Electronic Notes in Theoretical Computer Science 194 (2008) 59–84 

[www.elsevier.com/locate/entcs](http://www.elsevier.com/locate/entcs)

Linearity, Persistence and Testing Semantics in the Asynchronous Pi-Calculus

# Diletta Cacciagrano,[2](#_bookmark0) Flavio Corradini[3](#_bookmark0)

*Dipartimento di Matematica e Informatica Universita` degli Studi di Camerino, Italy*

Jesu´s Aranda [1](#_bookmark0) *,*[4](#_bookmark0)

*INRIA Futurs, LIX E´cole Polytechnique, France*

*Escuela de Ingenier´ıa de Sistemas y Computacio´n, Universidad del Valle, Colombia*

# Frank D. Valencia[5](#_bookmark0)

*CNRS and LIX E´cole Polytechnique, France*

Abstract

In [[24](#_bookmark42)] the authors studied the expressiveness of persistence in the asynchronous *π*-calculus (A*π*) wrt weak barbed congruence. The study is incomplete because it ignores the issue of divergence. In this paper, we present an expressiveness study of persistence in the asynchronous *π*-calculus (A*π*) wrt De Nicola and Hennessy’s testing scenario which is sensitive to divergence. Following [[24](#_bookmark42)], we consider A*π* and three sub-languages of it, each capturing one source of persistence: the *persistent-input* calculus (PIA*π*), the *persistent-output* calculus (POA*π*) and *persistent* calculus (PA*π*). In [[24](#_bookmark42)] the authors showed encodings from A*π* into the semi-persistent calculi (i.e., POA*π* and PIA*π*) correct wrt weak barbed congruence. In this paper we prove that, under some general conditions, there cannot be an encoding from A*π* into a (semi)-persistent calculus preserving the *must* testing semantics.

*Keywords:* Asynchronous Pi-Calculus, Linearity, Persistence, Testing Semantics.

1 The work of Jesu´s Aranda has been supported by COLCIENCIAS (Instituto Colombiano para el Desar- rollo de la Ciencia y la Tecnolog´ıa ”Francisco Jos´e de Caldas”) and INRIA Futurs.

2 Email:[diletta.cacciagrano@unicam.it](mailto:diletta.cacciagrano@unicam.it)

3 Email:[flavio.corradini@unicam.it](mailto:flavio.corradini@unicam.it)

4 Email:[jesus.aranda@lix.polytechnique.fr](mailto:jesus.aranda@lix.polytechnique.fr)

5 Email:[frank.valencia@lix.polytechnique.fr](mailto:frank.valencia@lix.polytechnique.fr)

1571-0661 © 2008 Published by Elsevier B.V. Open access under [CC BY-NC-ND license](http://creativecommons.org/licenses/by-nc-nd/3.0/).

doi:10.1016/j.entcs.2007.11.006

In [[24](#_bookmark42)] the authors present an expressiveness study of linearity and persistence of processes. Since several calculi presuppose persistence on their processes, the authors address the expressiveness issue of whether such persistence restricts the systems that we can specify, model or reason about in the framework. Their work is conducted using the standard notion of weak barbed congruence and hence it ignores divergence issues. Since divergence plays an important role in expressiveness studies, particularly in those studies involving persistence, in this work we aim at extending and strengthening their study by using the standard notion of testing equivalences. As elaborated below, our technical results contrast and complement those in [[24](#_bookmark42)]. More importantly, our results also clarify and support informal expressiveness claims in the literature.

Motivation: *Linearity* is present in process calculi such as CCS, CSP, the *π*- calculus [[20](#_bookmark39)] and Linear CCP [[31](#_bookmark49),[14](#_bookmark34)], where messages are consumed upon being received. In the *π*-calculus the system *x*¯*z* | *x*(*y*)*.P* | *x*(*y*)*.Q* represents a message with a datum *z*, tagged with *x*, that can be *consumed* by either *x*(*y*)*.P* or *x*(*y*)*.Q*. *Persistence of messages* is present in several process calculi. Perhaps the most prominent representative of such calculi is Concurrent Constraint Programming (CCP) [[32](#_bookmark50)]. Here the messages (or items of information) can be read but, unlike in Linear CCP, they cannot be consumed. Other prominent examples can be found in the context of calculi for analyzing and describing security protocols: Crazzolara and Winskel’s SPL [[12](#_bookmark32)], the Spi Calculus variants by Fiore and Abadi [[15](#_bookmark35)] and by Amadio et all [[2](#_bookmark21)], and the calculus of Boreale and Buscemi [[5](#_bookmark25)] are operationally defined in terms of configurations containing messages which cannot be consumed. *Persistent receivers* arise, e.g. in the notion of *omega receptiveness* [[29](#_bookmark47)], where the input of a name is always available—but always with the same continuation. In the *π*-calculus persistent receivers are used, for instance, to model functions, objects, higher-order communications, or procedure definitions. Furthermore, persistence of *both* messages and receivers arise in the context of CCP with universally-quantified persistent ask operations. In the context of calculi for security, persistent receivers can be used to specify protocols where principals are willing to run an unbounded number of times (and persistent messages to model the fact that every message can be remembered by the spy). In fact, the approach of specifying protocols in a persistent setting, with an unbounded number of sessions, has been explored in [[4](#_bookmark23)] by using a classic logic Horn clause representation of protocols (rather than a linear logic one).

Expressiveness of Persistence - Drawbacks and Conjectures: The study in

[[24](#_bookmark42)] is conducted in the *asynchronous π*-calculus (A*π*), which naturally captures the persistent features mentioned above. Persistent messages (and receivers) can simply be specified using the *replication* operator of the calculus which creates an unbounded number of copies of a given process. In particular, the authors in [[24](#_bookmark42)] investigate the existence of encodings from A*π* into three sub-languages of it, each capturing one source of persistence: the *persistent-input* calculus (PIA*π*), defined as

A*π* where inputs are replicated; *persistent-output* calculus (POA*π*), defined dually,

i.e. outputs rather than inputs are replicated; *persistent* calculus (PA*π*), defined as A*π* but with all inputs and outputs are replicated. The main result basically states that we need one source of linearity, i.e. either on inputs (PIA*π*) or outputs (POA*π*) to encode the behavior of arbitrary A*π* processes via weak barbed congruence. Nevertheless, the main drawback of the work [[24](#_bookmark42)] is that the notion of correct encoding is based on weak barbed bisimulation (congruence), which is not sensitive to *divergence*. In particular, the encoding provided in [[24](#_bookmark42)] from A*π* into PIA*π* is weak barbed congruent preserving but not divergence preserving. Although in some situations divergence may be ignored, in general it is an important issue to consider in the correctness of encodings [[8](#_bookmark27),[17](#_bookmark37),[16](#_bookmark36),[18](#_bookmark38),[7](#_bookmark28)].

In fact, the informal claims of extra expressivity of Linear CCP over CCP in [[3](#_bookmark24),[14](#_bookmark34)] are based on discrimination introduced by divergence that is clearly ignored by the standard notion of weak bisimulation. Furthermore, the author of [[11](#_bookmark29)] suggests as future work to extend SPL, which uses only persistent messages and replication, with recursive definitions to be able to program and model recursive protocols such as those in [[1](#_bookmark22),[25](#_bookmark43)]. Nevertheless, one can give an encoding of recursion in SPL from an easy adaptation of the composition between the A*π* encoding of recursion [[30](#_bookmark48)] (where recursive calls are translated into *linear* A*π* outputs and recursive definitions into persistent inputs) and the encoding of A*π* into POA*π* in [[24](#_bookmark42)]. The resulting encoding is correct up-to weak bisimulation. The encoding of A*π* into POA*π*, how- ever, introduces divergence and hence the composite encoding does not seem to invalidate the justification for extending SPL with recursive definitions. The above works suggest that the expressiveness study of persistence is relevant but incomplete if divergence is not taken into account.

This work: In this paper we shall therefore study the existence of encodings from A*π* into the persistent sub-languages mentioned above using testing semantics [[13](#_bookmark33)]. Our main contribution is to provide a uniform and general result stating that, under some reasonable conditions, A*π* cannot be encoded into any of the above (semi-) persistent calculi while preserving the *must* testing semantics. The general conditions involve compositionality on the encoding of constructors such as parallel composition, prefix, and replication. The main result contrasts and completes the ones in [[24](#_bookmark42)]. It also supports the informal claims of extra expressivity mentioned above. We shall also state other more specialized impossibility results for must preserving encodings from A*π* into the semi-persistent calculi, focusing on specific properties of each target calculus. This helps clarifying some previous assumptions on the interplay between syntax and semantics in encodings of process calculi. We believe that, since the study is conducted in A*π* with well-established notions of equivalence, we can easily adapt our results to other asynchronous frameworks such

as CCP languages and the above-mentioned calculi for security.

# The calculi

Here we define the calculi we study. We first recall the (monadic) *asynchronous*

*π-calculus* (A*π*). The other calculi are defined as syntactic restrictions of A*π*.

* 1. *The asynchronous pi-calculus*

Let N (ranged over by *x, y, z,.. .*) be a set of names. The set of the asynchronous *π*-calculus processes (ranged over by *P* , *Q*, *R .. .*) is generated by the following grammar:

*P, Q,...* ::= 0

*x*¯*z*

*x*(*y*)*.P*

*P* | *Q*

(*νx*)*P* ! *P*

Intuitively, an *output x*¯*z* represents a message *z* tagged with a name *x* indicating that it can be received (or *consumed* ) by an *input process x*(*y*)*.P* which behaves, upon receiving *z*, as *P* {*z/y*}. Furthermore, *x*(*y*)*.P* binds the names *y* in *P* . The other binder is the *restriction* (*νx*)*P* which declares a name *x* private to *P* . The *parallel composition P* | *Q* means *P* and *Q* running in parallel. The *replication* !*P* means *P* |*P* | *.. .*, i.e., !*P* represents a *persistent resource*.

We use the standard notations *bn*(*Q*) for the *bound names* in *Q*, and *fn*(*Q*) for the *free names* in *Q*. The set of names of *P* is defined as *n*(*P* ) = *fn*(*P* ) ∪ *bn*(*P* ). We let *σ, ϑ...* range over (non-capturing) substitutions of names on processes.

The *reduction* relation −→ is the least binary relation on processes satisfying

the rules in Table [1](#_bookmark1). −∗→ denotes the reflexive, transitive closure of −→ . The

reductions are quotiented by the *structural congruence* relation ≡.

Definition 2.1 [Structural congruence] Let ≡ be the smallest congruence over pro- cesses satisfying *α*-equivalence, the commutative monoid laws for composition with 0 as identity, the replication law !*P* ≡ *P* | !*P* , the restriction laws (*νx*)0 ≡ 0, (*νx*)(*νy*)*P* ≡ (*νy*)(*νx*)*P* and the extrusion law: (*νx*)(*P* | *Q*) ≡ *P* | (*νx*)*Q* if *x* /∈ *fn*(*P* ).

Com *x*¯*z* | *x*(*y*)*.P* −→

*P* −→ *P* '

Res

*P* | *Q* −→ *P* ' | *Q*

*P* {*z/y*}

*P* −→

*P* '

Par

(*νx*)*P* −→ (*νx*)*P* '

*P* ≡ *P* ' *P* ' −→ *Q*' *Q*' ≡ *Q*

Cong

*P* −→ *Q*

Table 1 Reduction Rules.

* 1. *The (semi-)persistent calculi*

The *persistent-input* calculus PIA*π* results from A*π* by requiring all input processes to be replicated. Processes in PIA*π* are generated by the following grammar:

*P, Q,...* ::= 0

! *x*(*y*)*.P*

*x*¯*y*

*P* | *Q*

(*νx*)*P* ! *P*

The *persistent-output* calculus POA*π* arises as from A*π* by requiring all outputs to be replicated. Processes in POA*π* are generated by the following grammar:

*P, Q,...* ::= 0

*x*(*y*)*.P*

! *x*¯*y*

*P* | *Q*

(*νx*)*P* ! *P*

Finally, we have the *persistent* calculus PA*π*, a subset of A*π* where output and input processes must be replicated. Processes in PA*π* are generated by the following grammar:

*P, Q,...* ::= 0

! *x*(*y*)*.P*

! *x*¯*y*

*P* | *Q*

(*νx*)*P* ! *P*

The relation −→ for PIA*π*, POA*π* and PA*π* can be equivalently defined as in Table [1](#_bookmark1), with Com replaced respectively with Com(PIA*π*), Com(POA*π*) and Com(PA*π*) rules (Table [2](#_bookmark2)). The new rules reflect the *persistent-input* and *linear- output* nature of PIA*π* (Rule Com(PIA*π*)), the *linear-input* and *persistent-output* nature of POA*π* (Rule Com(POA*π*)), and the *persistent* nature of PA*π* (Rule Com(PA*π*)).

Com(PIA*π*) *x*¯*z* | ! *x*(*y*)*.P* −→ *P* {*z/y*}| ! *x*(*y*)*.P*

Com(POA*π*) ! *x*¯*z* | *x*(*y*)*.P* −→ ! *x*¯*z* | *P* {*z/y*}

Com(PA*π*) ! *x*¯*z* | ! *x*(*y*)*.P* −→ *P* {*z/y*}| ! *x*¯*z* | ! *x*(*y*)*.P*

Table 2 Reduction Rules.

Notation 2.1 *We shall use* P *to range over the set of the calculi so-far deﬁned*

{*Aπ, PIAπ, POAπ, PAπ*}*.*

# Testing semantics

In [[13](#_bookmark33)] De Nicola and Hennessy propose a framework for defining pre-orders that is widely acknowledged as a realistic scenario for system testing. It means to de- fine formally when one process is a correct implementation of another considering specially unsafe contexts, in which is particularly important what is the revealed information of the process in any context or test. In this section we summarize the basic definitions behind the testing machinery for the *π*-calculi.

Definition 3.1 [Observers]

- The set of names N is extended as N ' = N ∪ {*ω*} with *ω* /∈ N . By convention we let *fn*(*ω*)= {*ω*} and *bn*(*ω*)= ∅ (*ω* is used to report success).

* The set O (ranged over by *o, o*'*, o*''*, E, E*'*,.. .*) of observers (tests) is defined like

P, where the grammar is extended with the production *P* ::= *ω.P* .

* −*ω*→ is the least predicate over O satisfying the inference rules in Table [3](#_bookmark3).

Omega *ω.E* −*ω*→

*E* −*ω*→

Res

(*νy*)*E* −*ω*→

*E* −*ω*→ *E*' −*ω*→ *E*' ≡ *E*

Par

1

*E* | *E* −*ω*→

Cong

1 2

*E* −*ω*→

Table 3ω Predicate −→ .

Definition 3.2 [Maximal computations] Given *P* ∈ P and *o* ∈ O, a maximal computation from *P* | *o* is either an infinite sequence of the form

*P* | *o* = *E*0 −→ *E*1 −→ *E*2 −→ *...*

or a finite sequence of the form

*P* | *o* = *E*0 −→ *E*1 −→ *...* −→ *En* −→ *.*

Definition 3.3 [May, must and fair relations [6](#_bookmark6) ] Given *P* ∈P and *o* ∈ O, define:

* *P may o* if and only if *there is* a maximal computation (as in Def. [3.2](#_bookmark4)) such that

*Ei* −*ω*→, for some *i* ≥ 0;

* *P must o* if and only if *for every* maximal computation (as in Def. [3.2](#_bookmark4)) there exists

*i* ≥ 0 such that *Ei* −*ω*→;

* *P fair o* [[6](#_bookmark30)] if and only if *for every* maximal computation (as in Def. [3.2](#_bookmark4)) and

∀*i* ≥ 0, ∃ *E*' such that *Ei* −∗→ *E*' and *E*' −*ω*→.

*i i* *i*

# Encoding linearity into persistence

First, we recall some notions about encodings. An encoding is a mapping from the terms of a calculus into the terms of another. In general a “good” encoding satisfies some additional requirements, but there is no agreement on a general notion of “good” encoding. Perhaps indeed there should not be a unique notion, but several, depending on the purpose. In this paper we shall study the existence of encodings **[**·]]: A*π* →P from *π* into P ∈ {PA*π,* PIA*π,* POA*π*} and focus on typical requirements such as compositionality w.r.t. certain operators, and the correctness

w.r.t. a given semantics.

6 It may be possible to give other equivalent definitions not based on maximal computations by using

properties of the calculi under consideration such as: if *P* −→ and *P* −→ *P* '

ω

then *P*

' ω

−→. For uni-

formity, however, we have used a well-known testing semantics definition based on the notion of *maximal*

computations.

Compositionality and multi-hole contexts: We shall use notion of (multi-hole) process *contexts* [[30](#_bookmark48)] to describe compositionality. Recall that a P context *C* with *k* holes is a term with occurrences of *k* distinct *holes* [ ]1*,... ,* [ ]*k* such that a P process must result from *C* if we replace all the occurrences of each [ ]*i* with a P process. The context *C* is *singularly-structured* if each hole occurs exactly once. For example, [ ]1 | *x*(*y*)*.*([ ]2 | [ ]1) is an A*π* non singularly-structured context with two holes. Given *P*1*,... , Pk* ∈P and a context *C* with *k* holes, *C*[*P*1*,... , Pk*] is the process that results from replacing the occurrences of each [ ]*i* with *Pi*. The names of a context *C* with *k* holes, *n*(*C*), are those of *C*[*Q*1*,... , Qk*] where each *Qi* is 0. The free and bound names of a context are defined analogously. We can regard the input prefix *x*(*y*), | and ! as the operators of arity 1, 2 and 1 respectively in A*π* in the obvious sense.

Definition 4.1 [Compositionality w.r.t. an operator] Let *op* be an *n*-ary operator of A*π*. An encoding **[**·]] : A*π* →P is *compositional* w.r.t. *op* iff there is a P context *Cop* with *n* holes such that **[***op*(*P*1*, .., Pn*)]] = *Cop*[[[*P*1]]*, ..,* [[*Pn*]]]*.*

In the following, *C*[·] denotes contexts with one hole and *C*[·*,* ·] contexts with two holes. Furthermore, given an encoding **[**·]] : A*π* → P, we define *Cop*[[·]] as the context *C* such that **[***op*(*P*1*,... , Pn*)]] = *C*[[[*P*1]]*,... ,* [[*Pn*]]]. We shall often omit the “[[·]]” in *Cop*[[·]] since it is easy to infer from the context.

Remark 4.2 [Homomorphism wrt parallel composition] An interesting case of compositionality is *homomorphism* w.r.t a given operator *op*: The operator is mapped into the same operator of the target language, i.e. **[***op*(*P*1*, .., Pn*)]]

= *op*([[*P*1]]*,... ,* [[*Pn*]])*.* Homomorphism w.r.t parallelism, also called distribution- preserving [[33](#_bookmark51),[26](#_bookmark44),[27](#_bookmark45)], can arguably be considered as a reasonable requirement for an encoding. In particular, the works [[33](#_bookmark51),[26](#_bookmark44),[27](#_bookmark45),[23](#_bookmark41),[9](#_bookmark31),[16](#_bookmark36),[17](#_bookmark37)] support the distribution- preserving hypothesis by arguing that it corresponds to requiring that the degree of distribution of the processes is maintained by the translation, i.e. no coordinator is added. Some of these works are in the context of solving electoral problems and some others in more general scenarios [[16](#_bookmark36),[17](#_bookmark37)]. Other works [[22](#_bookmark40),[28](#_bookmark46)], however, argue that the requirement can be quite demanding as it rules out practical implemen- tation of distributed systems. Some of our impossibility results will appeal to the distribution-preserving hypothesis.

Remark 4.3 Typically, the *Cop* mentioned in Definition [4.1](#_bookmark7) is a singularly- structured multi-hole context in encodings of operators such as input prefix, parallel composition and replication. Note that, if the encoding is homomorphic wrt *op*, then *Cop* is a singularly-structured multi-hole context.

Correctness wrt testing: Concerning semantic correctness, we consider preser- vation of *sat* testing, where *sat* can be respectively *may , must* and *fair* . Given an encoding *e* = [[·]]: A*π* → P, we assume that its lifted version *e*' from the set of observers of *π* to the ones of P is an encoding satisfying the following: *e*'(*o*)= *e*(*o*), in the case *o* has no occurrences of *ω*.

Definition 4.4 [Soundness, completeness and *sat*-preservation] Let **[**·]] : A*π* → P. We say that **[**·]] is:

* *sound w.r.t. sat* iff ∀ *P* ∈ A*π,* ∀ *o* ∈ O, [[*P* ]] *sat* [[*o* **]** implies *P sat o*;
* *complete w.r.t. sat* iff ∀ *P* ∈ A*π,* ∀ *o* ∈ O, *P sat o* implies **[***P* ]] *sat* [[*o*]];
* *sat*-*preserving* iff **[**·]] is *sound* and *complete* w.r.t. *sat*.
  1. *Some encodings from asynchronous pi-calculus into its semi-persistent subsets*

We consider the following encoding from A*π* to PIA*π*, defined in [[24](#_bookmark42)].

Definition 4.5 The encoding **[**·]] : A*π* → PIA*π* is a homomorphism for 0, parallel composition, restriction and replication, otherwise is defined as

- **[***x*¯*z*]] = *x*¯*z*, and

- **[***x*(*y*)*.P* ]] = (*νtf* )(*t*¯| !*x*(*y*)*.*(*νl*)(¯*l* | !*t.*!*l.*([[*P* ]] | !*f*¯) | !*f.*!*l.x*¯*y*))

where *t, f, l* /∈ *fn*(*P* ) ∪{*x, y*}*.* (The lifted version is given adding **[***ω.P* ]] = *ω.*[[*P* ]].)

This encoding enjoys a strong property: namely, for any *P,* [[*P* ]] ≈ *P* , where

≈ denotes weak barbed congruence [[30](#_bookmark48)]. This implies, in the testing scenario, a property stronger than *sat*-preservation.

Proposition 4.6 *Let* [[·]] : *Aπ* → *PIAπ as in Deﬁnition* [*4.5*](#_bookmark10)*.* ∀ *P* ∈ A*π,* ∀ *o* ∈O ⊆

PIA*π P sat o* iff **[***P* ]] *sat o*, where *sat* can be respectively *may* and *fair* .

To prove that the statement does not hold in the case of *must* semantics, consider *P* = (*a.*0 |!*a*¯) and *o* = *a.ω.*0: then *P must o* but **[***P* ]] *must o*.

Extending the notion of barb to *ω*, clearly *P* | *o* ≈ [[*P* | *o*]] as *P* | *o* ∈ A*π* and, by homomorphism w.r.t parallel composition, we obtain that *P* | *o* ≈ [[*P* ]] | [[*o*]]. This is enough to hold fair- and may-preserving.

In [[24](#_bookmark42)] the encoding in Definition [4.5](#_bookmark10) is used to get an encoding of A*π* into POA*π*, by composing it with the following mapping from PIA*π* into POA*π*.

Definition 4.7 The encoding *f* = [[·]]: PIA*π* → POA*π* is a homomorphism for 0, parallel composition, restriction, and replication, otherwise is defined as

- **[***x*¯*z*]] = (*νs*)(!*x*¯*s* | *s*(*r*)*.*!*r*¯*z*), and

- **[**!*x*(*y*)*.P* ]] =!*x*(*s*)*.*(*νr*)(!*s*¯*r* | *r*(*y*)*.*[[*P* ]])

where *s, r* /∈ *fn*(*P* ) ∪ {*x, z*}. (The lifted version is given adding **[***ω.P* ]] = *ω.*[[*P* ]].)

Let *g* be **[**·]] : A*π* → PIA*π* in Definition [4.5](#_bookmark10). The encoding *h* = [[·]] : A*π* → POA*π* is the composite function *f* ◦ *g*.

Because of this encoding maps a linear output into a replicated one with the same barb, the composite encoding *h* = [[·]] : A*π* → POA*π* in Definition [4.7](#_bookmark12) does not satisfy **[***P* ]] ≈ *P* . It has a weaker property: namely, *P* ≈ *Q* iff **[***P* ]] ≈ POA*π* [[*Q*]],

[·]

where **[***P* ]] ≈ POA*π* [[*Q*]] means that ∀*C* context in A*π*, [[*C*]][[[*P* ]]] and [[*C*]][[[*Q*]]] (as-

[·]

suming [[[ ]]] = [ ]) are weak barbed bisimilar [[30](#_bookmark48)]. Similarly, the results for the

composite encoding from A*π* into POA*π* in a testing scenario are weaker than these ones for the encoding from A*π* into PIA*π*. Obviously, the following proposition would not hold if *sat* were *must* . Consider *P* =!*a*¯ and *o* = *a.ω.*0: then *P must o* but **[***P* ]] *must* [[*o*]].

Proposition 4.8 Let *h* = [[·]] : A*π* → POA*π* as in Definition [4.7](#_bookmark12). 6*P* ∈ A*π,* 6*o* ∈ 0,

*P sat o* if and only if **[***P* ]] *sat* [[*o*]], where *sat* can be respectively *may* and *fair* .

# Uniform impossibility results for persistence

This section is the core of the paper and it focuses on general and uniform negative results for encodings of A*π* into PIA*π,* POA*π* and PA*π*, respectively. We identify some reasonable conditions which will guarantee that none of these encodings can be must-preserving. In particular, we show that there does not exist a must-preserving compositional encoding, homomorphic wrt replication, from *π*-calculus into any semi-persistent calculus. The proofs mainly rely on the following statement: if **[**·]] is an encoding from A*π* into У satisfying (1) compositionality w.r.t. input prefix, (2)

must-preservation and (3) **[***ω.*0]] —*ω*→ then 6*x, y* ∈ U , any hole is prefixed in *C*[[·]] .

*x*(*y*)

We believe that the hypothesis **[***ω.*0]] —*ω*→ is reasonable for an encoding. It can follow from the existence of a divergent process in the range of the encoding, which is necessary if the encoding preserves divergence—recall that *P diverges*, *P* †, if there is an infinite sequence of reductions from *P* . However, the hypothesis **[***ω.*0]] —*ω*→ can be also obtained in a purely syntactic way, i.e without divergence assumption, defining **[***ω.P* ]] = *ω.*[[*P* ]].

Theorem 5.1 Let **[**·]] : A*π* → У, with У ∈ {PIA*π,* POA*π,* PA*π*}, be an encoding satisfying:

1. compositionality w.r.t. input prefix, parallelism and replication,
2. **[***ω.*0]] —*ω*→ ,
3. E*x, y, z* : *n*(*C*[[·]]) ∩ *n*(*C*[[·]] )= *n*(*C*[[·]]) ∩ *n*([[*x*¯*z*]]) = *n*(*C*[[·]]) ∩ *n*(*C*[[·]])=∅,

! *x*(*y*) ! ! |

1. *C*[[·]]

!

is a singularly-structured context.

Then **[**· **]** is not must-preserving.

Proof. (Sketch of:) Suppose that **[**·]] in *C*[[·]]

!

is not in the scope of a replication.

Then it is possible to prove that the hole is prefixed in *C*[[·]]. Now it suffices to consider that *x*(*y*)*.*0 *must* !*ω.*0 but *Cx*(*y*)[[[0]]] *must C*![[[*ω.*0]]], since every hole is

!

prefixed in *C*[[·]]

*x*(*y*)

, the hole is prefixed in *C*[[·]]

and *Cx*(*y*)[[[0]]] | *C*![[[*ω.*0]]] /—→ by (3).

Now suppose that **[**·]] in *C*[[·]] is in the scope of a replication. Then it is possible to prove that 6*x*'*, z*' ∈U , either *C*![*Cx*(*y*)[[[*ω.*0]]]] | [[*x*¯*z* | *x*¯'*z*']] or *Cx*(*y*)[[[*ω.*0]]] | *C*![[[*x*¯*z* | *x*¯'*z*']]] has at least one infinite computation such that **[***ω.*0]] does not interact or participate in the computation. Now it suffices to consider both *P* | *o* (with **[***P* ]] | [[*o*]]) and *P* ' | *o*' (with **[***P* ']] | [[*o*']]), where *P* =!*x*(*y*)*.x*'(*y*')*.ω.*0, *o* = *x*¯*z* | *x*¯'*z*' (*x* /= *x*'), *P* ' =

!

!

*x*(*y*)*.x*'(*y*')*.ω.*0 and *o*' =!(*x*¯*z* | *x*¯'*z*'), obtaining that **[**· **]** cannot be must-preserving.

Let us discuss the premises in the above theorem. Compositionality is in gen- eral a reasonable condition for an encoding. As argued above, the second condition is validated if the encoding is to preserve divergence. The third condition is val- idated if in the encoding of each operator *op* the context where the encodings of the operands are placed, i.e. *Cop*, uses unique names only. Replication represents an infinite parallel composition, so it is arguably reasonable to require homomor- phism for replication since homomorphism for the parallel operator is arguably a reasonable requirement—see Remark [4.2](#_bookmark8). Regarding (4), we already pointed out in Remark [4.3](#_bookmark9) that in compositional encodings the contexts *Cop* are typically singularly-structured [7](#_bookmark16) .

We conclude this section with a theorem stating a general and uniform impos- sibility result for the existence of encodings from A*π* into any (semi-)persistent calculus. The statement results as an immediate consequence of Theorem [5.1](#_bookmark14) in the case of homomorphism w.r.t replication, as it implies *n*(*C*[[·]])= ∅*.*

!

Theorem 5.2 Let **[**·]] : A*π* → У, with У ∈ {PIA*π,* POA*π,* PA*π*}, be an encoding satisfying:

1. compositionality w.r.t. input prefix and parallelism,
2. homomorphism w.r.t replication,
3. **[***ω.*0]] —*ω*→ .

Then **[**· **]** is not must-preserving.

# Specialized impossibility results for persistence

In the previous section we gave a uniform impossibility result for the existence of encodings of A*π* into the (semi-)persistent calculi. In this section, we give further impossibility results, under different hypotheses, taking into account particular fea- tures of some of the (semi-)persistent calculi, namely PA*π* and PIA*π* [8](#_bookmark17) .

For technical reasons we introduce a particular kind of contexts in У that differ from those we have introduced in Section [4](#_bookmark5), in that brackets do not disappear once we “fill the holes” with process terms.

Definition 6.1 [Focusing contexts] A *focusing context C*{} for У is generated by the following grammar:

*C*{} := { }*σ*

0 *out*

*in.C*{}

(*νx*)*C*{}

*C*{} | *C*{}

!*C*{}

where *σ* is a (name) substitution, and *in* and *out* are resp. input and output,

according to У syntax. (e.g. *in* =!*x*(*y*), and either *out* =

*out* =!*x*¯*z* if У = PA*π*)

*x*¯*z* if У = PIA*π* or

7 Notice that the case **[**!*P* ]] = [[*P* ]] |![[*P* ]], where *C*! = [*.*]|![*.*] is not singularly-structured, can be rewritten via

≡ as **[**!*P* ]] =![[*P* **]**, where the corresponding *C*! =![·] is singularly-structured.

8 We also stated this kind of specialized result for POA*π* but for reasons of space and its restricted nature it has been moved in the appendix

Notation 6.1 *Given a focusing context C*{} *and P* ∈ У*, C*{*P* } *is the term obtained by replacing each occurrence* { }*σ in C*{ } *by* {*P* }*σ. We denote by* L(*P* ) *(ranged over by B, B*'*, ..) the set* {*C*{*P* }| *P* ∈ У*,C*{ } *is a focusing context*}*.*

*An occurrence of* {*P* }*σ is preﬁxed in B* ∈ L(*P* ) *if it is in the scope of an input*

*preﬁx. We write Pref*(*B*) *when every occurrence of* {*P* }*σ is preﬁxed in B.*

The structural congruence and the reduction semantics for the language L(*P* ) are both defined on the basis of the ones for У, the only difference being that terms are in L(*P* ) instead than in У and that unguarded braces (i.e. terms out of the scope of an input prefix like {*P* }*σ*) are assumed as deadlocked terms. This is not a concern, because for the proof of our main results, for every *σ* each occurrence of

{*P* }*σ* is prefixed, i.e. in the scope of an input prefix.

It is possible to prove that L(*P* ) is closed under substitution and, as a con- sequence, under reduction. Denoting by *Unbrace*(*B*) the У process obtained by removing all the braces from *B* and by applying the substitutions, it is also pos- sible to prove that: (i) *B* ∈ L(*P* ), then *B* —→ *B*' implies *B*' ∈ L(*P* ) and

*Unbrace*(*B*) —→ *Unbrace*(*B*'), and (ii) *Pref*(*B*) and *Unbrace*(*B*) —→ *R* implies

that E*B*' ∈ L(*P* ) such that *B* —→ *B*' and *R* ≡ *Unbrace*(*B*').

Focusing contexts are extended for the testing machinery, adding rule

{*ω.E*}*σ* —*ω*→ in Table [3](#_bookmark3). Notice that, since every *σ* is defined over U and *ω* /∈ U , then 6*E* ∈У and *B* ∈ L(*P* ), (i) {*ω.E*}*σ* —*ω*→; (ii) *B* —*ω*→ implies *Bσ* —*ω*→; (iii) *B* —*ω*→ if and only if *Unbrace*(*B*) —*ω*→, where *Bσ* represents the result of the application of *σ* to *B* (assuming to use *α*-equivalence to avoid collision of names).

Persistent calculus: To prove our main results, we define a function over L(*P* ),

*min*(*B*) (Table [4](#_bookmark18)), and a predicate, *Pr* (Table [5](#_bookmark19)).

*min*(*B*)= +∞ if *B* ∈ У;

*min*({*P* })= 0;

*min*(*x*(*y*)*.B*)=1 + *min*(*B*);

*min*((*νx*)*B*)= *min*(*B*);

*min*(*B* | *B*')= min{*min*(*B*)*, min*(*B*')};

*min*(!*B*)= *min*(*B*)*.*

Table 4 Function *min*.

*min*(!*x*(*y*)*.B*) ≥ 2

*Pr*(*B*)

Red Res

*Pr*(!*x*¯*z* | !*x*(*y*)*.B*) *Pr*((*νy*)*B*)

*Pr*(*B*') *, B*' ≡ *B*

Par

*Pr*(*B*1)

Cong

*Pr*(*B*1 | *B*2)

*Pr*(*B*)

Table 5 Predicate *Pr*.

We can prove that *Pr* is closed under reduction and it implies *Pref*. As a con- sequence, for every *B* ∈ L(*P* ) such that *Pr*(*B*), it is possible to build a non-empty

maximal computation from *B* where any term of the computation verifies the pred- icate *Pr*. We can now state a rather strong negative result for PA*π*.

Theorem 6.2 Let **[**· **]** be an encoding from A*π* into PA*π* that satisfies:

1. compositionality w.r.t. input prefix,
2. **[***ω.*0]] —*ω*→ .

Then **[**· **]** is not must-preserving.

Proof. By contradiction, it suffices to suppose **[**· **]** being must-preserving, consider *P* = *x*¯*z*|*x*¯*z* and *o* = *x*(*y*)*.x*(*y*)*.ω.*0. and observe that *Pr*([[*x*¯*z*|*x*¯*z*]]|*Cx*(*y*)[*Cx*(*y*){[[*ω.*0]]}]) holds. Hence, it is possible to prove that there is a non-empty maximal computa- tion from **[***x*¯*z* | *x*¯*z*]] |[[*x*(*y*)*.x*(*y*)*.ω.*0]] where any term of the computation verifies the predicate *Pr*, i.e. every term does not perform *ω* (since every occurrence of **[***ω.*0]] is prefixed).

The above theorem resembles the impossibility result in [[24](#_bookmark42)] about the existence of an encoding from A*π* into PA*π* wrt weak bisimulation (and output equivalence). However, the hypothesis of the result in [[24](#_bookmark42)] is different. Namely, it is restricted to encodings homomorphic wrt parallelism.

Persistent-input calculus: Regarding PIA*π* (and POA*π*), a *Pr*-like predicate does not preserve *Pref* (it suffices to consider *B*1 = ¯*b* | *c*¯ | !*b.*!*c.*{*P* }*σ*, where *P* ∈ PIA*π*, and *B*2 =!¯*b* | !*c*¯ | *b.c.*{*P* }*σ*, where *P* ∈ POA*π*). In the case of PIA*π*, an ad-hoc predicate, *Prin*, is defined. The predicate selects those processes *B* ∈ L(*P* ) such that - every {*P* }*σ* occurrence is in the scope of an input prefix *x*(*y*), for some *x* ∈ *fn*(*B*) and *y* ∈ U , - there exists an input component !*x*(*y*)*.B* (prefixing

{*P* }*σ*) such that *min*(!*x*(*y*)*.B*) ≥ 2, - every parallel component !*xi*(*y*)*.B* is such that

*min*(!*xi*(*y*)*.B*) ≥ 1 if *xi* = *x* and *min*(!*xi*(*y*)*.B*) ≥ 2 if *xi* /= *x*. The results for *Pr* can be proven in a similar way for *Prin*. In particular, whenever E*x* ∈ *fn*(*B*) such that *Prin*(*B, x*), it is possible to build a maximal computation from *B* where any term of the computation verifies the predicate *Prin*. Hence, it leads us to the negative result below.

Theorem 6.3 Let **[**· **]** be an encoding from A*π* into PIA*π* that satisfies:

1. compositionality w.r.t. input prefix,
2. **[***ω.*0]] —*ω*→ ,
3. if *fn*(*P* ) ∩ *bn*(*x*(*y*)) = ∅ then *fn*([[*P* ]]) ∩ *bn*(*C*[[·]]

*x*(*y*)

)= ∅,

1. **[***x*(*y*)*.P* ]] ≡ (*νx*1)*..*(*νxn*)(!*u*(*v*)*.C*[[[*P* ]]] | *T* ), for some *x*1*, .., xn, C,T* with *u* /= *xi* for any *i*.

Then **[**· **]** is not must-preserving.

Proof. It is possible to prove that E*h* ∈ *fn*(*C*[[·]]

*x*(*y*)

): *Prin*(*Cx*(*y*)[*Cx*(*y*){[[*ω.*0]]}]*, h*).

Now, it suffices to assume, by contradiction, **[**· **]** being must-preserving and proving

that *Prin*([[*x*¯*z* | *x*¯*z*]] | *Cx*(*y*)[*Cx*(*y*){[[*ω.*0]]}]*, h*) holds. Hence, it is possible to prove that there is a non-empty maximal computation from **[***x*¯*z* | *x*¯*z*]] | [[*x*(*y*)*.x*(*y*)*.ω.*0]] where

any term of the computation verifies the predicate *Prin*, i.e. every term does not perform *ω* (since every occurrence of **[***ω.*0]] is prefixed).

Notice that the encoding in Definition [4.5](#_bookmark10) satisfies every condition of the fol- lowing theorem and, more important, that *Prin* does not rely on any divergence as- sumption, differently from *Pr*. We have already argued for the first two conditions as being reasonable. Intuitively, the third condition expresses that a *non-binding* property wrt input prefix: if in a source term *x*(*y*)*.P* none of the free names of *P* is bound by the input prefix, then the free names of **[***P* **]** must not be bound either (by a binder in the context where **[***P* **]** is placed) in the encoding of **[***x*(*y*)*.P* ]]. Finally, the fourth condition basically expresses that A*π* inputs should be mapped into PIA*π* inputs possibly allowing some other material around it. This is validated, e.g., by encodings that preserve input/output polarities—i.e. A*π* inputs/outputs must be mapped into PIA*π* input/outputs [9](#_bookmark26) .

# Related work and concluding remarks

Most of the related work was discussed in the introduction. In a different con- text, in [[22](#_bookmark40)] it is shown that the separate choice encoding of the *π*-calculus into the asynchronous *π*-calculus is faithful with respect to weak bisimulation, while in [[8](#_bookmark27)] the authors prove that no must-preserving encoding of the (choiceless) syn- chronous pi-calculus into the asynchronous one exists. Hence must semantics is a good candidate to study the expressiveness of persistence when divergence is taken into account. Nevertheless, differently from [[8](#_bookmark27)], this work does not consider any synchronous language, i.e. the must semantics is studied in a uniform and purely asynchronous framework. As previously mentioned the study of persistence in [[24](#_bookmark42)] is incomplete as ignores the crucial issue of divergence. In this paper, we used the divergence-sensitive framework of testing semantics and adapted and exploited the techniques of [[8](#_bookmark27)] to give a more complete account of the expressiveness of persistence in asynchronous calculi. In particular, as discussed in the introduction, this work supports informal expressiveness loss claims in persistent asynchronous languages [[3](#_bookmark24),[14](#_bookmark34),[11](#_bookmark29)].

# References

1. J. Alves-Foss. An Efficient Secure Authenticated Group Key Exchange Algorithm for Large and Dynamic Groups. In *Proceedings of the 23rd National Information Systems Security Conference*, 2000.
2. R. Amadio and D. Lugiez and V. Vanackere. On the Symbolic Reduction of Processes with Cryptographic Functions. *TCS: Theoretical Computer Science* 290, 2003.
3. E. Best, F. de Boer, and C. Palamidessi. Partial order and sos semantics for linear constraint programs.

*In Proc. of Coordination’97*, volume 1282 of LNCS, 1997.

1. B. Blanchet. From linear to classical logic by abstract interpretation. *Information Processing Letters*

95(5), 2005.

1. M. Boreale and M. Buscemi. A Framework for the Analysis of Security Protocols, *Lecture Notes in Computer Science* 2421, 2002.

9 E.g., the encoding in Definition [4.5](#_bookmark10) satisfies all conditions of Theorem [6.3](#_bookmark20).

1. E. Brinksma, A. Rensink, W. Vogler. Fair Testing, *Proc. of CONCUR’95, LNCS* 962, pp. 313-327, 1995.
2. D. Cacciagrano, F. Corradini. On Synchronous and Asynchronous Commu- nication Paradigms, *Proc.* *of ICTCS ’01, LNCS* 2202, pp. 256-268, 2001.
3. D. Cacciagrano, F. Corradini, C. Palamidessi. Separation of Synchronous and Asynchronous Communication Via Testing. *Proc. of EXPRESS’05*. Electr. Notes Theor. Comput. Sci. 154(3): 95- 108, 2006. An extended version will appear in *Theoretical Computer Science*.
4. M. Carbone, S. Maffeis. On the Expressive Power of Polyadic Synchronisation in pi-calculus. *Nord. J. Comput.* 10(2): 70-98, 2003.
5. I. Castellani, M. Hennessy. Testing Theories for Asynchronous Languages, *Proc. of FSTTCS ’98, LNCS*

1530, pp. 90-101, 1998.

1. F. Crazzolara. Language, Semantics, and Methods for Security Protocols. *PhD Dissertation*, University of Aarhus, Denmark, 2003.
2. F. Crazzolara and G. Winskel. Events in security protocols, *Proceedings of the 8th ACM Conference on Computer and Communications Security*, ACM Press, 2001.
3. R. De Nicola, M. Hennessy. Testing Equivalence for Processes, *Theoretical Computer Science* 34, pp. 83-133, 1984.
4. F. Fages, P. Ruet, and S. Soliman. Linear concurrent constraint programming: operational and phase semantics. *Information and Computation*, 2001.
5. M. Fiore and M. Abadi. Computing symbolic models for verifying cryptographic protocols. *Proc. CSFW-14. IEEE*, 2001.
6. D. Gorla: On the Relative Expressive Power of Asynchronous Communication Primitives. *FoSSaCS 2006*, 47-62, 2006.
7. D. Gorla: Synchrony vs Asynchrony in Communication Primitives *Proc. of EXPRESS’06*, 47-62, 2006.
8. S. Maffeis and I. Phillips. On the computational strength of pure ambient calculi. *Proc. of EXPRESS ’03*, 2003.
9. R. Milner. *Communication and Concurrency*, Prentice-Hall International, 1989.
10. R. Milner, J. Parrow, D. Walker. A Calculus of Mobile Processes, Part I and II, *Information and Computation* 100, pp. 1-78, 1992.
11. M. Merro, D. Sangiorgi. On asynchrony in name-passing calculi, *Proc. of ICALP ’98, LNCS* 1443, 1998.
12. U. Nestmann. What is a ‘Good’ Encoding of Guarded Choice?, *Information and Computation* 156, pp. 287-319, 2000.
13. C. Palamidessi. Comparing the Expressive Power of the Synchronous and Asynchronous *π*-calculus, *Mathematical Structures in Computer Science* 13(5), pp. 685-719, 2003. A preliminary version appeared in the proceedings of POPL ’97.
14. C. Palamidessi, V. Saraswat, F. Valencia and B. Victor. On the Expressiveness of Linearity vs Persistence in the Asynchronous Pi Calculus. *LICS 2006:59-68*, 2006.
15. L. C. Paulson. Mechanized proofs for a recursive authentication protocol. In *10th Computer Security* *Foundations Workshop*, 1997.
16. I. Phillips and M. Vigliotti Electoral Systems in Ambient Calculi. *FoSSaCS’04*. 2004.
17. I. Phillips, M. Vigliotti. Leader Election in Rings of Ambient Processes. *Electr. Notes Theor. Comput. Sci*. 128(2): 185-199, 2005.
18. K.V.S. Prasad. Broadcast Calculus Interpreted in CCS up to Bisimulation. *In Proceedings of Express’01*, volume 52 of Electronic Notes in Theoretical Computer Science, pages 83-100. Elsevier, 2002.
19. D. Sangiorgi. The name discipline of uniform receptiveness. *Theoretical Computer Science*, 221(12):457493, 1999.
20. D. Sangiorgi and D. Walker. *The π-calculus: A Theory of Mobile Processes*. Cambridge University Press, 2001.
21. V. Saraswat and P. Lincoln. Higher-order linear concurrent constraint programming. *Technical report, Xerox PARC*, 1992.
22. V. Saraswat. Concurrent Constraint Programming. The MIT Press, 1993.
23. M. Vigliotti, I. Phillips, C. Palamidessi. Separation Results Via Leader Election Problems. *FMCO 2005*, 172-194, 2005.

# Appendix

In this section, we give the definitions and the proofs omitted in Section [4](#_bookmark5).

Definition A.1 Define *P* ↓*x*¯ iff E*z*1*,... , zn, y,R* : *P* ≡ (*νz*1)*..*(*νzn*)(*x*¯*y* | *R*) and

6*i* ∈ [1*..n*]*,x* /= *zi*. Furthermore, *P* ⇓*x*¯ iff E*Q* : *P* —∗→ *Q* ↓*x*¯.

Definition A.2 (*Barbed Bisimilarity, Barbed Congruence*) A *weak barbed bisimu- lation* is a symmetric relation R satisfying the following: (*P, Q*) ∈R implies that:

* 1. *P* —→ *P* ' then E*Q*' : *Q* —∗→ *Q*' ∧ (*P* '*, Q*') ∈ R.
  2. *P* ↓*x*¯ then *Q* ⇓*x*¯ *.*

We say that *P* and *Q* are *weak barbed bisimilar*, written *P*

*.*

≈ *Q*, iff (*P, Q*) ∈ R

for some weak barbed bisimulation R. Furthermore, *weak barbed congruence* ≈ is

*.*

defined as: *P* ≈ *Q* iff for every process context *C*[·], *C*[*P* ] ≈ *C*[*Q*].

Proposition [4.6](#_bookmark11) 6 *P* ∈ A*π,* 6 *o* ∈ 0 ⊆ PIA*π*, *P sat o* if and only if **[***P* ]] *sat o*, where *sat* can be respectively *may* and *fair* .

Proof. *P* ≈ [[*P* **]** implies that 6 *o* ∈ 0 ⊆ PIA*π*, *P* | *o* ≈ [[*P* ]] | *o*. Extending the

*ω*

notion of barb to *ω*, we have *T* ⇓*ω*

iff *T*

=⇒. Suppose *P fair o*. Then for every

maximal computation *P* | *o* = *E*0 —→ *E*1 —→ *..* —→ *Ei* [ —→ *.. .*] we have

*Ei* —∗→ *E*' ↓*ω*, for every *i* ≥ 0. Since *P* | *o* ≈ [[*P* ]] | *o*, then for every maximal

*i*

computation **[***P* ]] | *o* = *A*0 —→ *A*1 —→ *..* —→ *Ai* [ —→ *.. .*] *Ai* ⇓*ω*, for every

*i* ≥ 0. I.e. **[***P* ]] *fair o*. Notice that *may* is a special case of *fair* : *P may o* implies

*P* | *o* —∗→ *E*' ↓*ω* and, since *P* | *o* ≈ [[*P* ]] | *o*, it implies that [[*P* ]] | *o* ⇓*ω*, i.e. **[***P* ]] *may o*.

0

# Appendix

In this section, we give the proofs omitted in Section [5](#_bookmark13). We will use ⟨*P* ⟩ to denote some restricted version of *P* , i.e. any process of the form (*νx*1)*..*(*νxn*)*P* , for some *x*1*,... , xn* ∈ *fn*(*P* ).

Proposition B.1 Let **[**·]] : A*π* → У, with *P* ∈ {PIA*π,* POA*π,* PA*π*}, be an encoding satisfying:

1. must-preservation,
2. E*P* ∈ A*π* such that **[***P* ]] †. Then **[***ω.*0]] —*ω*→ .

Proof. Let *P* ∈ A*π* such that **[***P* ]] †. Since *P must ω.*0 and the encoding **[**·]] is

*must* -preserving, then **[***P* ]] *must* [[*ω.*0 **]**. Since **[***P* ]] †, we have [[*ω.*0]] —*ω*→.

Lemma B.2 Let **[**·]]: A*π* →У ∈{PIA*π,* POA*π,* PA*π*} be an encoding satisfying:

1. compositionality w.r.t. input prefix,
2. must-preservation,
3. **[***ω.*0]] —*ω*→ .

Then 6*x, y* ∈ U , every hole is prefixed in *C*[[·]] .

*x*(*y*)

Proof. By definition we have 0 *must x*(*y*)*.ω.*0, and since **[**· **]** is must-preserving,

*ω*

we have **[**0 **]** *must* [[*x*(*y*)*.ω.*0]]. Hence, **[**0]] *must Cx*(*y*)[[[*ω.*0 **]**]. Since **[***ω.*0]] —→ by

hypothesis, every occurrence of **[***ω.*0]] has to be prefixed in *C*[[·]] .

*x*(*y*)

The following two technical lemmas are used for proving our main results.

Lemma B.3 Let **[**·]]: A*π* →У ∈{PIA*π,* POA*π,* PA*π*} be an encoding satisfying:

1. compositionality w.r.t. input prefix and replication,
2. must-preservation,
3. **[***ω.*0]] —*ω*→ ,
4. E*x, y, z* : *n*(*C*[[·]]) ∩ *n*(*C*[[·]] )= *n*(*C*[[·]]) ∩ *n*([[*x*¯*z*]]) =∅,

! *x*(*y*) !

1. *C*[[·]]

!

is a singularly-structured context,

1. the hole in the context *C*[[·]]

!

is not in the scope of a replication.

Then the hole is prefixed in *C*[[·]].

!

Proof. Since *x*¯*zmust x*(*y*)*.x*(*y*)*.ω.*0, !*x*¯*zmust x*(*y*)*.x*(*y*)*.ω.*0 and **[**· **]** is must-preserving, we have **[***x*¯*z*]] *must* [[*x*(*y*)*.x*(*y*)*.ω.*0]] and **[**!*x*¯*z*]] *must* [[*x*(*y*)*.x*(*y*)*.ω.*0 **]**. Since there is an unsuccessful maximal computation from **[***x*¯*z*]] | [[*x*(*y*)*.x*(*y*)*.ω.*0]], then there is an unsuccessful maximal computation from ( **[***x*¯*z*]] | [[*x*(*y*)*.x*(*y*)*.ω.*0]])*α*, where *α* denotes *α*-equivalence.

By contradiction, suppose that the hole is not prefixed in *C*[[·]]. Then **[***x*¯*z*]] is not prefixed in *C*![[[*x*¯*z*]]] | *Cx*(*y*)[*Cx*(*y*)[[[*ω.*0]]]]. Since every hole is prefixed in *C*[[·]] and the hole is not in the scope of a replication in *C*[[·]], we can prove, by induction

!

!

*x*(*y*)

on the structure of *C*[[·]], that E*B* ∈ У such that *C*![[[*x*¯*z*]]] | *Cx*(*y*)[*Cx*(*y*)[[[*ω.*0]]]] is congruent to *B* = ⟨*T* |[[*x*¯*z*]]|*Cx*(*y*)[*Cx*(*y*)[[[*ω.*0]]]]⟩, where *bn*(*C*![[[*x*¯*z*]]]|*Cx*(*y*)[*Cx*(*y*)[[[*ω.*0]]]])

!

= *bn*(*B*) and *T* ∈ У (without loss of generality, we use the same notation for

*C*![[[*x*¯*z*]]] | *Cx*(*y*)[*Cx*(*y*)[[[*ω.*0]]]] before and after applying *α*-equivalence). Recall that

[[*x*¯*z*]], *C*[[·]]

*x*(*y*)

and *T* do not contain *ω*. Now, consider the following (unsuccessful)

maximal computation from **[***x*¯*z*]] | *Cx*(*y*)[*Cx*(*y*)[[[*ω.*0]]]] (there exists at least one):

[[*x*¯*z*]] | *Cx*(*y*)[*Cx*(*y*)[[[*ω.*0]]]] = *A*0 —→ *A*1 —→ *...* —→ *Ai*[ —→ *.. .*]

where 6*i* ≥ 0*, Ai* /—*ω*→ .

* If this computation is infinite: then there exists an unsuccessful maximal compu- tation from *B*, i.e. from [[!*x*¯*z*]] | [[*x*(*y*)*.x*(*y*)*.ω.*0]], contradicting the hypothesis.
* If this computation is finite: then *B* —∗→ ⟨*T* | *Ai*⟩, where *Ai* /—→ and *Ai* /—*ω*→.

If *T* †, again there exists an unsuccessful maximal computation from *B*, i.e.

from **[**!*x*¯*z*]] | [[*x*(*y*)*.x*(*y*)*.ω.*0]], contradicting the hypothesis.

Otherwise, *T Ai* /—*ω*→.

—∗→ *D*, i.e *B* —∗→ ⟨*D*| *Ai*⟩, where *D* /—→ , *Ai* /—→ , *D* /—*ω*→ and

Since ≡ does not change free names, *fn*(*C*![[[*x*¯*z*]]] | *Cx*(*y*)[*Cx*(*y*)[[[*ω.*0]]]]) =

*fn*(*B*). Since *B* is such that *bn*(*C*![[[*x*¯*z*]]] | *Cx*(*y*)[*Cx*(*y*)[[[*ω.*0]]]]) = *bn*(*B*),

we have *n*(*C*![[[*x*¯*z*]]] | *Cx*(*y*)[*Cx*(*y*)[[[*ω.*0]]]]) = *n*(*B*). Furthermore, *fn*(*Ai*) ⊆

(*n*([[*x*¯*z*]] | *Cx*(*y*)[*Cx*(*y*)[·]]) ∪ *n*([[*ω.*0]])) ⊆ (*n*([[*x*¯*z*]]) ∪ *n*(*C*[[·]]

*x*(*y*)

) ∪ *n*([[*ω.*0]])) and *fn*(*D*) ⊆

*n*(*T* ) ⊆ *n*(*C*[[·]]).

!

By initial hypothesis, every occurrence of **[***ω.*0]] is prefixed in *Ai* and *n*(*C*[[·]]) ∩

!

!

[[·]]

*n*(*C*

*x*(*y*)

)= *n*(*C*[[·]]) ∩ *n*([[*x*¯*z*]]) = ∅. It follows that ⟨*D*| *Ai*⟩ —→ . Since ⟨*D*| *Ai*⟩ —

*ω*→,

we contradict that **[**!*x*¯*z*]] *must* [[*x*(*y*)*.x*(*y*)*.ω.*0]].

Lemma B.4 Let **[**·]]: A*π* →У ∈{PIA*π,* POA*π,* PA*π*} be an encoding satisfying:

1. compositionality w.r.t. input prefix and replication,
2. **[***ω.*0]] —*ω*→ ,
3. E*x, y, z* : *n*(*C*[[·]]) ∩ *n*(*C*[[·]] )= *n*(*C*[[·]]) ∩ *n*([[*x*¯*z*]]) =∅,

! *x*(*y*) !

1. *C*[[·]]

!

is a singularly-structured context,

1. the hole in the context *C*[[·]]

!

is not in the scope of a replication.

Then **[**· **]** is not must-preserving.

Proof. By contradiction, suppose **[**·]] is *must* -preserving. Then *x*(*y*)*.*0 *must* !*ω.*0. Consider *Cx*(*y*)[[[0]]]|*C*![[[*ω.*0]]]: since every hole is prefixed in *Cx*(*y*), the hole is prefixed

in *C*[[·]]

!

and *Cx*(*y*)[[[0]]] | *C*![[[*ω.*0]]] /—→ by (3), we have *Cx*(*y*)[[[0]]] *must C*![[[*ω.*0]]].

Lemma B.5 Let **[**·]]: A*π* →У ∈{PIA*π,* POA*π,* PA*π*} be an encoding satisfying:

1. compositionality w.r.t. input prefix, parallelism and replication,
2. must-preservation,
3. **[***ω.*0]] —*ω*→ ,
4. E*x, y, z* : *n*(*C*[[·]]) ∩ *n*(*C*[[·]] )= *n*(*C*[[·]]) ∩ *n*([[*x*¯*z*]]) = *n*(*C*[[·]]) ∩ *n*(*C*[[·]])=∅,

! *x*(*y*) ! ! |

1. *C*[[·]]

!

is a singularly-structured context,

1. the hole in the context *C*[[·]]

!

is in the scope of a replication.

Then 6*x*'*, z*' ∈U , either *C*![*Cx*(*y*)[[[*ω.*0]]]] | [[*x*¯*z* | *x*¯'*z*']] or *Cx*(*y*)[[[*ω.*0]]] | *C*![[[*x*¯*z* | *x*¯'*z*']]] has at least one infinite computation such that **[***ω.*0]] does not interact or participate in the computation.

Proof. Let’s assume, by contradiction, that both *C*![*Cx*(*y*)[[[*ω.*0]]]] | [[*x*¯*z* | *x*¯'*z*']] and *Cx*(*y*)[[[*ω.*0]]] | *C*![[[*x*¯*z* | *x*¯'*z*' **]**] do not have infinite computations where **[***ω.*0]] interacts. Then *C*![*Cx*(*y*)[*ω.*0]], **[***x*¯*z* | *x*¯'*z*']], *Cx*(*y*)[[[*ω.*0]]] and *C*![[[*x*¯*z* | *x*¯'*z*' **]**] do not have infinite computations where **[***ω.*0]] don’t interact.

By must-preservation, **[**!*x*(*y*)*.ω.*0]] *must* [[*x*¯*z* | *x*¯'*z*']], where [[*ω.*0]] is prefixed in

*Cx*(*y*)[[[*ω.*0]]]]. From (2) and (5) we can show that **[**!*ω.*0]] —→ *..* —→ ⟨!⟨[[*ω.*0]]|*P* ⟩|*Q*⟩. From (1) we know that 6*U* , there is at least one computation such that **[**!*U* ]] —→

*..* —→ ⟨!⟨[[*U* ]]|*P* ⟩|*Q*⟩. From the above and considering *U* = *Cx*(*y*)[[[*ω.*0]]], there is at least one computation

[[!*x*(*y*)*.ω.*0]] = *C*![*Cx*(*y*)[[[*ω.*0]]]] —→ *...* —→ ⟨!⟨*Cx*(*y*)[[[*ω.*0]]] | *P* ⟩| *Q*⟩

where ⟨*Cx*(*y*)[[[*ω.*0]]] | *P* ⟩ /—→ (otherwise, !⟨*Cx*(*y*)[[[*ω.*0]]] | *P* ⟩ diverges without inter-

vention from **[***ω.*0]], as **[***ω.*0]] is prefixed in *C*[[·]] ).

*x*(*y*)

By (4), we know that ⟨!⟨*Cx*(*y*)[[[*ω.*0]]] | *P* ⟩| *Q*⟩ ≡ !*Cx*(*y*)[[[*ω.*0]]] | ⟨*P* | *Q*⟩. We also know that **[***x*¯*z* | *x*¯'*z*']] /—→ , (otherwise *C*![[[*x*¯*z* | *x*¯'*z*' **]**] would be divergent without the intervention from **[***ω.*0]]). From the above, we have at least one computation

*C*![*Cx*(*y*)[[[*ω.*0]]]] | [[*x*¯*z* | *x*¯'*z*']] —→ *...* —→ !*Cx*(*y*)[[[*ω.*0]]] | ⟨*P* | *Q*⟩| [[*x*¯*z* | *x*¯'*z*']]

*ω*

where *Cx*(*y*)[[[*ω.*0]]] /—→ , [[*x*¯*z* | *x*¯'*z*']] /—→ and *S* = ⟨*P* | *Q*⟩ —→ (as **[***ω.*0]] is prefixed

in any possible occurrence of *C*[[·]]

*x*(*y*)

in ⟨*P* | *Q*⟩).

As *Cx*(*y*)

[[[*ω.*0]]] / —→ *,* [[*x*¯*z* | *x*¯'*z*']] / —→ , *Cx*(*y*)

[[[*ω.*0]]] —

*ω*→, [[*x*¯*z* | *x*¯'*z*']]

—*ω*→ and by

must-preservation we know that *Cx*(*y*)[[[*ω.*0]]] *must* [[*x*¯*z* | *x*¯'*z*' **]**, then there must be at least one interaction between *Cx*(*y*)[[[*ω.*0]]] and **[***x*¯*z* | *x*¯'*z*']]. By compositionality w.r.t

bang and input prefix, the structure of *C*[[·]]

*x*(*y*)

and **[***x*¯*z* | *x*¯'*z*' **]** can be characterized in

one of the following ones:

1. either *Cx*(*y*)[·] ≡ ⟨*h*(*k*)*.P* '|*Q*'⟩ or ≡ ⟨!*h*(*k*)*.P* '|*Q*'⟩, and either **[***x*¯*z*|*x*¯'*z*']] ≡ ⟨*h*¯*k*|*R*'⟩

or ≡ ⟨!*h*¯*k*|*R*'⟩.

У = POA*π*: then !*Cx*(*y*)[[[*ω.*0]]] | [[*x*¯*z* | *x*¯'*z*']] †, without intervention from **[***ω.*0]]. As *C*![*Cx*(*y*)[[[*ω.*0]]]] |[[*x*¯*z* | *x*¯'*z*']] —→ *...* —→ !*Cx*(*y*)[[[*ω.*0]]] | *S* | [[*x*¯*z* | *x*¯'*z*']], *C*![*Cx*(*y*)[[[*ω.*0]]]] | [[*x*¯*z* | *x*¯'*z*' **]** diverges without intervention from **[***ω.*0]].

У = PIA*π*: then *Cx*(*y*)[[[*ω.*0]]] | ![[*x*¯*z* | *x*¯'*z*']] † without intervention from **[***ω.*0]]. As *Cx*(*y*)[[[*ω.*0]]] |*C*![[[*x*¯*z* | *x*¯'*z*']]] —→ *...* —→ *Cx*(*y*)[[[*ω.*0]]] | *S* | ![[*x*¯*z* | *x*¯'*z*']], *Cx*(*y*)[[[*ω.*0]]] | *C*![[[*x*¯*z* | *x*¯'*z*' **]**] diverges without intervention from **[***ω.*0]].

1. either *Cx*(*y*)[·] ≡ ⟨*h*¯*k*|*R*'⟩ or ≡ ⟨!*h*¯*k*|*R*'⟩, and either **[***x*¯*z*|*x*¯'*z*']] ≡ ⟨*h*(*k*)*.P* '|*Q*'⟩ or

≡ ⟨!*h*(*k*)*.P* '|*Q*'⟩.

У = POA*π*: then *Cx*(*y*)[[[*ω.*0]]]|![[*x*¯*z*|*x*¯'*z*']] † without intervention from **[***ω.*0]]. As *Cx*(*y*)[[[*ω.*0]]] |*C*![[[*x*¯*z* | *x*¯'*z*']]] —→ *...* —→ *Cx*(*y*)[[[*ω.*0]]] | *S* | ![[*x*¯*z* | *x*¯'*z*']], *Cx*(*y*)[[[*ω.*0]]] | *C*![[[*x*¯*z* | *x*¯'*z*' **]**] diverges without intervention from **[***ω.*0]].

У = PIA*π*: then !*Cx*(*y*)[[[*ω.*0]]] | [[*x*¯*z* | *x*¯'*z*']] †, without intervention from **[***ω.*0]]. As *C*![*Cx*(*y*)[[[*ω.*0]]]]| [[*x*¯*z* | *x*¯'*z*']] —→ *...* —→ !*Cx*(*y*)[[[*ω.*0]]] | *S*| [[*x*¯*z* | *x*¯'*z*']], *C*![*Cx*(*y*)[[[*ω.*0]]]]|[[*x*¯*z* | *x*¯'*z*' **]** diverges without intervention from **[***ω.*0]].

From the above, we can conclude that for any encoding **[**·]] from A*π* into PIA*π* or POA*π*, either *Cx*(*y*)[[[*ω.*0]]] | *C*![[[*x*¯*z* | *x*¯'*z*']]] or *C*![*Cx*(*y*)[[[*ω.*0]]]]|[[*x*¯*z* | *x*¯'*z*']] diverges without intervention from **[***ω.*0]] (or both in the case from A*π* into PA*π*).

Lemma B.6 Let **[**·]]: A*π* →У ∈{PIA*π,* POA*π,* PA*π*} be an encoding satisfying:

* 1. compositionality w.r.t. input prefix, parallelism and replication,
  2. **[***ω.*0]] —*ω*→ ,
  3. E*x, y, z* : *n*(*C*[[·]]) ∩ *n*(*C*[[·]] )= *n*(*C*[[·]]) ∩ *n*([[*x*¯*z*]]) = *n*(*C*[[·]]) ∩ *n*(*C*[[·]])=∅,

! *x*(*y*) ! ! |

* 1. *C*[[·]]

!

is a singularly-structured context,

* 1. the hole in the context *C*[[·]]

!

is in the scope of a replication.

Then **[**· **]** is not must-preserving.

Proof. Suppose that **[**· **]** is must-preserving. Consider *P* =!*x*(*y*)*.x*'(*y*')*.ω.*0, *o* = *x*¯*z* | *x*¯'*z*' (*x* /= *x*'), *P* ' = *x*(*y*)*.x*'(*y*')*.ω.*0 and *o*' =!(*x*¯*z* | *x*¯'*z*'). It is possible to verify that *P must o* and *P* ' *must o*'.

*P must o* implies **[***P* ]] *must* [[*o* **]** by must-preservation. It follows that

*C* [*C*

[*C* ' ' [[[*ω.*0]]]]] *must* [[*x*¯*z* | *x*¯'*z*']]. As [[*ω.*0]] is prefixed in *C*[[·]]

, then in every

! *x*(*y*)

*x* (*y* )

*x*'(*y*')

computation from *C*![*Cx*(*y*)[*Cx*'(*y*')[[[*ω.*0]]]]] | [[*x*¯*z* | *x*¯'*z*']], *Cx*'(*y*')[[[*ω.*0]]] must interact to unprefix one occurrence of **[***ω.*0]].

*P* ' *must o*' implies **[***P* ']] *must* [[*o*' **]** by must-preservation. It follows that

1. [*C* ' ' [[[*ω.*0]]]]] *must C* [[[*x*¯*z* | *x*¯'*z*']]]. As **[***ω.*0]] is prefixed in *C*[[·]]

, then in every

*x*(*y*) *x* (*y* ) !

*x*'(*y*')

computation from *Cx*(*y*)[*Cx*'(*y*')[[[*ω.*0]]]] | *C*![[[*x*¯*z* | *x*¯'*z*']]], *Cx*'(*y*')[[[*ω.*0]]] must interact to unprefix one occurrence of **[***ω.*0]].

By lemma [B.5](#_bookmark53), *C*![*Cx*(*y*)[[[*ω.*0]]]] | [[*x*¯*z* | *x*¯'*z*']] or *Cx*(*y*)[[[*ω.*0]]] | *C*![[[*x*¯*z* | *x*¯'*z*']]] has at least one infinite computation such that **[***ω.*0]] does not interact or partici- pate in the computation. Applying (1), either *C*![*Cx*(*y*)[*Cx*'(*y*')[[[*ω.*0]]]]] | [[*x*¯*z* | *x*¯'*z*']] or *Cx*(*y*)[*Cx*'(*y*')[[[*ω.*0]]]] | *C*![[[*x*¯*z* | *x*¯'*z*' **]**] has at least one infinite computation such that *Cx*'(*y*')[[[*ω.*0]]] does not interact or participate in this computation, i.e. either *C*![*Cx*(*y*)[*Cx*'(*y*')[[[*ω.*0]]]]] *must* [[*x*¯*z*|*x*¯'*z*']] or *Cx*(*y*)[*Cx*'(*y*')[[[*ω.*0]]]] *must C*![[[*x*¯*z* | *x*¯'*z*']]]. It contradicts that **[**· **]** is must-preserving.

# C Appendix

In this section, we give the proofs omitted in Section [6](#_bookmark15).

*A class of calculi with focusing contexts*

Proposition C.1 Let *B* ∈ L(*P* ). Then:

1. *B* —→ *B*' implies *B*' ∈ L(*P* ) and *Unbrace*(*B*) —→ *Unbrace*(*B*');
2. *Pref*(*B*) and *Unbrace*(*B*) —→ *R* implies that E*B*' ∈ L(*P* ) such that *B* —→ *B*'

and *R* ≡ *Unbrace*(*B*').

Proof. First, note that L(*P* ) is closed under substitution, and that the structural congruence is preserved by *Unbrace*. First we prove item (i). We proceed by induc- tion on the depth of the reduction *B* —→ *B*'.

Com *out* | *in.B* —→

⎪ *B*{*z/y*}| *in.B*

⎪

⎨

(PIA*π*) *in* =!*x*(*y*)*, out* = *x*¯*z*

*out* | *B*{*z/y*}

(POA*π*) *in* = *x*(*y*)*, out* =!*x*¯*z*

Par

*B*1 —→ *B*'

⎪⎪⎩ *B*{*z/y*}| *out* | *in.B* (PA*π*) *in* =!*x*(*y*)*, out* =!*x*¯*z*

1

Res

*B*1 | *B*2 —→ *B*' | *B*2

1

*B* —→ *B*'

(*νx*)*B* —→ (*νx*)*B*'

Cong

*B*1 ≡ *B*' *, B*'

1 1

—→ *B*' *, B*' ≡ *B*2

2 2

*B*1 —→ *B*2

Table C.1 Reduction Rules in L(*P* ).

* *B* ∈ У: the proof is trivial, since У is closed under —→ and 6*P* ∈ У, *P* =

*Unbrace*(*P* ).

* *B* = {*P* }*σ*: this case is not possible, since {*P* }*σ* /—→ .
* *B* = *out* | *in.B*'': we consider *in.B*'' = *x*(*y*)*.B*'' and *out* = *x*¯*z*, since the other combinations can be proven similarly. Then *B* —→ *B*''{*z/y*}. We also have *Unbrace*(*B*)= *out* | *in.Unbrace*(*B*'') —→ *Unbrace*(*B*''{*z/y*}).
* Cases *B* = (*νx*)*B*' —→ (*νx*)*B*'' and *B* = *B*1 | *B*2 —→ *B*' | *B*2 can be proven by

1

induction hypothesis on *B*' and on *B*1, respectively.

* Case *B* ≡ *B*1 —→ *B*2 is trivial, since ≡ is preserved by *Unbrace*.

Now we prove item (ii). We proceed by induction on the depth of the reduction

*Unbrace*(*B*)= *T* —→ *R*, assuming *Pref*(*B*).

* *T* = *out* | *in.T* '': we consider *in.T* '' = *x*(*y*)*.T* '' and *out* =

*x*¯*z*, since the other

combinations can be proven similarly. Then *T* = *out* | *in.T* '' —→ *T* ''{*z/y*}. Define *B* = *out* | *in.B*'', where *Unbrace*(*B*'')= *T* ''. Then *Unbrace*(*B*''{*z/y*}) = *T* ''{*z/y*} and *B* —→ *B*''{*z/y*}.

* Cases *T* = (*νx*)*T* ' —→ (*νx*)*T* '' and *T* = *T*1 | *T*2 —→ *T* ' | *T*2 can be proven by

1

induction hypothesis.

* Case *T* ≡ *T*1 —→ *T*2 is trivial, since ≡ is preserved by *Unbrace*.

Lemma C.2 Let *B*1*, B*2 ∈ L(*P* ) such that *B*1 ≡ *B*2. Then *min*(*B*1)= *min*(*B*2).

Proof. Only axiom !*B* ≡ *B* | !*B* can look difficult to prove. Other axioms are trivial. If *B* ∈ У, *min*(!*B*) = *min*(*B* | !*B*) = +∞. Suppose *B* /∈ У. Then we have *min*(!*B*)= *min*(*B*) and *min*(*B* | !*B*)= min{*min*(*B*)*, min*(!*B*)} = *min*(*B*).

*Negative results for PAπ*

Proposition C.3 Let *P* ∈ PA*π*, *B* ∈ L(*P* ), such that *Pr*(*B*). Then E*B*' ∈ L(*P* ) such that *B* —→ *B*' and *Pr*(*B*').

Proof. To prove the statement we proceed by induction on the depth of the deriva- tion of *Pr*(*B*). We recall that L(*P* ) is closed under —→ and that the cases *B* ∈У and *B* = {*P* }*σ* are not possible, since *Pr*(*B*) implies *Pref*(*B*), i.e. *min*(*B*) ∈ [1*..* + ∞).

* + *B* =!*x*¯*z* | !*x*(*y*)*.B*'', where *min*(!*x*(*y*)*.B*'') ≥ 2: then *B* —→ *B*' = *B*''{*z/y*}| *B*. Since *Pr*(*B*), it follows *Pr*(*B*').
  + Cases *B* = (*νx*)*B*', *B* = *B*1|*B*2 and *B* ≡ *B*1 can be proven by induction hypothesis on *B*' and on *B*1, assuming *Pr*(*B*'), *B*' —→ *B*'' and *Pr*(*B*'), and *Pr*(*B*1), *B*1 —→ *B*'

1

and *Pr*(*B*' ), respectively.

1

Proposition C.4 Let *P* ∈ PA*π*, *B* ∈ L(*P* ) such that *Pr*(*B*). Then there exists a non-empty maximal computation from *B*

*B* = *B*0 —→ *B*1 —→ *B*2 —→ *...* —→ *Bi*[ —→ *.. .*]

such that 6*i* ≥ 0*, Pref*(*Bi*).

Proof. By Proposition [C.3](#_bookmark54), E*B*1 ∈ L(*P* ) such that *B* —→ *B*1 and *Pr*(*B*1). Now it suffices to iterate, noticing that 6*i* ≥ 0, *Pr*(*Bi*) implies *Pref*(*Bi*).

Lemma C.5 Let **[**· **]** be an encoding from A*π* into PA*π* that satisfies:

1. compositionality w.r.t. input prefix,
2. must-preservation,
3. **[***ω.*0]] —*ω*→ .

Then 6*x, y, z* ∈ U , *Pr*([[*x*¯*z* | *x*¯*z*]] | *Cx*(*y*)[*Cx*(*y*){[[*ω.*0]]}]).

Proof. First, we prove that **[***x*¯*z* | *x*¯*z*]] | *Cx*(*y*)[*Cx*(*y*){[[*ω.*0]]}] —→ . By contradic- tion, suppose **[***x*¯*z* | *x*¯*z*]] | *Cx*(*y*)[*Cx*(*y*){[[*ω.*0]]}] /—→ . By Lemma [B.2](#_bookmark52), every hole

is prefixed in *C*[[·]]

*x*(*y*)

. This implies that **[***x*¯*z* | *x*¯*z*]] | *Cx*(*y*)

[*Cx*(*y*)

{[[*ω.*0]]}] —

*ω*→, that is

[[*x*¯*z* | *x*¯*z*]] | *C*

*x*(*y*)

[*Cx*(*y*)

[[[*ω.*0]]]] —→ and **[***x*¯*z* | *x*¯*z*]] | *Cx*(*y*)

[*Cx*(*y*)

[[[*ω.*0]]]] /—*ω*→. It means

that **[***x*¯*z* | *x*¯*z*]] *must* [[*x*(*y*)*.x*(*y*)*.ω.*0]], contradicting that *x*¯*z* | *x*¯*z must x*(*y*)*.x*(*y*)*.ω.*0.

Since every hole is prefixed in *C*[[·]]

*x*(*y*)

, we have that *Cx*(*y*)[*Cx*(*y*){[[*ω.*0]]}] ≡

⟨!*h*(*k*)*.C*[!*h*(*k*)*.C*'{[[*ω.*0]]}| *T* '] | *T* ⟩, where *T* ∈ PA*π* and *h* /∈ ⟨·⟩ (otherwise, we could

not unprefix one occurrence of **[***ω.*0]]). Since **[***x*¯*z* | *x*¯*z*]] | *Cx*(*y*)[*Cx*(*y*){[[*ω.*0]]}] —→ , it follows that **[***x*¯*z* | *x*¯*z*]] | *Cx*(*y*)[*Cx*(*y*){[[*ω.*0]]}] ≡ ⟨!*h*¯*k* | !*h*(*k*)*.C*[!*h*(*k*)*.C*'{[[*ω.*0]]}| *T* '] | *T* ''⟩.

Since *min*(!*h*(*k*)*.C*[!*h*(*k*)*.C*'{[[*ω.*0]]}]) ≥ 2, *Pr*([[*x*¯*z* | *x*¯*z*]] | *Cx*(*y*)[*Cx*(*y*){[[*ω.*0]]}]) holds.

Theorem C.6 Let **[**· **]** be an encoding from A*π* into PA*π* that satisfies:

1. compositionality w.r.t. input prefix,
2. **[***ω.*0]] —*ω*→ .

Then **[**· **]** is not must-preserving.

Proof. By contradiction, suppose **[**· **]** is must-preserving. Let *P* = *x*¯*z* | *x*¯*z* and

*o* = *x*(*y*)*.x*(*y*)*.ω.*0. Consider **[***P* ]] | [[*o*]] = [[*x*¯*z* | *x*¯*z*]] | [[*x*(*y*)*.x*(*y*)*.ω.*0]]. By Lemma

[C.5](#_bookmark56), *Pr*([[*x*¯*z* | *x*¯*z*]] | *Cx*(*y*)[*Cx*(*y*){[[*ω.*0]]}]). Hence *Pref*([[*x*¯*z* | *x*¯*z*]] | *Cx*(*y*)[*Cx*(*y*){[[*ω.*0]]}]).

By Proposition [C.4](#_bookmark55), there exists a non-empty maximal computation from *B* = [[*x*¯*z* | *x*¯*z*]] | *Cx*(*y*)[*Cx*(*y*){[[*ω.*0]]}]

*B* = *B*0 —→ *B*1 —→ *B*2 —→ *...* —→ *Bi*[ —→ *.. .*]

such that 6*i* ≥ 0*, Pref*(*Bi*). As a consequence, 6*i* ≥ 0*, Bi* /—*ω*→. It follows that there exists a maximal computation from *Unbrace*(*B*)= *P* | *o*

*Unbrace*(*B*)= *T*0 —→ *T*1 —→ *T*2 —→ *...* —→ *Ti*[ —→ *.. .*]

such that 6*i* ≥ 0*, Ti* /—*ω*→. This means that *P must o*, contradicting the must- preservation hypothesis of **[**·]].

*Negative results for PIAπ*

*B*1 ∈ L(*P* ) *, B*{*z/y*} = *C*1[!*x*(*y*)*.B*1]*, x* /∈ *bn*(*C*1)

P

*P* (!*xi*(*y*)*.B, x*)

*P* (!*x*(*y*)*.B, x*)

*Prin*(*B, x*) *, x* /= *y*

Base Res

*Prin*(!*x*(*y*)*.B, x*) *Prin*((*νy*)*B, x*)

*Prin*(*B*'*, x*) *, B*' ≡ *B*

Cong

Par1

*Prin*(*B*1*, x*) *, P* (!*xi*(*yi*)*.Bi, x*)

*Prin*(*B, x*)

*Prin*(*B*1*, x*) *, min*(!*x*(*yi*)*.Bi*)=1

Par2

*Prin*(*B*1 | !*x*(*yi*)*.Bi, x*)

*Prin*(*B*1 | !*xi*(*yi*)*.Bi, x*)

*Prin*(*B*1*, x*) *, B*2 ∈ PIA*π*

Par3

*Prin*(*B*1 | *B*2*, x*)

Table C.2 Predicate *Prin*.

Lemma C.7 Let *P* ∈ PIA*π*, *B* ∈ L(*P* ). E*x* ∈ *fn*(*B*) such that *Prin*(*B, x*) iff

*a b c*

*B* ≡ *N* (*x*)= (*νy*1)*..*(*νym*)( !*x*(*yi*)*.Bi* | !*x*(*yj*)*.Bj* | !*xh*(*yh*)*.Bh* | *T* )

*i*=1

*j*=1

*h*=1

where *a* ≥ 1, *m, b, c* ≥ 0, 6*k* ∈ [1*..m*] *x* /= *yk*, 6*i* ∈ [1*..a*] *P* (!*x*(*yi*)*.Bi, x*), 6*j* ∈

[1*..b*] *min*(!*x*(*yj*)*.Bj*)= 1, 6*h* ∈ [1*..c*] *xh* /= *x* and *P* (!*xh*(*yh*)*.Bh, x*), and *T* ∈ PIA*π*.

Proof. Consider the *if* implication: given the term *N* (*x*), *x* ∈ *fn*(*N* (*x*)) and *Prin*(*N* (*x*)*, x*) hold. For the *only if* implication it suffices to prove that for each rule in Table [C.2](#_bookmark57) (unless Rule P) the postcondition can be written, via ≡, as *N* (*x*).

Lemma C.8 Let **[**· **]** be an encoding from A*π* into PIA*π* that satisfies:

1. compositionality w.r.t. input prefix,
2. must-preservation,
3. **[***ω.*0]] —*ω*→ .

Then *fn*(*C*[[·]]

*x*(*y*)

) /= ∅.

Proof. Suppose *fn*(*C*[[·]]

*x*(*y*)

) = ∅. Since **[***x*¯*z*]] *must Cx*(*y*)[[[*ω.*0]]], it follows that 6*A*

such that *C*

*x*(*y*)

[[[*ω.*0]]] —→ *...* —→ *A*, *fn*(*A*)*/*{*ω*} = ∅ and *A* —→ *...* —→ *A*' —*ω*→.

Then *Cx*(*y*)[[[0]]] *must Cx*(*y*)[[[*ω.*0]]], i.e. *x*(*y*)*.*0 *must x*(*y*)*.ω.*0. It is a contradiction of

the must-preservation hypothesis on **[**·]].

Lemma C.9 Let **[**· **]** be an encoding from A*π* into PIA*π* that satisfies:

* 1. compositionality w.r.t. input prefix,
  2. must-preservation,
  3. **[***ω.*0]] —*ω*→ .

Then *Cx*(*y*)[*Cx*(*y*)[·]] ≡ ⟨!*u*(*v*)*.C*[!*u*'(*v*)*.C*'[·] | *T* '] | *T* ⟩.

Proof. It follows immediately from Lemma [B.2](#_bookmark52).

Lemma [C.10](#_bookmark59) and Proposition [C.11](#_bookmark60) are useful to prove Proposition [C.12](#_bookmark61).

Lemma C.10 Let *P* ∈ PIA*π*, *B* ∈ L(*P* ). E*x* ∈ *fn*(*B*): *Prin*(*B, x*) implies *Pref*(*B*).

Proof. Trivial.

Proposition C.11 Let *P* ∈ PIA*π*, *B* ∈ L(*P* ), E*x* ∈ *fn*(*B*) such that *Prin*(*B, x*) and *B* —→ *B*' for some *B*' ∈ L(*P* ). Then E*B*'' ∈ L(*P* ) such that *B* —→ *B*'', *x* ∈ *fn*(*B*'') and *Prin*(*B*''*, x*).

Proof. By Lemma [C.7](#_bookmark58), *B* can be written in the normal form *N* (*x*) as in Lemma [C.7](#_bookmark58). By operational Rule Cong, we consider *N* (*x*) —→ *B*'' ≡ *B*', for some *B*'' ∈ L(*P* ). We can suppose to apply *α*- equivalence in such a way 6*a, b* ∈ *bn*(*N* (*x*))*,a* /= *b* and 6*a* ∈ *bn*(*N* (*x*)) and 6*b* ∈ *fn*(*N* (*x*)), *a* /= *b*. We distinguish four cases:

* + 1. *T* —→ *T* ': trivial;
    2. *T* ≡ *x*¯*z* | *T* ' and !*x*(*yi*)*.Bi* | *T* —→ *Bi*{*z/yi*}| !*x*(*yi*)*.Bi* | *T* ' for some *i* ∈ [1*..a*] and *a* ≥ 1: without loss of generality, suppose that *i* = 1 and there is only one hole in *C*1 in Rule P of Table [C.2](#_bookmark57). Since *B*1{*z/y*1} = *C*1[!*x*(*y*)*.B*' ] and *x* /∈ *bn*(*C*1),

1

the case *min*(*B*' {*z/y*1}) ≥ 2 implies that *B*1{*z/y*1} ≡ ⟨!*α*(*β*)*.C*' [!*x*(*y*)*.B*' ] | *T* ''⟩,

1 1 1

*x* /∈ ⟨·⟩, *x* /∈ *bn*(*C*' ) and *x* /∈ *bn*(*T* '') (either *α* = *x* or *α* /= *x*), while the case

1

*min*(*B*1{*z/y*1}) = 1 implies that *B*1{*z/y*1} ≡ ⟨!*x*(*y*)*.B*' | *T* ''⟩, where *x* /∈ ⟨·⟩,

1

*x* /∈ *bn*(*T* '') and *T* '' ∈ PIA*π* in both cases. It is possible to prove that in both cases *B*'' can be written ina normal form *N* ''(*x*), i.e. *x* ∈ *fn*(*B*'') and *Prin*(*B*''*, x*).

* + 1. *T* ≡ *x*¯*z* | *T* ' and !*x*(*yj*)*.Bj* | *T* —→ *Bj*{*z/yj* }| !*x*(*yj*)*.Bj* | *T* ' for some *j* ∈ [1*..b*] and *b* ≥ 0: without loss of generality, suppose *j* = 1. Since there is at least one

!*x*(*yi*)*.Bi* for some *i* ∈ [1*..a*] (being *a* ≥ 1), we can replace this reduction with the reduction from !*x*(*yi*)*.Bi* | *T* , considered in item (b).

* + 1. *T* ≡ *x*¯*hz*|*T* ' and !*xh*(*yh*)*.Bh*|*T* —→ *Bh*{*z/yh*}|!*xh*(*yh*)*.Bh*|*T* ' for some *h* ∈ [1*..c*] and *c* ≥ 0: without loss of generality, suppose *h* = 1 and there is only one hole in *C*1 in Rule P of Table [C.2](#_bookmark57). We recall that *x*1 /= *x*. Since *B*1{*z/y*1} = *C*1[!*x*(*y*)*.B*' ] and

1

*x* /∈ *bn*(*C*1), *min*(*B*' {*z/y*1}) ≥ 2 implies *B*' {*z/y*1}≡ ⟨!*α*(*β*)*.C*' [!*x*(*y*)*.B*' ]|*T* ''⟩, *x* /∈

1 1 1 1

⟨·⟩, *x* /∈ *bn*(*C*' ) and *x* /∈ *bn*(*T* '') (either *α* = *x* or *α* /= *x*), while *min*(*B*1{*z/y*1})= 1 implies *B*1{*z/y*1}≡ ⟨!*x*(*y*)*.B*' | *T* ''⟩, where *x* /∈ ⟨·⟩, *x* /∈ *bn*(*T* '') and *T* '' ∈ PIA*π* in both cases. As in item (a), applying ≡ it is possible to prove that in both cases *B*'' can be written in a normal form *N* ''(*x*), i.e. *x* ∈ *fn*(*B*'') and *Prin*(*B*''*, x*).

1

1

Proposition C.12 Let *P* ∈ PIA*π*, *B* ∈ L(*P* ), E*x* ∈ *fn*(*B*) such that *Prin*(*B, x*). Then there exists a maximal computation from *B* (also empty)

*B* = *B*0 —→ *B*1 —→ *B*2 —→ *...* —→ *Bi*[ —→ *.. .*]

such that 6*i* ≥ 0*, Pref*(*Bi*).

Below, we prove that, under reasonable conditions, there exists a term satisfying the predicate *Prin* (Lemma [C.13](#_bookmark62)), and finally the impossibility result for PIA*π* (Theorem [C.14](#_bookmark63)).

Lemma C.13 Let **[**· **]** be an encoding from A*π* into PIA*π* that satisfies:

1. compositionality w.r.t. input prefix,
2. must-preservation,
3. **[***ω.*0]] —*ω*→ ,
4. if *fn*(*P* ) ∩ *bn*(*x*(*y*)) = ∅ then *fn*([[*P* ]]) ∩ *bn*(*C*[[·]]

*x*(*y*)

(Preservation of independence wrt input prefix)

1. **[***x*(*y*)*.P* ]] ≡ ⟨!*u*(*v*)*.C*[[[*P* ]]] | *T* ⟩, where *u* /∈ ⟨·⟩*.*

)= ∅*,*

Then E*h* ∈ *fn*(*C*[[·]]

*x*(*y*)

): *Prin*(*Cx*(*y*)[*Cx*(*y*){[[*ω.*0]]}]*, h*).

Proof. From (1), we know that *Cx*(*y*)[*Cx*(*y*){[[*ω.*0]]}] ≡ ⟨!*u*(*v*)*.C*[[[*x*(*y*)*.ω.*0]]] | *T* ⟩ ≡

⟨!*u*(*v*)*.C*[⟨!*u*(*v*)*.C*[[[*ω.*0]]] | *T* '⟩] | *T* ⟩, where *u* is a free name in both cases (in the more external case by (5) and in the internal case by (4)). Then we can verify that for *u*

in *fn*(*C*[[·]] ), *Prin*(*Cx*(*y*)[*Cx*(*y*){[[*ω.*0]]}]*, h*) holds.

*x*(*y*)

Theorem C.14 Let **[**· **]** be an encoding from A*π* into PIA*π* that satisfies:

1. compositionality w.r.t. input prefix,
2. **[***ω.*0]] —*ω*→ ,
3. E*h* ∈ *fn*(*C*[[·]]

*x*(*y*)

): *Prin*(*Cx*(*y*)[*Cx*(*y*){[[*ω.*0]]}]*, h*).

Then **[**· **]** is not must-preserving.

Proof. By contradiction, suppose **[**· **]** being must-preserving. We can apply *α*- equivalence to **[***x*¯*z* | *x*¯*z*]] | *Cx*(*y*)[*Cx*(*y*){[[*ω.*0]]}] in such a way to avoid collision among bound/free names. By (3), we have that *h* ∈ *fn*(*Cx*(*y*)[*Cx*(*y*)[·]]), and by Table [C.2](#_bookmark57), *Prin*([[*x*¯*z* | *x*¯*z*]] | *Cx*(*y*)[*Cx*(*y*){[[*ω.*0]]}]*, h*) holds. Moreover, we can prove that [[*x*¯*z* | *x*¯*z*]] | *Cx*(*y*)[*Cx*(*y*){[[*ω.*0]]}] —→ . **[***x*¯*z* | *x*¯*z*]] | *Cx*(*y*)[*Cx*(*y*){[[*ω.*0]]}] —→ would

imply **[***x*¯*z* | *x*¯*z*]] *must Cx*(*y*)[*Cx*(*y*){[[*ω.*0]]}], i.e. *x*¯*z* | *x*¯*z must x*(*y*)*.x*(*y*)*.ω.*0, contradict-

ing the must-preservation hypothesis on **[**· **]**. Now, it suffices to apply Proposition

[C.12](#_bookmark61).

*Negative results for POAπ*

The following theorem states a negative result for the 0-adic versions of A*π* and POA*π* (denoted resp. by ACCS and POAACCS). It can be reformulated for A*π* and POA*π* by imposing some syntactic restrictions to both source and target language. The hypotheses are quite strong, in particular (3) and (4). However, they are rea- sonable for acknowledgment-based encodings, where two partners **[***a*(*y*)*.P* ]] and [[*a*¯*z*]] start a communication protocol on a well-known channel *x* and keep on commu- nicating by means of private channels. Although an encoding could easily violate the above conditions, this result is important since, differently from the previous ones, no form of divergence is either introduced or hidden, i.e. the must-preserving property is violated without taking into any divergence hypothesis.

Theorem C.15 Let **[**· **]** be an encoding from ACCS into POAACCS that satisfies:

1. compositionality w.r.t. input prefix,
2. **[***ω.*0]] —*ω*→ ,
3. 6*a* ∈ U , |*fn*(*C*[[·]])| = |*fn*([[*a*¯]])| = 1;

*a*

1. *x* ∈ *fn*(*K*) implies #fn(*x, K*) = 1, where *K* ∈ {*C*[[·]]*,* [[*a*¯]]} and #fn(*x, K*) denotes

*a*

the number of free occurrences of *x* in *K*.

Then **[**· **]** is not must-preserving.

Proof. In the following, (!)*P* denotes both !*P* and *P* . By contradiction, suppose

[[· **]** is must-preserving. By Lemma [B.2](#_bookmark52), every hole is prefixed in *C*[[·]]. It follows that 6*a* ∈ U and 6*j* ≥ 1, **[**(*a*¯)*j*]] | *Ca*[[[*ω.*0]]] —→ , where (*a*¯)*j* denotes the parallel composition of *j* copies of *a*¯: if **[**(*a*¯)*j*]] | *Ca*[[[*ω.*0]]] / —→ , [[(*a*¯)*j*]] *must* [[*a.ω.*0]], i.e. (*a*¯)*j must a.ω.*0, contradicting the must-preservation hypothesis on **[**· **]**. It follows

*a*

that both *fn*(*Ca*[·]) and *fn*([[(*a*¯)*j* **]**) are not empty sets, i.e. item (3) is well-defined.

We can write **[**(*a*¯)*j*]] and *C*[[·]]

*a*

as follows:

1. either **[**(*a*¯)*j*]] ≡ (*νx*1)*..*(*νxm*)(!*x*¯ | *B*0), where *fn*(*B*0)= ∅ and 6*i, x* /= *xi*

and one of the following configurations

a *Ca*[·] ≡ (*νx*1)*..*(*νxn*)(*x.G*1[·] | *B*1),

b *Ca*[·] ≡ (*νx*1)*..*(*νxn*)(!*x.G*1[·] | *B*1),

c *Ca*[·] ≡ (*νx*1)*..*(*νxn*)(*x.C*1[·] | *B*1),

d *Ca*[·] ≡ (*νx*1)*..*(*νxn*)(!*x.C*1[·] | *B*1),

e *Ca*[·] ≡ (*νx*1)*..*(*νxn*)(*x.B*1 | *G*1[·]),

f *Ca*[·] ≡ (*νx*1)*..*(*νxn*)(!*x.B*1 | *G*1[·]),

where 6*i, x* /= *xi*, *fn*(*G*1)= *fn*(*C*1)= *fn*(*B*1)*/*{*ω*} = ∅, every hole is prefixed in *G*1 and not in *C*1,

1. or *Ca*[·] ≡ (*νx*1)*..*(*νxm*)(!*x*¯ | *G*2[·]), where *fn*(*G*2)= ∅, 6*i, x* /= *xi* and every hole is prefixed in *G*1

and one of the following configurations

a **[**(*a*¯)*j*]] ≡ (*νx*1)*..*(*νxn*)(*x.B*1 | *B*2),

b **[**(*a*¯)*j*]] ≡ (*νx*1)*..*(*νxn*)(!*x.B*1 | *B*2), where 6*i, x* /= *xi*, *fn*(*B*1)= *fn*(*B*2)= ∅.

In the cases of (1-a), (1-b), (1-e), (1-f), (2-a) and (2-b), we can deduce that [[*a*¯]]*must* [[*a.ω.*0]], implying that *a*¯*must a.ω.*0. This contradicts the must-preservation hypothesis on **[**·]].

Consider the case (1-c). This implies (up to *α*-equivalence) **[***a*¯]] | [[*a.a.ω.*0]] ≡

⟨!*x*¯ | *x.C*1[⟨*x.C*1[[[*ω.*0]]] | *B*1⟩] | *B*0 | *B*1⟩ —→ ⟨!*x*¯ | *C*1[⟨*x.C*1[[[*ω.*0]]] | *B*1⟩] | *B*0 | *B*1⟩ ≡

⟨!*x*¯ | (!)(*x.C*1[[[*ω.*0]]] | *B*1) | *B*2 | *B*0 | *B*1⟩. Suppose (!)(*x.C*1[[[*ω.*0]]] | *B*1)= *x.C*1[[[*ω.*0]]] | *B*1

(the other case is similar). It follows that ⟨!*x*¯ | *x.C*1[[[*ω.*0]]] | *B*1 | *B*2 | *B*0 | *B*1⟩ —→

⟨!*x*¯ | *C*1[[[*ω.*0]]] | *B*1 | *B*2 | *B*0 | *B*1⟩ ≡ ⟨!*x*¯ | [[*ω.*0]] | *B*2 | *B*1 | *B*2 | *B*0 | *B*1⟩. This implies that **[***a*¯]] *must* [[*a.a.ω.*0]], that is *a*¯ *must a.a.ω.*0, contradicting the must-preservation hypothesis on **[**·]].

The case (1-d) implies that **[***a*¯ | *a*¯]] *must* [[*a.a.ω.*0]], that is *a*¯ | *a*¯ *must a.a.ω.*0, again contradicting the must-preservation hypothesis on **[**·]].