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

Rob J. van Glabbeek\textsuperscript{a,b}

\textsuperscript{a}Data61, CSIRO, Sydney, Australia
\textsuperscript{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, quality criteria for encodings, π-calculus

1. Introduction

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

To order these calculi w.r.t. expressiveness, encodings between them have been studied \textsuperscript{50, 48, 51, 20, 5, 39, 35, 36, 34, 14, 47, 9, 13, 2, 38, 37, 45, 12, 53, 11, 25, 46, 52, 26, 10, 13, 12}. Process calculus \( L_1 \) is said to be at least as expressive as process calculus \( L_2 \) iff there exists a valid encoding from \( L_1 \) into \( L_2 \). 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 \textsuperscript{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 \textsuperscript{22, 28, 44, 11, 12, 16, 17, 18, 19}. The goal of this paper is fill this gap and formally establish that the encodings of \textsuperscript{6, 27} indeed are valid à la Gorla.

Section 2 recalls Gorla’s proposal for validity of an encoding; for their motivation see \textsuperscript{22}. Section 3 presents the encodings of \textsuperscript{6} and \textsuperscript{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 \textsuperscript{21}.

2. Valid encodings

In \textsuperscript{22} a process calculus is given as a triple \( \mathcal{L}=(\mathcal{P}, \rightarrow, \sim) \), where

- \( \mathcal{P} \) is the set of language terms (called processes), built up from \( k \)-ary composition operators \( \text{op} \).
- \( \rightarrow \) is a binary reduction relation between processes.
- \( \sim \) 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 \( xy \) 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; \( \text{fn}(\vec{P}) \) 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 \( \sigma: \mathcal{N} \rightarrow \mathcal{N} \); it extends componentwise to \( k \)-tuples of names. If \( P \in \mathcal{P} \) and \( \sigma \) 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 \( C[\vec{a}]; \ldots; \vec{a}] \) is a term build by the composition operators of \( \mathcal{L} \) from holes \( \vec{a}; \ldots; \vec{a} \); each of
these holes must occur exactly once in the context. If $C[\cdot;\ldots;\cdot]$ 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 $\cdot$ for each $i=1,\ldots,k$, while renaming bound names to avoid capture.

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

For the purpose of comparing the expressiveness of languages, a constant $\sqrt{\cdot}$ is added to each of them [22]. A term $P$ in the upgraded language is said to capture.

Write $L \rightarrow P$ into $P$ if $P$ is a constant.
The terms of the source and target languages for languages, a constant $\sqrt{\cdot}$ must occur exactly once in the context. If $\sqrt{\cdot}$ occurs in $P$, it has an unguarded occurrence of $\sqrt{\cdot}$, that is, if $P = \mathcal{C}[\sqrt{\cdot}]$ where the hole $\cdot_1$ appears unguarded in $C[\cdot]$. Here a hole $\cdot_1$ appears unguarded in a context $C[\cdot]$ when

$$C[P] \rightarrow \text{iff } P \rightarrow \emptyset.$$ Write $P \downarrow$ if $P \rightarrow P'$ for a process $P$ with $P' \downarrow$.

**Definition 1 (22).** An encoding of $\mathcal{L}_1 = (P_1,\rightarrow_1,=)$ into $\mathcal{L}_2 = (P_2,\rightarrow_2,=)$ is a pair $[\cdot;\ldots;\cdot]$ such that $P_1 \rightarrow P_2$ is called translation and $\varphi_{\cdot} : \mathcal{N} \rightarrow \mathcal{N}$ for some $k \in \mathbb{N}$ is called renaming policy and is such that for $u \neq v$ the $k$-tuples $\varphi_{\cdot}(u)$ and $\varphi_{\cdot}(v)$ 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 (22).** An encoding is valid if it satisfies the following five criteria.

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

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

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

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

   $$[S\sigma] = [S]\sigma' \quad \text{if } \sigma \text{ is injective}$$

   $$[S\sigma] \approx_2 [S]\sigma' \quad \text{otherwise}$$

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

3. **Operational correspondence:**

   **Completeness** if $S \implies 1 \implies 1'$ then $[S] \implies_1 [S']$

   **Soundness** and if $[S] \implies_1 1$ then $\exists 1' : S \implies_1 1'$ and $T \implies_1 2 \implies_{1'} 2' \implies_{1'} [S]$.

4. **Divergence reflection:** if $[S] \implies_1 1'$ then $S \implies_{1'} 1$.

5. **Success sensitiveness:** $S \downarrow$ if $[S] \downarrow$.

For this purpose $[\cdot;\ldots;\cdot]$ is extended to deal with the added constant $\sqrt{\cdot}$ by taking $[\sqrt{\cdot}] = \sqrt{\cdot}$.

3. **Encoding a synchronous into an asynchronous $\pi$-calculus**

Consider the $\pi$-calculus as presented by Milner in [31], i.e.,
the one of Sangiorgi and Walker without matching, $\pi$-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

$$P ::= 0 \ | \ xy.P \ | \ x(z).P \ | \ P | P' \ | (z)P \ | \ !P$$

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

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

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

\begin{align*}
(1) \quad P_1[P_2/P_3] = P_3(P_1[P_2/P_3]) & \equiv 0 \\
(2) \quad P_1P_2 = P_2P_1 & \equiv (z)(u)P \equiv (u)(z)P \\
(3) \quad P0 \equiv P & \equiv (w)(Q)(P) \equiv P(w)Q \\
(4) \quad !P \equiv P!P & \equiv x(z).P \equiv P(w/z).P' \\
Q & \equiv P \equiv P' \equiv P' & \equiv Q' \\
Q & \equiv Q & \equiv Q'
\end{align*}

The asynchronous $\pi$-calculus, as introduced by Honda & Tokoro [27] and by Boudol in [8], is the sublanguage $\pi^\alpha$ of the fragment $\pi$ of the $\pi$-calculus presented above where all subexpressions $xy.P$ have the form $x(y).0$. and are written $xy$.

Boudol [8] defined an encoding $[\cdot;\ldots;\cdot]_\pi$ from $\pi$ to $\pi^\alpha$ inductively as follows:

$$[0]_\pi = 0$$

$$[xy.P]_\pi = (u)(xu)(v)(v)(P)$$

$$[x(z).P]_\pi = x(u), (v)w(x)(P)$$

$$[P]_\pi = (P)$$

$$[!P]_\pi = !P$$

$$[(x)P]_\pi = (x)P$$

always choosing $u, v \notin \text{fn}(P) \cup \{x, y\}$, $u \neq v$. The encoding $[\cdot;\ldots;\cdot]_{\pi^\alpha}$ of Honda & Tokoro [27] differs only in the clauses for the input and output prefix:

$$[xy.P]_{\pi^\alpha} = x(u), (u)(P)$$

$$[(x)P]_{\pi^\alpha} = u(xu)(z)(P)$$

again choosing $u \notin \text{fn}(P) \cup \{x, y\}$.
4. Validity of Boudol’s encoding

In this section I show that Boudol’s encoding satisfies all five criteria of Gorla. I will drop the subscript p.

4.1. Compositionality

Boudol’s encoding is compositional by construction, for it is defined in terms of the contexts \( C_{\text{Sp}} \) 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 \((\bar{s}_1, \varphi_{\bar{s}_1})\), of which the second component, the renaming policy, is relevant only for satisfying the criterion of name invariance. Here I take \( k = 1 \) and \( \varphi_{\bar{s}_1} : N \to N \) the identity mapping.

Lemma 1. Let \( S \in \mathcal{P}_\pi \). Then \( \text{fn}(\text{S}) = \text{fn}(S) \).

Moreover, \( \text{S}(\text{u} / \text{v}) = \text{S}(\text{u} / \text{v}) \) for any \( u, v \in N \).

Proof. A straightforward structural induction on \( S \).

This implies that \( \text{S} \sigma = \text{S} \sigma \) for any renaming \( \sigma : N \to N \), injective or otherwise. So the criterion of name invariance is satisfied.

4.3. Operational correspondence

A process calculus à la Gorla is a triple \( \mathcal{L} = (\mathcal{P}, \to, \infty) \); so far I defined \( \mathcal{P} \) and \( \infty \) only. The semantic equivalence \( \equiv \) of the source language plays no role 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 \( \approx_\pi \) and \( \approx_\pi \) the identity relations.

If \( S \equiv R \) for \( S, R \in \mathcal{P}_\pi \) then there exists a sequence \( S_0 \equiv R_0 \equiv \cdots \equiv R_n = S_n \), such that each each step \( S_i \equiv R_i \), \( 0 \leq i < n \) is an application of one of the rules (1)–(9) of Definition 4 or their symmetric counterparts (1)–(9). (In fact, there is no need for rules (5), (6), (5) and (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 \( S_i = C[R] \) for some unary context \( C[L] \).

Operational completeness.

Lemma 2. If \( S \equiv R \) for \( S, R \in \mathcal{P}_\pi \) then \( [S] \equiv [R] \).

Proof. Using the reflexivity, symmetry and transitivity of \( \equiv \) one may restrict attention to the case that \( S \equiv R \) is a single application of a rule (1)–(9) of Definition 4. The proof proceeds by structural induction on the context \( C[L] \). The case that \( C[L] = [\cdot] \), 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 \([\cdot] \).

Lemma 3. Let \( S, S' \in \mathcal{P}_\pi \). If \( S \to S' \) then \( [S] \to [S'] \).

Proof. With induction on the derivation of \( S \to S' \).

- Let \( S = \bar{x}y.P | x(z).Q \) and \( S' = P[Q[\bar{y}/\bar{z}]] \). Pick \( u, v \notin \text{fn}(P) \cup \text{fn}(Q) \), with \( u \neq v \). Write \( P^* := \bar{y}[\bar{v}]P \) and \( Q^* := \bar{v}[\bar{u}]Q \).

\[
[S] = (u)(\bar{x}u(\bar{v})(\bar{v})(\bar{u})Q^*) \quad \to \quad (u)(\bar{v})(P^* \to Q^*)
\]

\[
[P][Q[\bar{y}/\bar{z}]] \quad \text{(using Lemma 1)}
\]

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

- Let \( S = (z)P \) and \( S' = (z)P' \), with \( P \to P' \). By the induction hypothesis, \( [P] \to [P'] \). Therefore, \( [S] \to [S'] \), as \( [S] = (z)[P] \) and \( [S'] = (z)[P'] \).

- The case that \( S = P[Q] \) and \( S' = P'[Q] \) with \( P \to P' \) proceeds likewise.

The above yields that \( S \to S' \) implies \( [S] \to [S'] \). So the criterion of operational completeness is satisfied.

Remark 1. The above proof shows that \( \to \) in Lemma 3 may be replaced by \( \to \) in \( \to \). As a direct consequence \( S \to \infty \) implies \([S] \to \infty \) (divergence preservation).

Operational soundness. The following result provides a normal form up to structural congruence for reduction steps in the asynchronous \( \pi \)-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 \( \bar{u} \) or \( 0 \) or \( 1 \). Moreover, \( (\bar{w})P \) for \( w = (w_1, \ldots, w_n) \subseteq N \) with \( n \in \mathbb{N} \) denotes \( \{w_1 \ldots w_n\}P \) for some arbitrary order of the \( (w_i) \). Without the statements that \( C \) is plain and \( \bar{w} \subseteq \text{fn}(\bar{x}y)x(z).R[C] \), this lemma is a simplification, by restricting attention to the syntax of \( \pi \)-calculus, of Lemma 1.2.20 in [49], established for the full \( \pi \)-calculus.

Lemma 4. If \( T \to T' \) with \( T, T' \in \mathcal{P}_{\pi} \) then there are \( w \subseteq N, x, y, z \in N \) and terms \( R, C \in \mathcal{P}_{\pi} \) with \( C \equiv \bar{x}y \), such that \( T \equiv (\bar{w})(\bar{x}y)x(z).C \to (\bar{w})(\bar{0}R[\bar{y}/\bar{z}])C \equiv T' \) and \( \bar{w} \subseteq \text{fn}(\bar{x}y)x(z).R[C] \).

Proof. The reduction \( T \to T' \) 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 \( T^{\text{pre}} \) and \( T^{\text{post}} \) such that \( T \equiv R(T^{\text{pre}} \to T^{\text{post}}) \equiv R(T') \), and the reduction \( T^{\text{pre}} \to T^{\text{post}} \) is generated by the first three rules of Definition 4. Applying rules (8), (9), (2) and (7) of structural congruence, the terms \( T^{\text{pre}} \) and
$T^\text{post}$ can be brought in the forms $(\tilde{w})^\text{pre}$ and $(\tilde{w})^\text{post}$, with $P^\text{pre}$ and $P^\text{post}$ plain, at the same time moving all applications of the reduction rule for restriction $(\ast)P$ after all applications of the rule for parallel composition. Applying rules $(\tilde{1}), (\tilde{3})$, all applications of the reduction rule for parallel composition can be merged into a single application. After this proof normalisation, the reduction $T^\text{pre} \rightarrow T^\text{post}$ is generated by one application of the first reduction rule of Definition [4] followed by one application of the rule for $(\ast)$, followed by applications of the rule for restriction. Now $T^\text{pre}$ has the form $(\tilde{w})((xy)(x)(R))(C)$ and $T^\text{post} = (\tilde{w})((0)(R)[x/y])(C)$ with C plain.

Rules (3), (7), (5) and (3) of structural congruence, in combination with α-conversion (rules (8) and (9)), allow all names $w$ with $w \notin \text{fn}(xy)(x)(R)$ to be dropped from $\tilde{w}$, while preserving the validity of $T^\text{pre} \rightarrow T^\text{post}$.

Write $P \equiv_S Q$ if $P$ can be converted into $Q$ using applications of rules (1)–(3),(5),(9) only, in either direction, and $P \Rightarrow_Q$ if this can be done with rule (4), from left to right.

**Lemma 5.** Lemma [6] can be strengthened by replacing $T \equiv_R (\tilde{w})((xy)(x)(R))(C)$ by $T \Rightarrow_\ast (\tilde{w})((xy)(x)(R))(C)$.

**Proof.** Define rule $(\tilde{1})$ to commutes over rule (1) if for each sequence $P_0 \equiv_R Q_0 \equiv_R R$ with $P \equiv_R Q$ an application of rule (4) and $Q \equiv_R R$ an application of rule (1), there exists a term $Q'$ such that $P \equiv_R Q'$ holds by (possibly multiple) applications of rule (1) and $Q' \equiv_R R$ by applications of rule (4). As indicated in the table below, rule (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 (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 *.

<table>
<thead>
<tr>
<th>(1)</th>
<th>(2)</th>
<th>(3)</th>
<th>(4)</th>
<th>(5)</th>
<th>(6)</th>
<th>(7)</th>
<th>(8)</th>
<th>(9)</th>
</tr>
</thead>
<tbody>
<tr>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
</tr>
</tbody>
</table>

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

Since applications of rule (4) could be shifted to the right in this sequence, all applications of rule (4) can be shifted to the left. Hence $T \Rightarrow_\ast (\tilde{w})((xy)(x)(R))(C)$.

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

**Proof.** Similar to the proof of Lemma [2].

Note that a variant of Lemma [3] with (2), (3), (3'), or (7) in the rôle of (4) would not be valid.

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

**Observation 1.** If $(\tilde{w})P \equiv_S (\tilde{v})Q$ with $P, Q$ plain, $\tilde{w} \subseteq \text{fn}(P)$ and $\tilde{v} \subseteq \text{fn}(Q)$, then there is an injective renaming $\sigma:N \rightarrow N$ such that $\sigma(\tilde{w}) = \tilde{w}$ and $P \equiv_S Q$. Thus, for each parallel component $P'$ of $P$ of the form $xy.R$ or $x.(z).R$ or $\sigma$ or $\sigma'$ there is a parallel component $Q'$ of $Q$ with $P' \equiv_S Q'$.

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

**Lemma 7.** If $[S] \equiv_S (\tilde{w})(C)[x](x)(R).R$ with $S \in \mathcal{P}_\pi$, $C$ plain and $\tilde{w} \subseteq \text{fn}(C)[x](x)(R).R$, then there are $D, R_1, R_2 \in \mathcal{P}_\pi$, $F \in \mathcal{P}_\pi$, $y, z, v_1, v_2 \in N$ and $\tilde{t}, \tilde{s} \subseteq N$ such that $S \equiv_S ((\tilde{t})(D)\mid[x]R_1 \mid[x]R_2), v_1 \neq y \neq u, \tilde{w} \supseteq \tilde{w} \supseteq \tilde{w}[u/v][R]\in C \equiv \{F[v_1(1), v_1 (v_1) [R_1]], u, v_1 \notin \text{fn}(F), \{D \equiv_S \{F\} \equiv \{F(v_1)\mid[v_1] \mid[R_2]\}, \tilde{r} \neq v_2, v_2 \notin \text{fn}(F)\} \{\tilde{z}\}$.

**Proof.** By applying rules (8), (9), and $(\tilde{1})$ only, the term $[P_1] \ldots [P_n] \in \mathcal{P}_\pi$ can be brought into the form $(\tilde{q})P$ with $P$ plain; moreover the set $\tilde{q} \equiv \text{fn}(P)$ can be chosen disjoint from $\tilde{p}$.

Each $q \subseteq \tilde{q}$ is a renaming of the name $u$ in a term $[P] = [\tilde{q}, \tilde{y}].Q = (u)[xu][u]\{\tilde{y}([Q])\}$, so that $q \subseteq \text{fn}(P)$.

So $(\tilde{w})(C)[x](x)(R).R \equiv_S \{\tilde{q}\} \{\tilde{p}\}$. Let $\sigma$ be the renaming that exists by Observation [4] so that $\sigma(q) \equiv \sigma(\tilde{q}) = \tilde{w}$. Then $(x)[xu][x](x)(R).R \equiv_S \sigma(P)$. So $\tilde{xu} and (\tilde{x})r.(\tilde{y})r$ are both parallel components of $P_\pi$.

Let $\tilde{\sigma}$ be the restriction of $\sigma$ to $\tilde{p}$ and take $\tilde{s} := \tilde{s}^\prime(\tilde{\sigma})$. Let $S'' := ((\tilde{t})(P_1)^\sigma \ldots (P_n)^\sigma)$. Then $S \equiv_S S''$ and $[S] \equiv_S [S''] = (\tilde{t})(P_1)^\sigma \ldots (P_n)^\sigma)$. Since $[P_1] \ldots [P_n] \equiv_S (\tilde{q})(P)^\sigma$ by applications of rules of $(8), (9)$ and $(\tilde{1})$, $[P_1] \ldots [P_n] \equiv_S (\tilde{q})(P)^\sigma$ can be converted into $(\tilde{q})(P)^{\prime}\sigma$ and even into $(\tilde{q})(\tilde{\sigma})\{\tilde{\sigma}(\tilde{p})\}$ by applications of these rules. One can apply (8),(9) first, so that each $[P_1]^\sigma$ is converted into some term $Q_1$ by applications of (8),(9), and $Q_1 \ldots Q_n$ is converted into $(\tilde{q})(\tilde{\sigma})(\tilde{\sigma}(\tilde{p}))\{\tilde{\sigma}(\tilde{p})\}$ by applications of (2) and (7) only.

One of the $P_i$ must be of the form $xy.R_1$, so that $[P_i]^\sigma = [\tilde{y}](R_i) = (u')[(xu'[xv_1(1)].(v_1y)[R_i]])$ with $u', v_1 \notin \text{fn}(R_i) \cup \{x, y\}$, while $u'$ is renamed into $u$ in $Q_i = (u)[xu][u(v_1)].(v_1 y)[R_1])$. For this is the only way $\tilde{xu}$ can end up as a parallel component of $P_\pi$. It follows that $u, v_1 \notin \text{fn}(R_1)$ and $v_1 \neq y \neq u$ in $\sigma(\tilde{q})$. Let $t := \tilde{q} \setminus u$. One obtains $\tilde{w} := \tilde{w} \setminus t \setminus u$. For Searching for an explanation of the parallel component $x(r)R$. $\{\tilde{q}\} \Rightarrow_\ast \{\tilde{q}\}$ of $P_\pi$, the existence of the component $\tilde{xu}$ of $P_\pi$ excludes the possibility that one of the $P_i^\sigma$ is of the form $\tilde{v}'yR_2$ so that $[P_i]^\sigma' = (u')[(\tilde{v}'u'[\tilde{v}')(\tilde{r}'y')[[R_2]]],$
while $u'$ is renamed into $x$ and $r'$ into $r$ in the expression $Q_1 = (x)(t^r x)(r)(y^v)[R_2])$.  

Hence one of the $P\sigma'$ is of the form $x(z), R_2$, so that $[P\sigma'] = [P\sigma'] \equiv [x(z), R_2] = x(r').(v')'(r'(v')'(z), [R_2])$ with $r' \neq v'$ and $r', v' \notin \text{fn}(R_2) \{ z \}$, while $r', v'$ and $z$ are renamed into $r_2$ and $z'$ in $Q_1 = (x)(r_2)(r_2(v_2)(v_2(z), [R_2])$, where $(z') R_2 \equiv (s)_y(z) [R_2]$. Thus, $r_2 \notin \text{fn}(R_2) \{ z \}$ and $r \neq r_2$. Further, $x(z), R_2 \equiv (s)_y Q_1$, so $R_2 \equiv (v_2)(v_2(v_2(z), [R_2])$. Let $D$ collect all parallel components $P\sigma'$ other than the above discussed components $\bar{x}, R_1$ and $x(z). R_2$. Then $S \equiv S'' \equiv (s)(D) | \bar{x}, R_1 | x(z). R_2)$.  

One has $(\bar{u}(C)\bar{x}u(x)(r). R) \equiv S \equiv [S'] \equiv S' \equiv (s)(D) | \bar{x}, R_1 | x(z). R_2) = (\bar{u}(C)\bar{x}u'(v)(v). \bar{v}(y)[\bar{R}_1])) \equiv (s)(D) | \bar{x}(u(v)u(v)(v). \bar{v}(y)[\bar{R}_1])) \equiv (s)(D) | \bar{x}(u(v)u(v)(v). \bar{v}(y)[\bar{R}_1])) \equiv (s)(D) | \bar{x}(u(v)u(v)(v). \bar{v}(y)[\bar{R}_1])) \equiv (s)(D) | \bar{x}(u(v)u(v)(v). \bar{v}(y)[\bar{R}_1])) \equiv (s)(D) | \bar{x}(u(v)u(v)(v). \bar{v}(y)[\bar{R}_1])) \equiv (s)(D) | \bar{x}(u(v)u(v)(v). \bar{v}(y)[\bar{R}_1]))$.
In a reduction step of the form $P \Rightarrow Q$, the reacting prefixes $ab$ and $a(e).D$ are always found in the scope of a restriction operator $(a)$, and without a ! between $(a)$ and $ab$ or $a(e).D$, such that in this scope there are no other unguarded occurrences of prefixes $ad$ or $a(e).E$. This follows by a trivial induction on the definition of $\Rightarrow$. In particular, this property is preserved when applying structural congruence to $P$. Consequently, the plain term $C''$ has no parallel components of the form $\bar{y}(z).R''$.

Case 1: $x = u$. Then, by the above, $(z)R \equiv (r)R''$, $y = v$ and $C \equiv C''$. Consequently, $P' \equiv Q$.

Case 2: $x \neq u$. Then $\bar{w}$ and $u(r).R''$ (up to $\equiv$) must be a parallel components of $C$, and $\bar{xy}$ and $x(z).R$ (up to $\equiv$) must be parallel components of $D$, so that $P \equiv R \bar{P}_0 = (\bar{w})(\bar{xy}[x(z).R]([\bar{w}u(r).R''])D)$, where $C \equiv \bar{w}u(r).R''$ and $C'' \equiv (\bar{xy}[x(z).R])D$. This shows that the reductions $P \Rightarrow Q$ and $P' \Rightarrow P''$ are concurrent, so that there is a $Q'$ with $Q \Rightarrow Q'$ and $P' \Rightarrow Q'$.

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

**Proof.** By repeated application of Lemma 9.

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

By combining Corollary 9 with Observations 8 and 1 one finds that the criterion of operational soundness is met.

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

**Proof.** With induction on the length $n$ of the sequence $[S] \Rightarrow T$. The base case $n = 0$ is trivial: take $S' := S$. So let $[S] \Rightarrow T_1 \Rightarrow T$, where $T_1 \Rightarrow T$ has length $n$. By Observation 9 there is $S_1 \Rightarrow S_1$ and $T_1 \Rightarrow [S_1]$. By Corollary 8 $3T'$ with $[S_1] \Rightarrow T'$ and $T \Rightarrow^* T'$.

Furthermore, the sequence $[S_1] \Rightarrow T'$ has length $\leq n$. By induction, there is a $S'$ with $S \Rightarrow^* S'$ and $T' \Rightarrow [S']$. Hence $S \Rightarrow S'$ and $T \Rightarrow [S']$, using Observation 8.

4.4. Divergence reflection

**Corollary 3.** If $P \Rightarrow Q$ and $P \not\Rightarrow^* Q$ then $Q \not\Rightarrow^* Q$.

**Proof.** By repeated application of Lemma 9 (Compare the proof of Corollary 1).

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

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

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

4.5. Success sensitiveness

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

$$\downarrow \quad \begin{array}{c} P \downarrow \quad (PQ) \downarrow \quad ((z)P) \downarrow \quad (!P) \downarrow \\
\end{array}$$

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

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

**Proof.** A trivial structural induction.

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

**Proof.** By Lemma 3 there are $\bar{w} \subseteq N$, $x, u, r \in N$ and $R, C \in \mathcal{P}_\pi$ with $C$ plain, such that $\bar{w} \subseteq \text{fn}((\bar{x}u[x(r)].R)(C)$ and $T \equiv (\bar{w})((\bar{x}u[x(r)].R)(C) \Rightarrow (\bar{w})((0)R)(\bar{y}r)) (C) \equiv T'$. Since $T \downarrow$, it must be that $C \downarrow$ and hence $T' \downarrow$.

By combining Lemmas 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 $S \not\downarrow$ if $[S] \not\downarrow$.

**Proof.** Suppose that $S \not\downarrow$. Then $S \Rightarrow S'$ for a process $S'$ with $S' \not\downarrow$. By Lemma 3 $[S] \Rightarrow [S']$. By Lemma 10 $[S'] \not\downarrow$. Hence $[S] \not\downarrow$.

Now suppose $[S] \not\downarrow$. Then $[S] \Rightarrow T$ for a process $T$ with $T \not\downarrow$. By Theorem 4 there is a $S'$ with $S \Rightarrow S'$ and $T \Rightarrow [S']$. By Lemma 11 $[S'] \not\downarrow$. By Lemma 10 $S' \not\downarrow$. Hence $S \not\downarrow$.

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 [ ] stands for $\downarrow^\uparrow$.

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

**Proof.** With induction on the derivation of $S \Rightarrow S'$.

- Let $S = \bar{xy}.P[x(z).Q, y \notin \text{bn}(Q)]$ and $S' = P[Q[\bar{y}z]]$.

  Pick $u \notin \text{fn}(P) \cup \text{fn}(Q) \cup \{x, y\}$. Then

  $$[S] = \begin{array}{c} x\bar{u}_y, (\bar{u}[\bar{y}[P]]) \quad (u)(\bar{u}[xu][z].[Q]) \\
\end{array}

- (using Lemma 1)

  $$= \begin{array}{c} [P] \quad [Q[\bar{z}]] \quad [P] \quad [Q[\bar{y}[z]]] \quad (using \text{Lemma 1})

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

- The other three cases proceed as in the proof of Lemma 8.
Lemma 13. If \([S] \equiv_S (\overline{u})(\overline{C}[\overline{w}(x.R)].R)\) with \(R \in \mathcal{P}_\pi\),
\(C\) plain and \(\overline{w} \subseteq \text{fin}(\overline{C}[\overline{w}(x.R)].R)\), then there are terms
\(D, R_1, R_2 \in \mathcal{P}_\pi\), \(F \in \mathcal{P}_\pi\), \(x, y \in N\)
and \(s, t, i \subseteq N\) such that \(S \equiv_S (\overline{s})(D) \overline{x}.R_1 | x.(R_2).R_2\), \(\overline{w} = \overline{s} \cup \overline{t} \cup \{u\}
\),
\(C \equiv_S F[\overline{u}(x)].R_2\), \(u \notin \text{fin}(\overline{R}_2)\setminus\{z\}\), \([D] \equiv_S (\overline{t})F\), \(r \neq y\),
\(R \equiv_S r[y][R_1]\) and \(r \notin \text{fin}(\overline{R}_1)\).

Proof. The first two paragraphs proceed exactly as in the proof of Lemma 7.
Each \(q \in \overline{g}\) is a renaming of the name \(u\) in a term
\([P]_1 = [x(z).Q] = (u)(\overline{x}u(z).[Q])\), so that \(q \in \text{fin}(P)\).
The next two paragraphs proceed exactly as in the proof of Lemma 7.

One of the \(P_i\sigma'\) must be of the form \([x(z).R]\), \(R_2\), so that
\([P]_1\sigma' = [P_1\sigma'] = [x(z).R_2] = (u')(\overline{x}u'(u'(z).R_2)]\) with \(u' \notin \text{fin}(R_2)\setminus\{z\} \cup \{x\}\),
while \(u'\) and \(z\) are renamed into \(u\) and \(z'\) in \(Q_1 = \overline{u}(\overline{x}u(z').R_2)\), where \((z')R_2 \equiv_S \{s\}(9)\)
\([R_2]\) for this is the only way \(x\) can end up as a parallel component of \(P\).
It follows that \(u \notin \text{fin}(R_2)\setminus\{z\}\)
and \((u')R_2 \in \overline{u}(\overline{g})\). One obtains \(\overline{w} = \overline{s} \cup \overline{u}[\overline{w}][u]\).

Searching for an explanation of the parallel component
\((x.R).R\) (up to \(\equiv_S\)) of \(P\), the existence of the component
\(\overline{x}u\) of \(P\) 
\(\sigma\) excludes the possibility that one of the \(P_i\sigma'\) is
of the form \((r'\sigma').R\) so that \([P]_1\sigma' = \overline{u}(r'u'u'(r'.R_1)]\),
while \(u'\) is renamed into \(x\) and \(r'\) into \(r\) in the expression
\(Q_1 = (u')(\overline{x}u(x)[r].R_1)\).

Hence one of the \(P_i\sigma'\) is of the form \(\overline{x}u.R_1\) and \((x.\overline{z}).R_2\).
Then \(S \equiv_S S' \equiv (\overline{s})(D) \overline{x}.R_1 | x.(R_2).R_2\).

One has \(\overline{w}(\overline{C}[\overline{x}u(x).R].R) \equiv_S \overline{w}C \equiv_S S'
\equiv_S [(\overline{s})(D) \overline{x}.R_1 | x.(R_2).R_2] = \overline{s}(D) \overline{x}.R_1 | x.(R_2).R_2)\).

Here \(E\) is the parallel composition of all components \(Q_i\) obtained by renaming of the parallel components \(P_i\sigma'\)
of \([D]\), and \((\overline{l})F\) with \(F\) plain is obtained from \(E\) by rules
\((2),(7)\) or \(C \equiv_S \overline{x}u(x).R_2 \equiv_S \overline{x}u(x).R_2 | x.(R).R\) by Observation 9.
It follows that \(C \equiv_S F | u(z).[R_2].R\).

Lemma 14. Let \(S \in \mathcal{P}_\pi\) and \(T \in \mathcal{P}_\pi\). If \([S] \to S'\) then there is a \(S'\) with \(S \equiv S' \to T' \equiv [S']\).

Proof. The first paragraph proceeds as in the proof of Lemma 8
but incorporating the conclusion of Lemma 10 instead of Lemma 7.
Again, one finds, for all \(t, i\), that \(t, u \notin \text{fin}(D) \cup \{x, y\} \cup \text{fin}(R_1) \cup \text{fin}(R_2) \setminus \{z\}\).

Take \(S' \equiv (\overline{s})(D) | R_1 | R_2[\overline{y}z]\). Then \(S' \equiv S' \to S'\) and
\(T \equiv (\overline{s})(\overline{C}) | R_2[\overline{y}z]\).

\(\equiv_R (\overline{s})(\overline{C}) | R_2[\overline{y}z]\) \(\equiv_R (\overline{s})(\overline{C}) | R_2[\overline{y}z]\) \(\equiv_R (\overline{s})(\overline{C}) | R_2[\overline{y}z]\) \(\equiv_R (\overline{s})(\overline{C}) | R_2[\overline{y}z]\)
\(\equiv_R (\overline{s})(\overline{C}) | R_2[\overline{y}z]\) \(\equiv_R (\overline{s})(\overline{C}) | R_2[\overline{y}z]\)
\(\equiv_R (\overline{s})(\overline{C}) | R_2[\overline{y}z]\) \(\equiv_R (\overline{s})(\overline{C}) | R_2[\overline{y}z]\)
source language. Making the dependence on the choice of set $N$ of names explicit, this method calls $\pi$ expressible into $\alpha\pi$ if for each $N$ there exists an $N'$ such that there is a valid encoding of $\pi(N)$ into $\alpha\pi(N')$.

Operational soundness. Operational soundness stems from Nestmann & Pierce [32], who proposed two forms of it:

(1) if $[S] \mapsto T$ then $\exists S': S \mapsto_1 S'$ and $T \mapsto_2 [S']$.
(2) if $[S] \mapsto T$ then $\exists S': S \mapsto_1 S'$ and $T \mapsto_2 [S']$.

The version of Gorla is the common weakening of these: (1) if $[S] \mapsto T$ then $\exists S': S \mapsto_1 S'$ and $T \mapsto_2 [S']$. An interesting intermediate form is

(2) if $[S] \mapsto T$ then $\exists S': S \mapsto_1 S'$ and $T \mapsto_2 [S']$.

Nestmann & Pierce observed that “nonsense encodings”, that “allow administrative (or book-keeping) steps to precede a committing step”, “do not satisfy (3)”. For such encodings, which include the ones studied here, they proposed (1). As I have shown, the encodings of Boudol and of Honda & Tokoro, satisfy not only (1) but even (2). It remains an interesting open question whether they satisfy (2). Clearly, they do not when taking $\mapsto_2$ to be the identity relation—as I did here—structural congruence. However, it is conceivable that (2) holds for another reasonable choice of $\mapsto_2$. (An unreasonable choice, such as the universal relation, tells us nothing.)


First of all, Gorla requires $\sqrt{\ }$ 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 $\sqrt{\ }$. Here I simply allowed $\sqrt{\ }$ 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 work has been used [22, 28, 44, 41, 43, 16, 17, 18, 19]. A traditional choice of barb in the $\pi$-calculus is to take $\Omega = \{x, \overline{x} | x \in \mathcal{N}\}$, writing $P_{\overline{x}x}$, resp. $P_{\overline{x}x}$, when $x \in \text{fin}(P)$ and $P$ has an unguarded occurrence of a subterm $x(z).R$, resp. $\overline{x}y.R$ [32]. The philosophy behind the asynchronous $\pi$-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 $0 = x(y).\overline{x}y$, for in both cases the environment may observe $\overline{x}y$ only if it supplied $\overline{x}y$ itself first. Yet, these processes differ on their input barbs $(\downarrow_\alpha)$. For this reason, in $\alpha\pi$ normally only output barbs $\downarrow_\alpha$ 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 $\sqrt{\ }$.

Validity up to a semantic equivalence. In [21] a compositional encoding is called valid up to a semantic equivalence $\sim \subseteq P \times P$ if $[P] \sim P$ for all $P \in P$. 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 [19] shows that the encodings of Boudol and Honda & Tokoro are valid up to reduction-based success respecting coupled similarity ($CS^\perp$). Earlier, [18] established that Boudol’s encoding is valid up to may testing [15] and fair testing equivalence [6, 53]—both results now follow from the validity up to $CS^{\perp}$. On the other hand, [19] 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 $\pi$ into $\alpha\pi$.

In 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 [17], 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.

References


graph 1, North-Holland, pp. 89–138.