $\stackrel{\alpha}{=}$ denotes equivalence up to *α-conversion*, renaming of bound names and variables, here corresponding with rules (8) and (9) of structural congruence. This suffices to rescue the current encodings. It is an open question whether there are examples of intuitively valid encodings that essentially need the dependence of N allowed by [23], i.e., where $C^{N_1}_{\texttt{op}}$ and $C^{N_2}_{\texttt{op}}$ differ by more than α-conversion.

[2] The complications lay chiefly with proving operational soundness; for some of the other criteria the proofs are straightforward.

Another method of dealing with the fresh names u and v that are used in the present encodings, also proposed in [21], is to equip the target language with two fresh names that do not occur in the set of names available for the source language. Making the dependence on the choice of set $\mathcal{N}$ of names explicit, this method calls π expressible into aπ if for each $\mathcal{N}$ there exists an $\mathcal{N}'$ such that there is a valid encoding of $\pi(\mathcal{N})$ into a$\pi(\mathcal{N}')$.

Operational soundness. Operational soundness stems from Nestmann & Pierce [35], who proposed two forms of it:
($\mathfrak{I}$) if $[\![S]\!] \longmapsto_2 T$ then $\exists S'\colon S \longmapsto_1 S'$ and $T \asymp_2 [\![S']\!]$.
($\mathfrak{S}$) if $[\![S]\!] \Longmapsto_2 T$ then $\exists S'\colon S \Longmapsto_1 S'$ and $T \Longmapsto_2 [\![S']\!]$.
The version of Gorla is the common weakening of these:
($\mathfrak{G}$) if $[\![S]\!] \Longmapsto_2 T$ then $\exists S'\colon S \Longmapsto_1 S'$ and $T \Longmapsto_2 \asymp_2 [\![S']\!]$.
An interesting intermediate form is
($\mathfrak{W}$) if $[\![S]\!] \Longmapsto_2 T$ then $\exists S'\colon S \Longmapsto_1 S'$ and $T \asymp_2 [\![S']\!]$.
Nestmann & Pierce observed that "nonprompt encodings", that "allow administrative (or book-keeping) steps to precede a committing step", "do not satisfy ($\mathfrak{I}$)". For such encodings, which include the ones studied here, they proposed ($\mathfrak{S}$). As I have shown, the encodings of Boudol and of Honda & Tokoro, satisfy not only ($\mathfrak{G}$) but even ($\mathfrak{S}$). It remains an interesting open question whether they satisfy ($\mathfrak{W}$). Clearly, they do not when taking $\asymp_2$ to be the identity relation—as I did here—or structural congruence. However, it is conceivable that ($\mathfrak{W}$) holds for another reasonable choice of $\asymp_2$. (An unreasonable choice, such as the universal relation, tells us nothing.)

Success sensitivity. My treatment of success sensitivity differs slightly from the one of Gorla [23]. Gorla requires $\surd$ to be a constant of any two languages whose expressiveness is compared. Strictly speaking, this does not allow his framework to be applied to the encodings of Boudol or Honda & Tokoro, as these deal with languages not featuring $\surd$. Here I simply allowed $\surd$ to be added, which is in line with the way Gorla's framework has been used [22, 28, 44, 41, 43, 16, 17, 18, 19]. A consequence of this decision is that I have to specify how $\surd$ is translated—see the last sentence of Definition 2—as the addition of $\surd$ to both languages happens after a translation is proposed. This differs from [23], where it is explicitly allowed to take $[\![\surd]\!] \neq \surd$.

Gorla's success predicate is one of the possible ways to provide source and target languages with a set of *barbs* Ω, each being a unary predicate on processes. For $\omega \in \Omega$, write $P{\downarrow_\omega}$ if process P has the barb ω, and $P{\Downarrow_\omega}$ if $P \Longmapsto P'$ for a process P' with $P'{\downarrow_\omega}$. The standard criterion of barb sensitivity is then $S{\Downarrow_\omega} \Leftrightarrow [\![S]\!]{\Downarrow_\omega}$ for all $\omega \in \Omega$.

A traditional choice of barb in the π-calculus is to take $\Omega = \{x, \bar{x} \mid x \in \mathcal{N}\}$, writing $P{\downarrow_x}$, resp. $P{\downarrow_{\bar{x}}}$, when $x \in \mathrm{fn}(P)$ and P has an unguarded occurrence of a subterm $x(z).R$, resp. $\bar{x}y.R$ [49]. The philosophy behind the asynchronous π-calculus entails that input actions $x(z)$ are not directly observable (while output actions can be observed by means of a matching input of the observer). This leads to semantic identifications like $\mathbf{0} = x(y).\bar{x}y$, for in both cases the environment may observe $\bar{x}z$ only if it supplied $\bar{x}z$ itself first. Yet, these processes differ on their input barbs (${\downarrow_x}$). For this reason, in aπ normally only output barbs ${\downarrow_{\bar{x}}}$ are considered [49]. Boudol's encoding satisfies the criterion of output barb sensitivity (and in fact also input barb sensitivity). However, the encoding of Honda & Tokoro does not, as it swaps input and output barbs. As such, it is an excellent example of the benefit of the external barb $\surd$.

Validity up to a semantic equivalence

In [21] a compositional encoding is called *valid up to* a semantic equivalence $\sim \subseteq \mathcal{P} \times \mathcal{P}$ if $[\![P]\!] \sim P$ for all $P \in \mathcal{P}$.[3] A given encoding may be valid up to a coarse equivalence, and invalid up to a finer one. The equivalence for which it is valid is then a measure of the quality of the encoding.

Combining the results of the current paper with those of [40] shows that the encodings of Boudol and Honda & Tokoro are valid up to reduction-based *success respecting coupled similarity* ($CS^{\surd}$). Earlier, [10] established that Boudol's encoding is valid up to *may testing* [15] and *fair testing equivalence* [7, 33]—both results now follow from the validity up to $CS^{\surd}$. On the other hand, [10] also shows that Boudol's encoding is not valid up to a form of *must testing*; in [12] this result is strengthened to pertain to any encoding of π into aπ.

An interesting open question is whether the encodings of Boudol and Honda & Tokoro are valid up to reduction-based *success respecting weak bisimilarity* or *weak barbed bisimilarity*. In [47], a polyadic version of Boudol's encoding is assumed to be valid up to the version of weak barbed bisimilarity that uses output barbs only; see Lemma 17. Yet, as no proof is provided, the question remains open.

Acknowledgement. Fruitful feedback from Stephan Mennicke and from IPL referees is gratefully acknowledged.

References

[1] D. Austry & G. Boudol (1984): *Algèbre de processus et synchronisations. Theoretical Computer Science* 30(1), pp. 91–131, doi:10.1016/0304-3975(84)90067-7.

[2] M. Baldamus, J. Parrow & B. Victor (2004): *Spi Calculus Translated to π-Calculus Preserving May-Tests.* In: Proc. LICS'04, pp. 22–31, doi:10.1109/LICS.2004.1319597.

[3] J.A. Bergstra & J.W. Klop (1986): *Algebra of communicating processes.* In: *Mathematics and Computer Science*, CWI Monograph 1, North-Holland, pp. 89–138.

[4] T. Bolognesi & E. Brinksma (1987): *Introduction to the ISO Specification Language LOTOS.* Computer Networks 14, pp. 25–59, doi:10.1016/0169-7552(87)90085-7.

[5] M. Boreale (1998): *On the Expressiveness of Internal Mobility in Name-Passing Calculi.* TCS 195(2), pp. 205–226, doi:10.1016/S0304-3975(97)00220-X.

[3] [21] distinguishes between a process term and its meaning or denotation, $\sim$ being defined on the denotations. Here, in line with [23] and most other related work, I collapse syntax and semantics by only considering process terms without process variables, taking the denotation of a term to be itself (up to α-conversion).

[6] G. Boudol (1992): *Asynchrony and the π-calculus (Note)*. Tech. Rep. 1702, INRIA.

[7] E. Brinksma, A. Rensink & W. Vogler (1995): *Fair Testing*. In: Proc. CONCUR'95, LNCS 962, Springer, pp. 313–327, doi:10.1007/3-540-60218-6_23.

[8] S.D. Brookes, C.A.R. Hoare & A.W. Roscoe (1984): *A theory of communicating sequential processes*. Journal of the ACM 31(3), pp. 560–599, doi:10.1145/828.833.

[9] N. Busi, M. Gabbrielli & G. Zavattaro (2009): *On the expressive power of recursion, replication and iteration in process calculi*. MSCS 19(6), pp. 1191–1222, doi:10.1017/S096012950999017X.

[10] D. Cacciagrano & F. Corradini (2001): *On Synchronous and Asynchronous Communication Paradigms*. In: Proc. 7th Italian Conference on Theoretical Computer Science, ICTCS'01, LNCS 2202, Springer, pp. 256–268, doi:10.1007/3-540-45446-2_16.

[11] D. Cacciagrano, F. Corradini, J. Aranda & F.D. Valencia (2008): *Linearity, Persistence and Testing Semantics in the Asynchronous Pi-Calculus*. ENTCS 194(2), pp. 59–84, doi:10.1016/j.entcs.2007.11.006.

[12] D. Cacciagrano, F. Corradini & C. Palamidessi (2007): *Separation of synchronous and asynchronous communication via testing*. Theoretical Computer Science 386(3), pp. 218–235, doi:10.1016/j.tcs.2007.07.009.

[13] M. Carbone & S. Maffeis (2003): *On the Expressive Power of Polyadic Synchronisation in pi-calculus*. Nordic Journal of Computing 10(2), pp. 70–98.

[14] L. Cardelli & A.D. Gordon (2000): *Mobile ambients*. Theoretical Computer Science 240(1), pp. 177–213, doi:10.1016/S0304-3975(99)00231-5.

[15] R. De Nicola & M. Hennessy (1984): *Testing equivalences for processes*. Theoretical Computer Science 34, pp. 83–133, doi:10.1016/0304-3975(84)90113-0.

[16] T. Given-Wilson (2014): *Expressiveness via Intensionality and Concurrency*. In: Proc. ICTAC'14, LNCS 8687, pp. 206–223, doi:10.1007/978-3-319-10882-7_13.

[17] T. Given-Wilson (2014): *On the Expressiveness of Intensional Communication*. In: Proc. EXPRESS/SOS'14, EPTCS 160, pp. 30–46, doi:10.4204/EPTCS.160.4.

[18] T. Given-Wilson & A. Legay (2015): *On the Expressiveness of Joining*. In: Proc. ICE'15, EPTCS 189, pp. 99–113, doi:10.4204/EPTCS.189.9.

[19] T. Given-Wilson & A. Legay (2016): *On the Expressiveness of Symmetric Communication*. In: Proc. ICTAC'16, LNCS 9965, Springer, pp. 139–157, doi:10.1007/978-3-319-46750-4_9.

[20] R.J. van Glabbeek (1994): *On the expressiveness of ACP*. In: Proc. ACP'94, Workshops in Computing, Springer, pp. 188–217, doi:10.1007/978-1-4471-2120-6_8.

[21] R.J. van Glabbeek (2012): *Musings on Encodings and Expressiveness*. In: Proc. EXPRESS/SOS'12, EPTCS 89, pp. 81–98, doi:10.4204/EPTCS.89.7.

[22] D. Gorla (2010): *A taxonomy of process calculi for distribution and mobility*. Distributed Computing 23(4), pp. 273–299, doi:10.1007/s00446-010-0120-6.

[23] D. Gorla (2010): *Towards a unified approach to encodability and separation results for process calculi*. I&C 208(9), pp. 1031–1053, doi:10.1016/j.ic.2010.05.002.

[24] J.F. Groote & M.R. Mousavi (2014): *Modeling and Analysis of Communicating Systems*. MIT Press.

[25] B. Haagensen, S. Maffeis & I. Phillips (2008): *Matching Systems for Concurrent Calculi*. ENTCS 194(2), pp. 85–99, doi:10.1016/j.entcs.2007.11.004.

[26] M. Hatzel, C. Wagner, K. Peters & U. Nestmann (2015): *Encoding CSP into CCS*. In: Proc. EXPRESS/SOS'15, EPTCS 190, pp. 61–75, doi:10.4204/EPTCS.190.5.

[27] K. Honda & M. Tokoro (1991): *An Object Calculus for Asynchronous Communication*. In: Proc. ECOOP'91, LNCS 512, Springer, pp. 133–147, doi:10.1007/BFb0057019.

[28] I. Lanese, J.A. Pérez, D. Sangiorgi & A. Schmitt (2010): *On the Expressiveness of Polyadic and Synchronous Communication in Higher-Order Process Calculi*. In: Proc ICALP'10, LNCS 6199, pp. 442–453, doi:10.1007/978-3-642-14162-1_37.

[29] R. Milner (1980): *A Calculus of Communicating Systems*. LNCS 92, Springer, doi:10.1007/3-540-10235-3.

[30] R. Milner (1983): *Calculi for synchrony and asynchrony*. TCS 25, pp. 267–310, doi:10.1016/0304-3975(83)90114-7.

[31] R. Milner (1992): *Functions as Processes*. MSCS 2(2), pp. 119–141, doi:10.1017/S0960129500001407.

[32] R. Milner, J. Parrow & D. Walker (1992): *A Calculus of Mobile Processes, Part I + II*. Information and Computation 100(1), pp. 1–77, doi:10.1016/0890-5401(92)90008-4.

[33] V. Natarajan & R. Cleaveland (1995): *Divergence and Fair Testing*. In: Proc. ICALP'95, LNCS 944, Springer, pp. 648–659, doi:10.1007/3-540-60084-1_112.

[34] U. Nestmann (2000): *What is a "Good" Encoding of Guarded Choice?* Information and Computation 156(1-2), pp. 287–319, doi:10.1006/inco.1999.2822.

[35] U. Nestmann & B.C. Pierce (2000): *Decoding Choice Encodings*. I&C 163(1), pp. 1–59, doi:10.1006/inco.2000.2868.

[36] C. Palamidessi (2003): *Comparing The Expressive Power Of The Synchronous And Asynchronous Pi-Calculi*. MSCS 13(5), pp. 685–719, doi:10.1017/S0960129503004043.

[37] C. Palamidessi, V.A. Saraswat, F.D. Valencia & B Victor (2006): *On the Expressiveness of Linearity vs Persistence in the Asychronous Pi-Calculus*. In: Proc. LICS'06, IEEE Computer Society Press, pp. 59–68, doi:10.1109/LICS.2006.39.

[38] C. Palamidessi & F.D. Valencia (2005): *Recursion vs Replication in Process Calculi: Expressiveness*. Bulletin of the EATCS 87, pp. 105–125.

[39] J. Parrow (2000): *Trios in concert*. In: Proof, Language, and Interaction, Essays in Honour of Robin Milner, The MIT Press, pp. 623–638.

[40] K. Peters & R.J. van Glabbeek (2015): *Analysing and Comparing Encodability Criteria*. In: EXPRESS/SOS'15, EPTCS 190, pp. 46–60, doi:10.4204/EPTCS.190.4.

[41] K. Peters & U. Nestmann (2012): *Is It a "Good" Encoding of Mixed Choice?* In: Proc. FoSSaCS'12, LNCS 7213, pp. 210–224, doi:10.1007/978-3-642-28729-9_14.

[42] K. Peters & U. Nestmann (2016): *Breaking symmetries*. MSCS 26(6), pp. 1054–1106, doi:10.1017/S0960129514000346.

[43] K. Peters, U. Nestmann & U. Goltz (2013): *On Distributability in Process Calculi*. In: Proc. ESOP'13, LNCS 7792, pp. 310–329, doi:10.1007/978-3-642-37036-6_18.

[44] K. Peters, J.-W. Schicke & U. Nestmann (2011): *Synchrony vs Causality in the Asynchronous Pi-Calculus*. In: Proc. EXPRESS'11, EPTCS 64, pp. 89–103, doi:10.4204/EPTCS.64.7.

[45] I. Phillips & M.G. Vigliotti (2006): *Leader election in rings of ambient processes*. Theoretical Computer Science 356(3), pp. 468–494, doi:10.1016/j.tcs.2006.02.004.

[46] I. Phillips & M.G. Vigliotti (2008): *Symmetric electoral systems for ambient calculi*. Information and Computation 206(1), pp. 34–72, doi:10.1016/j.ic.2007.08.005.

[47] P. Quaglia & D. Walker (2000): *On Synchronous and Asynchronous Mobile Processes*. In: Proc. FoSSaCS'00, LNCS 1784, Springer, pp. 283–296, doi:10.1007/3-540-46432-8_19.

[48] D. Sangiorgi (1993): *From π-calculus to Higher-Order π-calculus — and back*. In: Proc. TAPSOFT'93, LNCS 668, pp. 151–166, doi:10.1007/3-540-56610-4_62.

[49] D. Sangiorgi & D. Walker (2001): *The π-calculus: A Theory of Mobile Processes*. Cambridge University Press.

[50] R. de Simone (1985): *Higher-level synchronising devices in* MEIJE-*SCCS*. Theoretical Computer Science 37, pp. 245–267, doi:10.1016/0304-3975(85)90093-3.

[51] F.W. Vaandrager (1993): *Expressiveness Results for Process Algebras*. In: Proc. REX Workshop on Semantics: Foundations and Applications, LNCS 666, Springer, pp. 609–638, doi:10.1007/3-540-56596-5_49.

[52] C. Versari, N. Busi & R. Gorrieri (2009): *An expressiveness study of priority in process calculi*. MSCS 19(6), pp. 1161–1189, doi:10.1017/S0960129509990168.

[53] M.G. Vigliotti, I. Phillips & C. Palamidessi (2007): *Tutorial on separation results in process calculi via leader election problems*. TCS 388(1-3), pp. 267–289, doi:10.1016/j.tcs.2007.09.001.