Expressiveness of Point-to-Point versus Broadcast Communications

Cristian ENE[1,2], Traian MUNTEAN[1]

[1] University of Marseille
Laboratoire d'Informatique de Marseille (LIM-CNRS)
Parc Scientifique de Luminy - Case 925 - ESIL/ES2I;
F-13288 Marseille, France
muntean@lim.univ-mrs.fr, cene@esil.univ-mrs.fr
[2] on leave from University "Al. I. Cuza", Faculty of Computer Science,
6600 Iasi, Romania

Abstract. In this paper we address the problem of the expressive power of point-to-point communication to implement broadcast communication. We demonstrate that point-to-point communication as in CCS [M89] is "too asynchronous" to implement broadcast communication as in CBS [P95]. Milner's π-calculus [M91] is a calculus in which all communications are point-to-point. We introduce $b\pi$-calculus, using broadcast instead of rendez-vous primitive communication, as a variant of value-passing CBS in which communications are made on channels as in Hoare's CSP [H85] - and channels can be transmitted too as in π-calculus - but by a broadcast protocol: processes speak one at a time and are heard instantaneously by all others. In this paper, using the fact that π-calculus enjoys a certain interleaving property, whereas $b\pi$-calculus does not, we prove that there does not exist any uniform, parallel-preserving translation from $b\pi$-calculus into π-calculus, up to any "reasonable" equivalence. Using arguments similar to [P97], we also prove a separation result between CBS and CCS.

1 Introduction

Communication within processes is the main aspect of concurrency within distributed systems. One can specify basic communications from several points of view; primitives interactions can be, for instance, synchronous or asynchronous, associated to point-to-point or broadcast (one-to-many) message exchange protocols.

In theory (and in practice too), it arises naturally the question whether one mechanism can be "expressed" using the other (or, whether one can be implemented by the other). The first aspect (of synchrony/asynchrony) was, for instance, recently studied in [P97] in the framework of the π-calculus ([M91], [MPW92]). It was proved that there does not exist any uniform, parallel constructor-preserving, translation from π-calculus into the asynchronous π-calculus, up to any "reasonable" notion of equivalence.

In this paper, we address the question of the power of expressiveness of point-to-point communications versus broadcast communications. Indeed, we stress here that the broadcast is a natural manner of communication in concurrent and communicating systems found in many application fields. It has been also chosen as the hardware primitive exchange protocol for some networks, and in this case point-to-point message passing (when needed) is to be implemented on top of it. Broadcast can also be a natural primitive for many applications programming models (e.g. multimedia, data mining, etc.). Hence, it is natural to understand if a calculus for parallel/distributed computing based on basic broadcast communications ([EM99]) results in more powerful process calculi, or it is just a way of adding "syntactic sugar" to existing point-to-point based calculi. We chose Milner's π-calculus to study this problem, since it is recognised as one of the richest paradigm for concurrency introduced so far. In addition, the basic mechanism for communication in π-calculus is a point-to-point exchange event. When trying to express broadcast communication using point-to-point communication, what seems difficult, is how one can anticipate the size of the set of point-to-point communications needed when the number of potential receivers is "a priori" unknown. In addition, in π-calculus a system of two processes which exchange messages, can behave similarly in any context, while a broadcast communication is always "open" for the given environment. This is the intuition that we exploit in this paper to prove that broadcast communication cannot be "reasonably" simulated by using point-to-point communications of π-calculus.

The rest of the paper is as follows. In section 2 we briefly remind the bases of the π-calculus and then introduce the $b\pi$-calculus as a variant of a broadcast calculus (inspired from [P95]) together with some definitions concerning electoral systems. Section 3 presents the main result of the paper which proves the non-existence of any uniform encoding of $b\pi$-calculus into π-calculus. Section 4, discusses related works and presents future directions of research.

2 Preliminaries

In this section we briefly present the π-calculus, the $b\pi$-calculus (which is a variant of broadcast calculus), and then we introduce the notion of *electoral systems*.

2.1 The π-calculus

Let Ch_p be a countable set of *channels*. Then processes are defined in Table 1.

$p ::= \quad nil \mid A\langle\tilde{x}\rangle \mid \Sigma_{i\in I}\alpha_i.p_i \mid p_1 \parallel p_2 \mid \nu x p \mid (rec\ A\langle\tilde{x}\rangle.p)\langle\tilde{y}\rangle$

Table 1. Processes

where α_i belongs to the set of prefixes $\alpha ::= x(y) \mid \bar{x}y$.

Prefixes represent the basic actions of processes: $x(y)$ is the *input* of the name y on the channel x, and $\bar{x}y$ is the *output* of the name y on the channel

x. $\Sigma_{i\in I}\alpha_i.p_i$ represents guarded choice; when possible it begins by executing one of the atomic action α_i, and then its continuation is p_i. νxp is the creation of a new local channel x (whose initial scope is the process p). $p_1 \parallel p_2$ is the parallel composition of p_1 and p_2. $(recA\langle\tilde{x}\rangle.p)\langle\tilde{y}\rangle$ is a recursive process (allows to represent processes with infinite behaviour).

The operators νx and $y(x)$, are $x-binders$, i.e. in νxp and $y(x).p$, x appears bound, and $bn(p)$ represents the set of bound names of p. The free names of p are those that do not occur in the scope of any binder, and are denoted with $fn(p)$. The set of names of p is denoted with $n(p)$. The *alpha-conversion* is defined as usual.

In literature there have been defined relations among processes which relate processes which are "almost the same". Such a relation is a congruence, and allows to substitute a process by another congruent process, when needed.

Definition 1. *Structural congruence, denoted $\equiv$, is the smallest congruence over the set of processes which satisfies the conditions of Table 2.*

1	$p \equiv q$ if p and q are α-convertible
2	$(p \parallel q) \parallel r \equiv p \parallel (q \parallel r)$
3	$p \parallel q \equiv q \parallel p$
4	$p \parallel \nu xq \equiv \nu x(p \parallel q)$ if $x \notin fn(p)$

Table 2. Structural congruence

Definition 2. Actions, *ranged over α,β are given by the following syntax:*

$$\alpha \stackrel{def}{=} a(x) \mid \bar{a}x \mid \nu x\bar{a}x \mid \tau$$

where $a, x \in Ch_p$ and which reads as follows: an action is either a reception, a (possibly bound) output, or the silent action τ, denoting an uncontrollable transition.

We give an operational semantic of our calculus in terms of transitions over the set $\mathcal{P}_p$ of processes. Transitions are labeled by actions.

Definition 3. Transition system *The operational semantics of π-calculus is defined as a labeled transition system defined over the set $\mathcal{P}_p$ of processes. The judgement $p \xrightarrow{\alpha} p'$ means that that the process p is able to perform action α to evolve to p'. The operational semantics is given in Table 3.*

The semantic is an early one, i.e. the bound names of an input are instantiated as soon as possible, in the rule for input.

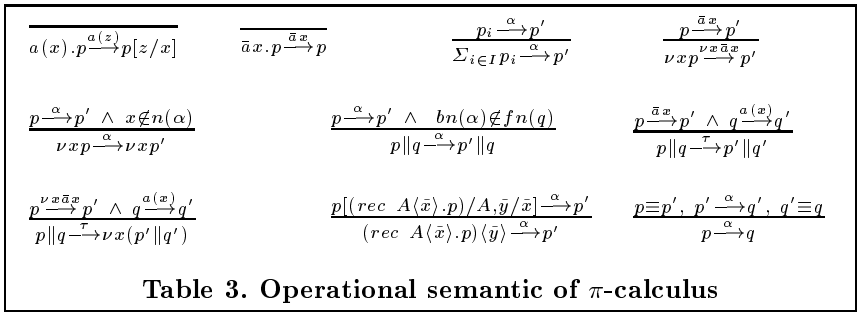

Table 3. Operational semantic of π-calculus

2.2 The $b\pi$-calculus

The $b\pi$-calculus is a process calculus in which broadcast is the fundamental communication paradigm. It is derived from broadcast calculus proposed by Prasad in [P95], and π-calculus. It differs from broadcast calculus, in that communications are made on channels (and transmitted values are channels too) which belong to a countable set Ch_b (like in π-calculus), and from π-calculus in the manner of use of channels: only for broadcast communications.

The syntax of *processes* is similar to that of π-calculus, as given in Table 1, and the meaning of process constructors is the same. Also, the set of *actions* (denoted $\mathcal{A}$), is defined similarly. The operational semantic (which is defined as a labeled transition system defined over the set $\mathcal{P}_b$ of $b\pi$-calculus processes) is different, and we describe it in Table 5. Before, we define similarly to [HR95], a relation $\longrightarrow\subseteq \mathcal{P}_b \times \mathcal{A}$ denoted $p \xrightarrow{\alpha:}$ (instead of $p \longrightarrow \alpha$) and which can be read "p discards the action α ".

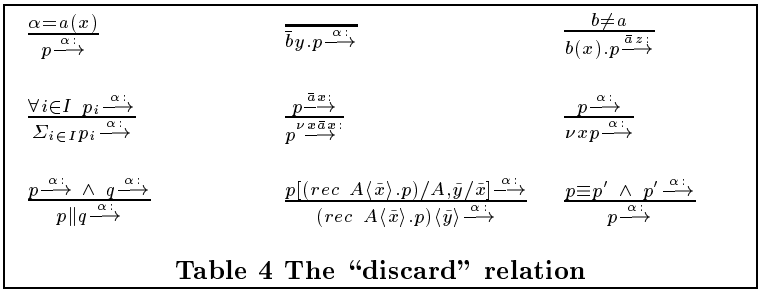

Table 4 The "discard" relation

The operational semantic is given, like for π-calculus, via a transition system labeled by actions. Communication between processes is performed through unbuffered broadcasts. Comparing with π-calculus, outputs are non-blocking, i.e. there is no need of a receiving process. One of processes broadcasts an output and the remaining processes either receive or ignore the broadcast according to whether they are "listening" or not on the channel which serves as support of the output. A process which "listen" on a channel a, can not ignore any value send on this channel.

$$\frac{}{a(x).p \xrightarrow{a(z)} p[z/x]} \qquad \frac{}{\bar{a}\,x.p \xrightarrow{\bar{a}\,x} p} \qquad \frac{p_i \xrightarrow{\alpha} p'}{\Sigma_{i\in I} p_i \xrightarrow{\alpha} p'}$$

$$\frac{p \xrightarrow{\alpha} p' \ \wedge \ x \notin n(\alpha)}{\nu x p \xrightarrow{\alpha} \nu x p'} \qquad \frac{p \xrightarrow{\bar{a}\,x} p'}{\nu x p \xrightarrow{\nu x \bar{a}\,x} \nu x p'} \qquad \frac{p \xrightarrow{\bar{a}\,x} p'}{\nu a p \xrightarrow{\tau} \nu a p'}$$

$$\frac{p \xrightarrow{\nu x \bar{a}\,x} \nu x p' \ \wedge \ q \xrightarrow{a(x)} q'}{p \| q \xrightarrow{\nu x \bar{a}\,x} \nu x (p' \| q')} \qquad \frac{p \xrightarrow{\bar{a}\,x} p' \ \wedge \ q \xrightarrow{a(x)} q'}{p \| q \xrightarrow{\bar{a}\,x} p' \| q'} \qquad \frac{p \xrightarrow{a(x)} p' \ \wedge \ q \xrightarrow{a(x)} q'}{p \| q \xrightarrow{a(x)} p' \| q'}$$

$$\frac{p \xrightarrow{\alpha} p' \ \wedge \ q \xrightarrow{\alpha :} \wedge \ \ bn(\alpha) \notin fn(q)}{p \| q \xrightarrow{\alpha} p' \| q} \qquad \frac{p[(rec \ A\langle \tilde{x} \rangle .p)/A, \tilde{y}/\tilde{x}] \xrightarrow{\alpha} p'}{(rec \ A\langle \tilde{x} \rangle .p)\langle \tilde{y} \rangle \xrightarrow{\alpha} p'} \qquad \frac{p \equiv p', \ p' \xrightarrow{\alpha} q', \ q' \equiv q}{p \xrightarrow{\alpha} q}$$

Table 5. Operational semantic of $b\pi$-calculus

2.3 Electoral systems

In this subsection, we present the notion of an electoral system as given by Palamidessi in [P97]. All the notions given below hold both for the π-calculus and for the $b\pi$-calculus. We also use actions of the form $\bar{a}$ or a, when the names which are send or received do not matter.

A *cluster* is a system of parallel processes $P = P_1 \parallel P_2 \parallel \ldots \parallel P_n$. A *computation* C for the cluster is a (possibly ω-infinite) sequence of transitions[1]:

$$\begin{aligned} P_1 \parallel P_2 \parallel \ldots \parallel P_n &\xrightarrow{\alpha_1} P_1^1 \parallel P_2^1 \parallel \ldots \parallel P_n^1 \\ &\xrightarrow{\alpha_2} P_1^2 \parallel P_2^2 \parallel \ldots \parallel P_n^2 \\ &\ \ \vdots \\ &\xrightarrow{\alpha_m} P_1^m \parallel P_2^m \parallel \ldots \parallel P_n^m \\ (\quad &\xrightarrow{\alpha_{m+1}} \ldots \quad) \end{aligned}$$

If $\tilde{\alpha} = \alpha_1.\alpha_2.\ldots.\alpha_m$, we will represent the computation C also by $C : P \stackrel{\tilde{\alpha}}{\Longrightarrow} P^m$ (or by $C : P \stackrel{\tilde{\alpha}}{\Longrightarrow} \ldots$ if C is infinite).

C' *extends* C if $C : P \stackrel{\tilde{\alpha}}{\Longrightarrow} P^m$, and there exists $C'' : P^m \stackrel{\tilde{\alpha}}{\Longrightarrow} P^{m+m'}$ or $C'' : P^m \stackrel{\tilde{\alpha}}{\Longrightarrow} \ldots$ such that $C' = CC''$, where the two occurrences of P^m are collapsed. The *projection* of C over P_i (C, P given as above), denoted $Proj(C, i)$, is defined as the "contribution" of P_i to the computation C.

We define the restriction of a sequence of actions $\tilde{\alpha}$ w.r.t. to a set of channels A (denoted $\tilde{\alpha}/A$) as being a word of the $(A \cup \bar{A})^*$, given as follows:

[1] As in [P97], for the sake of keeping notations simple, we suppose that each binder νx is pushed "to the top level" using repeatedly the rules for structural congruence. In addition, we do not represent explicitly the binders at top level and we suppose that the cluster will never perform a visible action on one of the names restricted by a binder

$$\tilde{\alpha}/A = \begin{cases} \epsilon & \text{if } \tilde{\alpha} = \epsilon, \\ \tilde{\beta}/A & \text{if } \tilde{\alpha} = \tau.\tilde{\beta}, \\ \tilde{\beta}/A & \text{if } \alpha = \bar{a}x.\tilde{\beta} \text{ or } \alpha = \nu x\bar{a}x.\tilde{\beta} \text{ or } \tilde{\alpha} = a(x).\tilde{\beta}, \text{ with } a \notin A, \\ \bar{a}.(\tilde{\beta}/A) & \text{if } \alpha = \bar{a}x.\tilde{\beta} \text{ or } \alpha = \nu x\bar{a}x.\tilde{\beta}, \text{ with } a \in A, \\ a.(\tilde{\beta}/A) & \text{if } \tilde{\alpha} = a(x).\tilde{\beta}, \text{ with } a \in A. \end{cases} \quad (1)$$

Like in [P97], for the definition of an electoral system, we assume that Ch_p and Ch_b contain the natural names $\mathbb{N}$, which will represent the identity of processes in a network. We shall use a definition slightly different from those used in [P97].

Let $\mathcal{C}(P) \stackrel{def}{=} \{C | C \text{ is a computation for } P\}$ be the set of all computations of P and $\mathcal{C}_1(P) \stackrel{def}{=} \{C | C \text{ is a computation for } P, \exists i, k \in \{1, \ldots, n\}, Proj(C, i) \text{ contains } \bar{k}\}$.

Definition 4. (Electoral systems)

- *A cluster $P = P_1 \parallel P_2 \parallel \ldots \parallel P_n$ is a* **(strong) electoral system** *if for every computation $C \in \mathcal{C}(P)$, there exists an extension C' of C, and there exists $k \in \{1, \ldots, n\}$ (the "leader") such that for each $i \in \{1, \ldots, n\}$ the projection $Proj(C', i)$ contains exactly one output action of the form $\bar{k}$, and any trace of a P_i may contain at most one action of the form $\bar{l}$, with $l \in \{1, \ldots, n\}$.*
- *A cluster $P = P_1 \parallel P_2 \parallel \ldots \parallel P_n$ is a* **weak electoral system** *if for every computation $C \in \mathcal{C}_1(P)$, there exists an extension C' of C, and there exists $k \in \{1, \ldots, n\}$ (the "leader") such that for each $i \in \{1, \ldots, n\}$ the projection $Proj(C', i)$ contains exactly one output action of the form $\bar{k}$, $\mathcal{C}_1(P)$ is not empty, and any trace of a P_i may contain at most one action of the form $\bar{l}$, with $l \in \{1, \ldots, n\}$.*

Note that for a strong electoral system, any infinite computation must contain already all the necessary output actions, because it cannot be strictly extended, and also, that for a strong or a weak electoral system, there exists always a finite computation which satisfies the requirements of Definition 4.

3 Encoding $b\pi$-calculus into π-calculus

In this section we prove the non-existence of an uniform encoding of the $b\pi$-calculus into π-calculus, under certain requirements on the encoding, which preserves "reasonable" semantics.

When translating a term from one calculus into another, we would like the translation be independent of the context, i.e. the encoding of t_1 in $C'[t_1]$ and in $C''[t_1]$ to be the same regardless of contexts C' and C''. This requires that the encoding is compositional. Concerning concurrent systems, it is reasonable to

require at least that the parallel construction is preserved under the encoding, and more, that is exactly mapped in the parallel constructor, i.e. that

$$[\![P \parallel Q]\!] = [\![P]\!] \parallel [\![Q]\!] \tag{2}$$

Also, it seems reasonable to require that the encoding "behaves well" with renamings, i.e.

$$[\![\sigma(P)]\!] = \sigma([\![P]\!]) \tag{3}$$

We will call *uniform* an encoding which satisfies (2) and (3) and which in addition translates outputs (inputs) in the first calculus in related outputs (inputs) in the second calculus.

Definition 5. Uniform encoding

An encoding $[\![\ ]\!] : \mathcal{P}_x \longrightarrow \mathcal{P}_y$ *(where* $(x, y) \in \{(p, b), (b, p)\}$*) is called uniform, if it satisfies the following conditions:*

1. $[\![P \parallel Q]\!] = [\![P]\!] \parallel [\![Q]\!]$,
2. $[\![\sigma(P)]\!] = \sigma([\![P]\!])$, *for any substitution* $\sigma : \mathbb{N} \longrightarrow \mathbb{N}$,
3. $(\exists \tilde{\alpha} : P \stackrel{\tilde{\alpha}}{\Longrightarrow} P'$ *if and only if* $\exists \tilde{\beta} : [\![P]\!] \stackrel{\bar{\beta}}{\Longrightarrow} [\![P']\!])$, *where* $\tilde{\alpha}/\mathbb{N} = \tilde{\beta}/\mathbb{N}$,
4. *if* $[\![P]\!] \stackrel{\bar{\beta}}{\Longrightarrow} Q$ *then* $\exists \tilde{\gamma} : [\![P]\!] \stackrel{\bar{\beta}}{\Longrightarrow} Q \stackrel{\bar{\gamma}}{\Longrightarrow} [\![P']\!]$.

Now we prove that the conditions from Definition 5 are strong enough to ensure that strong (weak) electoral systems from $b\pi$-calculus are translated into strong (weak) electoral systems from π-calculus.

Lemma 1. *Any uniform encoding translates a strong (weak) electoral system* P *from* $b\pi$*-calculus into a strong (weak) electoral system* $[\![P]\!]$ *from* π*-calculus.*

Proof. We give the proof for the case of weak electoral system. The other case follows imediatly. Let $P = P_1 \parallel P_2 \parallel \ldots \parallel P_n$ be a weak electoral system. We shall prove that $R \stackrel{def}{=} [\![P]\!]$ is a weak electoral system too.

We have

$$R \stackrel{def}{=} [\![P_1 \parallel P_2 \parallel \ldots \parallel P_n]\!] = [\![P_1]\!] \parallel [\![P_2]\!] \parallel \ldots \parallel [\![P_n]\!] = R_1 \parallel R_2 \parallel \ldots \parallel R_n$$

where $R_i \stackrel{def}{=} [\![P_i]\!]$.

Let $R_i \stackrel{\bar{\beta}^i}{\Longrightarrow} Q_i$ be an arbitrary trace of R_i. From the Condition 4 of the Definition 5 we obtain that there exists a continuation

$$R_i = [\![P_i]\!] \stackrel{\bar{\beta}^i}{\Longrightarrow} Q_i \stackrel{\bar{\gamma}^i}{\Longrightarrow} R_i' = [\![P_i']\!]$$

Then, using the Condition 3 of the Definition 5, we obtain a corresponding trace of P_i

$$P_i \stackrel{\bar{\alpha}^i}{\Longrightarrow} P_i'$$

such that $\tilde{\alpha^i}/\mathbb{N} = (\tilde{\beta^i}\tilde{\gamma^i})/\mathbb{N}$. Since P is a weak electoral system, α^i contains at most one action of the form $\bar{l}$, with $l \in \{1, \ldots, n\}$, and this is also true for the derivation $R_i \stackrel{\bar{\beta^i}}{\Longrightarrow} Q_i$.

Let

$$D : R = R_1 \parallel R_2 \parallel \ldots \parallel R_n \stackrel{\bar{\beta}}{\Longrightarrow} Q$$

be a computation from $\mathcal{C}_1([\![P]\!])$ $(\tilde{\beta}/\mathbb{N} \neq \epsilon)$. From the Condition 4 of the Definition 5 we obtain that there exists E, extension of D,

$$E : R = [\![P]\!] \stackrel{\bar{\beta}}{\Longrightarrow} Q \stackrel{\bar{\gamma}}{\Longrightarrow} R' = [\![P']\!]$$

Using the Condition 3 of the Definition 5, we obtain a corresponding computation

$$C : P \stackrel{\bar{\alpha}}{\Longrightarrow} P'$$

such that $\tilde{\alpha}/\mathbb{N} = (\tilde{\beta}\tilde{\gamma})/\mathbb{N} \neq \epsilon$ and hence $C \in \mathcal{C}_1(P)$. Since P is a weak electoral electoral system, there is an extension C' of C

$$C : P \stackrel{\bar{\alpha}}{\Longrightarrow} P' \stackrel{\bar{\alpha'}}{\Longrightarrow} P''$$

and there exists $k \in \{1, \ldots, n\}$ (the "leader") such that for each $i \in \{1, \ldots, n\}$ the projection $Proj(C', i)$ contains exactly one output action of the form $\bar{k}$. Using the Condition 3 of the Definition 5 we obtain the computation

$$D' : R = [\![P]\!] \stackrel{\bar{\beta}}{\Longrightarrow} Q \stackrel{\bar{\gamma}}{\Longrightarrow} [\![P']\!] \stackrel{\bar{\beta'}}{\Longrightarrow} [\![P'']\!]$$

such that $(\tilde{\beta}\tilde{\gamma}\tilde{\beta'})/\mathbb{N} = (\tilde{\alpha}\tilde{\alpha'})/\mathbb{N}$. Because in $\tilde{\beta}\tilde{\gamma}\tilde{\beta'}$ there are n outputs of the form $\bar{k}$, with $k \in \{1, \ldots, n\}$, and since every component R_i can make at most one such action, it follows that for each $i \in \{1, \ldots, n\}$ the projection $Proj(D', i)$ contains exactly one output action of the form $\bar{k}$.

Since P is a weak electoral system, there exists a computation

$$C : P \stackrel{\bar{\alpha}}{\Longrightarrow} P'$$

such that $C \in \mathcal{C}_1(P)$ $(\tilde{\alpha}/\mathbb{N} \neq \epsilon)$.

Using the Condition 3 of the Definition 5, we obtain a corresponding computation

$$D : R = [\![P]\!] \stackrel{\bar{\beta}}{\Longrightarrow} [\![P']\!]$$

such that $\tilde{\beta}/\mathbb{N} = \tilde{\alpha}/\mathbb{N} \neq \epsilon$, and it follows that $\mathcal{C}_1([\![P]\!]) \neq \emptyset$. □

Lemma 2. *Let* $P_i = \bar{a}i \parallel a(x).\bar{x}$.

For all $n \geq 2$, $P(n) = \prod_{i=1}^{n} P_i$ *is a weak electoral system.*

Proof. If

$$C : P(n) = P_1 \parallel P_2 \parallel \ldots \parallel P_n \stackrel{\alpha}{\longrightarrow} P_1^1 \parallel P_2^1 \parallel \ldots \parallel P_n^1 = P^1(n)$$

in a step, then there is $k \in \{1, \ldots, n\}$ such that $\alpha = \bar{a}k$ or there is b such that $\alpha = ab$. For the first case, we can extend C to C':

$$C' : P(n) = P_1 \parallel \ldots \parallel P_k \parallel \ldots \parallel P_n \xrightarrow{\bar{a}\,k} \bar{a}1 \parallel \bar{k} \parallel \ldots \parallel nil \parallel \bar{k} \parallel \ldots \parallel \bar{a}n \parallel \bar{k}$$

$$\xrightarrow{\bar{k}} \bar{a}1 \parallel nil \parallel \ldots \parallel nil \parallel \bar{k} \parallel \ldots \parallel \bar{a}n \parallel \bar{k}$$

$$\vdots$$

$$\xrightarrow{\bar{k}} \bar{a}1 \parallel nil \parallel \ldots \parallel nil \parallel nil \parallel \ldots \parallel \bar{a}n \parallel nil$$

If $\alpha = ab$ and $k \in \{1, \ldots, n\}$ we proceed similarly, if not, the computation $C \notin \mathcal{C}_1(P)$ and we are done.□

Then, the main result of this paper is:

Theorem 1. *There exists no uniform encoding of the $b\pi$-calculus into π-calculus.*

Proof. Let suppose that there exists an uniform encoding $[\![\bullet]\!]$ of $b\pi$-calculus into π-calculus.

Let us denote $R_i \stackrel{def}{=} [\![P_i]\!]$, and $R(n) \stackrel{def}{=} R_1 \parallel R_2 \parallel \ldots \parallel R_n$, where P_i are the same as in the Lemma 2. We have

$$R(n) = R_1 \parallel R_2 \parallel \ldots \parallel R_n = [\![P_1]\!] \parallel [\![P_2]\!] \parallel \ldots \parallel [\![P_n]\!] =$$

$$[\![P_1 \parallel P_2 \parallel \ldots \parallel P_n]\!] = [\![P(n)]\!]$$

Since $\forall n \geq 2, P(n)$ is a weak electoral system, using the Lemma 1, we obtain that $\forall n \geq 2, R(n)$ is a weak electoral system too.

Let $m, m' \geq 2$, and let the renaming $\sigma : \mathbb{N} \longrightarrow \mathbb{N}, \sigma(i) = i + m'$, $\forall i \in \{1, \ldots, m\}$ and identity otherwise.

We have

$$R(m + m') = R_1 \parallel R_2 \parallel \ldots \parallel R_{m'} \parallel R_{m'+1} \parallel \ldots \parallel R_{m'+m} =$$

$$R_1 \parallel R_2 \parallel \ldots \parallel R_{m'} \parallel [\![P_{m'+1}]\!] \parallel \ldots \parallel [\![P_{m'+m}]\!] =$$

$$R_1 \parallel R_2 \parallel \ldots \parallel R_{m'} \parallel [\![\sigma(P_1)]\!] \parallel \ldots \parallel [\![\sigma(P_m)]\!] =$$

$$R_1 \parallel R_2 \parallel \ldots \parallel R_{m'} \parallel \sigma([\![P_1]\!]) \parallel \ldots \parallel \sigma([\![P_m]\!]) =$$

$$R_1 \parallel R_2 \parallel \ldots \parallel R_{m'} \parallel \sigma(R_1) \parallel \ldots \parallel \sigma(R_m) =$$

$$R(m') \parallel \sigma(R(m))$$

Since $R(m')$ is a weak electoral system, then there exists a computation $C_1 : R(m') \stackrel{\bar{\alpha}}{\Longrightarrow} R^p(m')$ and a $k \in \{1, \ldots, m'\}$, such that for each $i \in \{1, \ldots, m'\}$ the projection $Proj(C_1, i)$ contains exactly one output action of the form $\bar{k}$. Similar, because $R(m)$ is a weak electoral system, then there exists a computation $C_2 : R(m) \stackrel{\bar{\beta}}{\Longrightarrow} R^q(m)$ and a $k' \in \{1, \ldots, m\}$, such that for each $i \in \{1, \ldots, m\}$ the projection $Proj(C_2, i)$ contains exactly one output action of the form $\bar{k'}$.

Hence, there exists a computation $C_3 : \sigma(R(m)) \stackrel{\sigma(\bar{\beta})}{\Longrightarrow} \sigma(R^q(m))$ of $\sigma(R(m))$ and a $k' \in \{1, \ldots, m\}$, such that for each $i \in \{m'+1, \ldots, m'+m\}$ the projection $Proj(C_3, i)$ contains exactly one output action of the form $\overline{\sigma(k')}$.

Then we have the following computation

$$C_4 : R(m+m') = R(m') \parallel \sigma(R(m)) \stackrel{\bar{\alpha}}{\Longrightarrow} R^p(m') \parallel \sigma(R(m))$$

$$\stackrel{\sigma(\bar{\beta})}{\Longrightarrow} R^p(m') \parallel \sigma(R^q(m))$$

such that for each $i \in \{1, \ldots, m'\}$ the projection $Proj(C_4, i)$ contains exactly one output action of the form $\bar{k}$, and for each $i \in \{m'+1, \ldots, m'+m\}$ the projection $Proj(C_4, i)$ contains exactly one output action of the form $\overline{\sigma(k')}$. Since $\sigma(k') \in \{m'+1, \ldots, m'+m\}$, we have that $k \neq \sigma(k')$, and C_4 cannot be extended to a computation C as in the Definition 4, hence $R(m+m')$ cannot be a weak electoral system. □

If we call $b\pi_a$-calculus the asynchronous variant of $b\pi$-calculus (obtained from $b\pi$-calculus by forbidding output as prefix, asynchronous variant of π-calculus being obtained similarly from π-calculus), we obtain the following result:

Corollary 1. *There exists no uniform encoding of the $b\pi_a$-calculus into π-calculus.*

Remark. Note that the condition 2 is essential to establish our result. We do not claim that there is no encoding at all; hence we cannot guarantee on the non-existence of a slighter encoding, which remains compositional, mapping $[\![P \parallel Q]\!]$ onto $C[[\![P]\!], [\![Q]\!]]$.

4 Conclusion and related work

The Theorem 1 corresponds to the separability result between synchronous and asynchronous π-calculus obtained by Palamidessi. She uses in her proof the fact that asynchronous π-calculus enjoys a certain kind of confluence. Hence, there are symmetric electoral systems in the π-calculus, whereas this does not hold in the asynchronous case, since whenever a first action occurs, all the other processes can execute their corresponding output action as well, and so on, in this way generating an infinite computation which never makes outputs on a special channel o (used for sending to the environment the leader).

In our paper we exploit another difference between the two calculus: while π-calculus enjoys an interleaving semantic (if $P \stackrel{\bar{\alpha}}{\Longrightarrow} P'$ and $Q \stackrel{\bar{\beta}}{\Longrightarrow} Q'$, then $P \parallel Q \stackrel{\bar{\alpha}}{\Longrightarrow} P' \parallel Q \stackrel{\bar{\beta}}{\Longrightarrow} P' \parallel Q'$), this does not hold for the $b\pi$-calculus ($P \stackrel{\bar{\alpha}}{\Longrightarrow} P'$ does not imply $P \parallel Q \stackrel{\bar{\alpha}}{\Longrightarrow} P' \parallel Q$).

The problem of encoding a broadcast calculus into π-calculus or CCS ([M89]) was already stated in [H93]. Holmer gives an encoding of CBS into $SCCS$ and he makes the conjecture that it is not possible to find a compositional translation from broadcast calculus to CCS: "CCS is << *too asynchronous* >>

to interpret CBS in". His variant of broadcast calculus (CBS) is without value-passing. We can give a partial answer to his conjecture, by proving that value-passing CBS cannot be uniformly encoded in π-calculus and that CBS without value-passing cannot be uniformly encoded in CCS:

Proposition 1. *There exists no uniform encoding of value-passing CBS into π-calculus.*

Hint of the proof. In newer variants of CBS, choice is also denoted by &, and outputs (inputs) are denoted differently ($v!P$ means a process which says v, and become P, while $x?Q$ is a process which listen, and once he heard v, it evolves to $Q[v/x]$).

Let $P_i = i! \ \& \ x?x!$. Then it is easy to remark that for all $n \geq 2$, $P(n) = \prod_{i=1}^{n} P_i$ is a weak electoral system and using arguments similar to Theorem 1, we obtain the result. □

Proposition 2. *There exists no uniform encoding of CBS without value-passing into CCS.*

Hint of the proof. In CBS without value-passing, processes communicate by exchanging signals (by synchronisations). $v!P$ is a process which says v, and become P, while $v?Q$ is a process which listen, and once it heards v, it evolves to Q.

Let $P_i = a_i! \parallel \sum_{j=1}^{n} a_j?j!$. Then it is easy to remark that $P(n) = \prod_{i=1}^{n} P_i$ is a symmetric strong electoral system (for definition see Definition 3.1 and Definition 3.2 from [P97]).

Combining this with a stronger version of the Theorem 5.2 from [P97] (which admits a similar proof for our definition of Electoral System), we obtain the result. □

Acknowledgements

We have gratefully appreciated the useful comments received from Catuscia Palamidessi and Gianluigi Zavattaro. The referees have also contributed by their remarks to improve the presentation of the paper.

References

[EM99] C. Ene and T. Muntean: A distributed calculus for nomadic processes. submitted, 1999.

[HR95] M. Hennessy and J. Rathke: Bisimulations for a Calculus of Broadcasting Systems. In I. Lee and S. Smolka, editors, *Proceedings of CONCUR 95*, Philadelphia, (1995) Lecture Notes in Computer Science **962**, Springer-Verlag 486–500.

[H85] C. A. R. Hoare: Communicating Sequential Processes. Prentice-Hall, 1985.

[H93] U. Holmer: Interpreting Broadcast Communication in SCCS. In *Proceedings of CONCUR 93*, (1993) Lecture Notes in Computer Science **715**, Springer-Verlag.

[M89] R. Milner: Communication and concurrency. Prentice-Hall, 1989.

[M91] R. Milner: The Polyadic π-Calculus: A Tutorial. In Friedrich L. Bauer and Wilfried Brauer and Helmut Schwichtenberg, editors, (1995) *Logic and Algebra of Specification* **94**. Available as Technical Report ECS-LFCS-91-180, University of Edinburgh, October 1991.

[MPW92] R. Milner and J. Parrow and D. Walker: A Calculus of Mobile Processes, Part I/II. Journal of Information and Computation **100** (1992) 1–77.

[P97] Catuscia Palamidessi: Comparing the Expressive Power of the Synchronous the Asynchronous π-calculus. In *Proceedings of POPL 1997* (ACM, Jan. 1997) 256–265.

[P95] K. V. S. Prasad: A Calculus of Broadcasting Systems. (1995) Science of Computer Programming **25**.