Available online at www.sciencedirect.com
ScienceDirect
Electronic Notes in Theoretical Computer Science
ELSEVIER
Electronic Notes in Theoretical Computer Science 249 (2009) 269–286
www.elsevier.com/locate/entcs

On the Relative Expressive Power of Calculi for Mobility

Daniele Gorla

Dip. di Informatica, Univ. di Roma "La Sapienza"

Abstract

In this paper, we comparatively analyze some mainstream calculi for mobility: asynchronous π-calculus, distributed π-calculus and Mobile/Boxed/Safe ambients. In particular, we focus on their relative expressive power, i.e. we try to encode one in the other while respecting some reasonable properties. According to the possibility or the impossibility for such results, we set up a hierarchy of these languages.

Keywords: Expressiveness, Encodings, Mobility, Process Calculi.

1 Introduction

In the last ten years, one of the main research lines in the field of concurrency theory has been the development of new formalisms, paradigms and environments that better model distributed and mobile systems. Several terms have been coined to name this fortunate research line (network-aware programming, WAN/global/ubiquitous computing, ...) that has attracted many computer scientists around the world. In this scenario, the term *mobility* has become the reference keyword to denote several possible dynamic evolutions of systems, ranging from name mobility to mobile computation and mobile computing. A lot of efforts have been spent to define more and more sophisticated type theories, behavioural equivalences and implementations for these calculi. What is still missing is, from the practical side, real-life applications where the distinctive features of such formalisms are essential and, from the theoretical side, an exhaustive comparative analysis of all these proposals, from a linguistic perspective.

In this paper, we approach the last issue by comparing some mainstream calculi for mobility: asynchronous π-calculus (written π_a) [15,1], distributed π-calculus (written Dπ) [14], and Mobile/Boxed/Safe ambients (written MA/BA/SA) [7,2,17]. Our results formally prove some claims informally appeared in the literature or differently prove results already known. Moreover, for the sake of systematization, we also consider and compare languages that, to the best of our knowledge, have

doi:10.1016/j.entcs.2009.07.094

never been contrasted, neither informally.

To this aim, we compare languages via their *relative expressive power*, i.e. we try to encode one language in another, while respecting some reasonable properties. Of course, the definition of such properties is crucial for the meaningfulness of our study: too liberal properties would lead to poorly informative results (most of the encodings would be possible), whereas too stringent properties would be fruitless (very few encodings would be possible).

In principle, a good encoding should satisfy at least two properties: *compositionality* (the encoding of a compound term must be expressed in terms of the encoding of its components) and *faithfulness* (the encoding of a term must have exactly the same functionalities as the original term). There are different ways to formalize these notions; mainly for the second one, a number of different proposals have been considered in the literature (e.g., sensitiveness to barbs/divergence/deadlock, operational correspondence, full abstraction, ...). Here, we take the proposal in [13] and consider only encodings that satisfy the following properties: *compositionality*, *name invariance* (i.e., the encodings of two source processes that differ only in their free names must only differ in the associated free names), *operational correspondence* (i.e., computations of a source term must correspond to computations in the encoded term, and vice versa), *divergence reflection* (i.e., terminating processes must be translated into terminating processes) and *success sensitiveness* (i.e., successful terms – for some notion of success – must be translated into successful terms, and vice versa). In [13] we show that these criteria form a valid proposal for language comparison: most of the encodings in the literature respect them (so our notion is consistent with the common understanding of the community), but there still exist encodings that do not satisfy them (so our notion is non-trivial); moreover, the best known separation results can be proved (in a much easier and uniform way) by relying on our proposal. Here, we furthermore vindicate the validity of our proposal by showing that some widely believed (but never formally proved) separation results can be established by relying on the above mentioned criteria.

Our results are summarized in the side figure. There, we put an arrow from $\mathcal{L}_1$ to $\mathcal{L}_2$ if $\mathcal{L}_1$ can be encoded in $\mathcal{L}_2$ but not vice versa (the dashed arrow denotes an encodability result slightly weaker than the solid arrow). We say that $\mathcal{L}_2$ is *more expressive than* $\mathcal{L}_1$ if there is a sequence of arrows from $\mathcal{L}_1$ to $\mathcal{L}_2$; $\mathcal{L}_1$ and $\mathcal{L}_2$ are *incomparable* if neither $\mathcal{L}_1$ is more expressive than $\mathcal{L}_2$ nor vice versa.

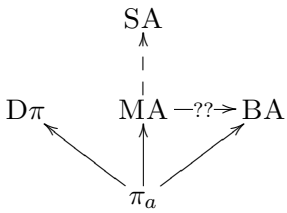


Figure 1

Some of our results are expectable: for example, we confirm that π_a is the minimal common denominator of calculi for mobility, since it can be encoded in all the languages considered. However, though expectable, some results turned out very difficult to prove. For example,

to encode π_a in MA we had to develop quite a complex encoding since one of our criteria is operational correspondence: what we propose is, to the best of our knowledge, the first encoding that does not introduce any 'spurious' computation in the encoding of a π_a process. Ruling out computations that are not present in the source process is a sensible task when dealing with MA, because of the high possibility of interferences between MA processes. A simpler encoding of π_a is possible, e.g., in SA (see [17]), because the latter language is "more controlled" than MA. Another issue that turned out surprisingly difficult to prove is the encodability of MA in BA, that we believe *not to* hold; this is the only conjecture that this paper leaves open.

This paper is organized as follows. In Section 2, we formally present the syntax and operational semantics of the languages considered. In Section 3, we recall from [13] the properties that encodings should satisfy. In Section 4, we formally build up our hierarchy: for every pair of languages, we give an encodability/separation result. Finally, in Section 5 we conclude the paper by also mentioning some related work. In this extended abstract we have omitted the motivations behind the languages; for more details, see the original papers where such languages were introduced.

2 The Languages

A *process calculus* is a triple $\mathcal{L} = (\mathcal{P}, \longmapsto, \simeq)$, where

- $\mathcal{P}$ is the set of language terms, usually called *processes* and ranged over by $P, Q, R, \ldots$. All the process calculi we are going to consider have a common core syntax given by:

$$P \ ::= \ \mathbf{0} \ \Big| \ (\nu n)P \ \Big| \ P_1|P_2 \ \Big| \ !P \ \Big| \ \surd$$

  As usual, $\mathbf{0}$ is the terminated process, whereas $\surd$ denotes success (see the discussion on Property 5 in Section 3); $P_1|P_2$ denotes the parallel composition of two processes; $(\nu n)P$ restricts to P the visibility of n and binds n in P; finally, $!P$ denotes the replication of process P. We have assumed here a very simple way of modeling recursive processes; all what we are going to prove does not rely on this choice and can be rephrased under different forms of recursion.

- $\longmapsto$ is the operational semantics, needed to specify how a process computes; following common trends in process calculi, we specify the operational semantics by means of *reductions*, whose inference rules shared by all our process calculi are:

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

  where $\mathcal{E}(\cdot)$ denotes an evaluation context and '$\equiv$' denotes structural equivalence (used to equate different ways of writing the same process). Of course, the operational axioms, the evaluation contexts and structural equivalence are peculiar to every language. As usual, $\Longmapsto$ denotes the reflexive and transitive closure of $\longmapsto$.

- $\simeq$ is a behavioural equivalence/preorder, needed to describe the abstract behaviour of a process; usually, $\simeq$ is a congruence/precongruence at least with respect to parallel composition.

We now present the syntax and reduction-based operational semantics of the specific calculi considered in this paper. In what follows, we assume a countable set of names, $\mathcal{N}$, ranged over by $a, b, c, \ldots, l, k, \ldots, m, n, \ldots, u, v, w, \ldots, x, y, z, \ldots$ and their decorated versions. To simplify reading, we shall use $a, b, c, \ldots$ to denote channels, $l, k, \ldots$ to denote localities, $m, n, \ldots$ to denote ambients and $x, y, z, \ldots$ to denote input variables; $u, v, w, \ldots$ are used to denote generic names (channels and variables in π_a; channels, localities and variables in Dπ; ambients and variables in MA, SA and BA).

2.1 The asynchronous π-calculus (π_a)

We consider the asynchronous version of the π-calculus, as defined in [1]. This language is nowadays widely considered the minimal common denominator of calculi for mobility, it is a good compromise between expressiveness and simplicity, and it also has a good implementation [24]. Its syntax extends the common syntax of processes by letting

$$P \quad ::= \quad \ldots \;\Big|\; \bar{u}\langle v\rangle \;\Big|\; u(x).P$$

Intuitively, $\bar{u}\langle v\rangle$ represents message v unleashed along channel u. Dually, $u(x).P$ waits for some message from channel u and, once received, replaces with such a message every occurrence of variable x in P. Processes $u(x).P$ and $(\nu a)P$ bind x and a in P, respectively; a name occurring in P that is not bound is called free. Consequently, we define the free and bound names of a process P, written $fn(P)$ and $bn(P)$; alpha-conversion is then defined accordingly.

Evaluation contexts are defined as follows:

$$\mathcal{E}(\cdot) \quad ::= \quad \cdot \;\Big|\; \mathcal{E}(\cdot) \,|\, P \;\Big|\; P \,|\, \mathcal{E}(\cdot) \;\Big|\; (\nu n)\mathcal{E}(\cdot)$$

The *structural equivalence* relation, $\equiv$, is the least equivalence on processes closed by evaluation contexts, including alpha-conversion and satisfying the following axioms:

$$P|\mathbf{0} \equiv P \qquad P_1|P_2 \equiv P_2|P_1 \qquad P_1|(P_2|P_3) \equiv (P_1|P_2)|P_3$$

$$!P \equiv P|!P \qquad (\nu a)\mathbf{0} \equiv \mathbf{0} \qquad (\nu a)(\nu b)P \equiv (\nu b)(\nu a)P$$

$$P_1|(\nu a)P_2 \equiv (\nu a)(P_1|P_2) \;\text{ if }\; a \notin fn(P_1)$$

The *reduction relation*, $\longmapsto$, is the least relation on processes closed by the inference rules described above and satisfying the following axiom:

$$a(x).P \mid \bar{a}\langle b\rangle \longmapsto P\{{}^{b}/_{x}\}$$

where $P\{^b/_x\}$ denotes the capture-avoiding substitution of each occurrence of x in P with an occurrence of b.

2.2 Distributed π-calculus (Dπ)

We present a slightly simplified version of [14]; mainly, we elided typing information from the syntax. The main syntactic entity are *nets*, that are collections of *located processes*, possibly sharing restricted names:

$$N \ ::= \ \mathbf{0} \ \Big| \ l : P \ \Big| \ N|N \ \Big| \ (\nu u)N$$

Processes are obtained from the common syntax by letting

$$P \ ::= \ \ldots \ \Big| \ u(x).P \ \Big| \ \bar{u}\langle v\rangle.P \ \Big| \ go\_u.P$$

The main differences between Dπ and π_a are: processes and channels are located at a specified locality; communication can only happen between co-located processes and, hence, there is a primitive to let processes migrate between localities (viz. action go_u); finally, communication is synchronous (i.e., it blocks both the sending and the receiving process).

Since the main syntactic entity is the set of nets, evaluation contexts, reductions and structural equivalence will be given for nets.

$$\mathcal{E}(\cdot) \ ::= \ \cdot \ \Big| \ \mathcal{E}(\cdot) \,|\, N \ \Big| \ N \,|\, \mathcal{E}(\cdot) \ \Big| \ (\nu n)\mathcal{E}(\cdot)$$

The structural axioms are:

$$l : P|\mathbf{0} \equiv l : P \qquad l : P_1|P_2 \equiv l : P_1 \mid \ l : P_2 \qquad l : !P \equiv l : P|!P$$

$$N|\mathbf{0} \equiv N \qquad N_1|N_2 \equiv N_2|N_1 \qquad N_1|(N_2|N_3) \equiv (N_1|N_2)|N_3$$

$$(\nu u)(\nu w)N \equiv (\nu w)(\nu u)N \qquad (\nu n)\mathbf{0} \equiv \mathbf{0} \qquad l : (\nu u)P \equiv (\nu u)l : P \ \text{ if } \ u \neq l$$

$$(\nu l)N \equiv (\nu l)(N \mid l : \mathbf{0}) \qquad N_1|(\nu u)N_2 \equiv (\nu u)(N_1|N_2) \ \text{ if } \ u \notin fn(N_1)$$

The reduction axioms are:

$$l : a(x).P \mid \ l : \bar{a}\langle b\rangle.Q \longmapsto l : P\{^b/_x\} \mid \ l : Q$$

$$l : go\_l'.P \mid l' : \mathbf{0} \longmapsto l : \mathbf{0} \mid l' : P$$

Notice that a migration at l' is legal only if l' is an existing locality of the net. In the original paper [14], this check was carried out, among other tasks, by the type system. We prefer the present formulation for the sake of simplicity; however, all what are going to prove does not rely on this choice.

2.3 Mobile Ambients (MA)

We consider the Ambient calculus as presented in [7].

$$P ::= \ldots \mid (x).P \mid \langle M \rangle \mid M.P \mid u[P]$$
$$M ::= u \mid in\_u \mid out\_u \mid open\_u \mid M.M$$

MA is somewhat related to Dπ in the sense that processes are located within ambients (viz. $u[P]$) and only co-located processes can communicate via a monadic, asynchronous and anonymous communication: $(x).P$ represents the anonymous input prefix, whereas $\langle M \rangle$ represents the asynchronous and anonymous output particle, where message M can be not only a raw name but also a sequence of actions. However, differently from Dπ, entire ambients can move: an ambient n can enter into a sibling ambient m via the in_m action or exit from the enclosing ambient m via the out_m action. Moreover, an ambient n can be opened via the $open\_n$ action.

Evaluation contexts are defined as follows:

$$\mathcal{E}(\cdot) \quad ::= \quad \cdot \mid \mathcal{E}(\cdot) \mid P \mid P \mid \mathcal{E}(\cdot) \mid (\nu n)\mathcal{E}(\cdot) \mid n[\mathcal{E}(\cdot)]$$

The structural equivalence relation extends structural equivalence of π_a with the following axioms:

$$(M.M').P \equiv M.(M'.P) \qquad m[(\nu n)P] \equiv (\nu n)m[P] \quad \text{if } n \neq m$$

The reduction axioms are:

$$n[in\_m.P_1 | P_2] \mid m[P_3] \longmapsto m[P_3 \mid n[P_1 | P_2]] \qquad open\_n.P_1 \mid n[P_2] \longmapsto P_1 \mid P_2$$

$$m[n[out\_m.P_1 | P_2] \mid P_3] \longmapsto n[P_1 | P_2] \mid m[P_3] \qquad (x).P \mid \langle M \rangle \longmapsto P\{^{M}/_{x}\}$$

MA, like all the following Ambient-like languages, strongly relies on a type system to avoid inconsistent processes like, e.g., $m.P$ or $in\_n[P]$; these two processes can arise after the (ill-typed) communications $(x).x.P \mid \langle m \rangle$ and $(x).x[P] \mid \langle in\_n \rangle$. For MA, like for SA and BA, we shall always consider the sub-language formed by all the well-typed processes, as defined in [6,17,2].

2.4 Safe Ambients (SA)

We consider the Safe Ambient calculus as presented in [17]. SA extends MA by adding *co-actions*, though which ambient movements/openings must be authorized by the target ambient. Hence, the syntax of SA is the same as MA's, with

$$M ::= \ldots \mid \overline{in}\_u \mid \overline{out}\_u \mid \overline{open}\_u$$

Evaluation contexts and structural equivalence are the same as for MA; the reduction axioms are:

$$(x).P \mid \langle M\rangle \longmapsto P\{^{M}/_{x}\} \qquad open\_n.P_1 \mid n[\overline{open}\_n.P_2|P_3] \longmapsto P_1 \mid P_2 \mid P_3$$

$$n[in\_m.P_1|P_2] \mid m[\overline{in}\_m.P_3|P_4] \longmapsto m[P_3 \mid P_4 \mid n[P_1|P_2]]$$

$$m[n[out\_m.P_1|P_2] \mid \overline{out}\_m.P_3 \mid P_4] \longmapsto n[P_1|P_2] \mid m[P_3|P_4]$$

*2.5 Boxed Ambients (*BA*)*

We consider the Boxed Ambient calculus as presented in [2]. BA evolves MA by removing the *open* action that is considered too powerful and, hence, potentially dangerous. To let different ambients communicate, BA allows a restricted form of non-local communication: in particular, every input/output action can be performed locally (if tagged with direction $\star$), towards the enclosing ambient (if tagged with direction $\uparrow\uparrow$) or towards an enclosed ambient n (if tagged with direction n).

$$P ::= \ldots \mid (x)^{\eta}.P \mid \langle M\rangle^{\eta}.P \mid M.P \mid u[P]$$

$$M ::= u \mid in\_u \mid out\_u \mid M.M \qquad\qquad \eta ::= \star \mid {\uparrow\uparrow} \mid u$$

Evaluation contexts and structural equivalence are the same as for MA; the reduction axioms are:

$$n[in\_m.P_1|P_2] \mid m[P_3] \longmapsto m[P_3 \mid n[P_1|P_2]]$$

$$m[n[out\_m.P_1|P_2] \mid P_3] \longmapsto n[P_1|P_2] \mid m[P_3]$$

$$(x)^{\star}.P_1 \mid \langle M\rangle^{\star}.P_2 \longmapsto P_1\{^{M}/_{x}\} \mid P_2$$

$$(x)^{\star}.P_1 \mid n[\langle M\rangle^{\uparrow\uparrow}.P_2|P_3] \longmapsto P_1\{^{M}/_{x}\} \mid n[P_2|P_3]$$

$$(x)^{n}.P_1 \mid n[\langle M\rangle^{\star}.P_2|P_3] \longmapsto P_1\{^{M}/_{x}\} \mid n[P_2|P_3]$$

$$\langle M\rangle^{\star}.P_1 \mid n[(x)^{\uparrow\uparrow}.P_2|P_3] \longmapsto P_1 \mid n[P_2\{^{M}/_{x}\}|P_3]$$

$$\langle M\rangle^{n}.P_1 \mid n[(x)^{\star}.P_2|P_3] \longmapsto P_1 \mid n[P_2\{^{M}/_{x}\}|P_3]$$

3 Properties of Encodings

A *translation* of $\mathcal{L}_1 = (\mathcal{P}_1, \longmapsto_1, \simeq_1)$ into $\mathcal{L}_2 = (\mathcal{P}_2, \longmapsto_2, \simeq_2)$, written $[\![\cdot]\!] : \mathcal{L}_1 \to \mathcal{L}_2$, is a function from $\mathcal{P}_1$ to $\mathcal{P}_2$. We shall call *encoding* any translation that satisfies the five properties we are going to present now. There, to simplify reading, we let S range over processes of the source language (viz., $\mathcal{L}_1$) and T range over processes of the target language (viz., $\mathcal{L}_2$).

As already said in the introduction, an encoding should be compositional. To formally define this notion, we exploit the notion of *k-ary context*, written $\mathcal{C}(\__1;\ldots;\__k)$, that is a term where k occurrences of $\mathbf{0}$ are replaced by the k holes $\__1,\ldots,\__k$.

Property 1 *A translation* $[\![\cdot]\!] : \mathcal{L}_1 \to \mathcal{L}_2$ *is* compositional *if, for every k-ary* $\mathcal{L}_1$*-operator* $\mathtt{op}$ *and finite subset of names* N*, there exists a k-ary context* $\mathcal{C}^N_{\mathtt{op}}[\__1;\ldots;\__k]$ *such that* $[\![\,\mathtt{op}(S_1,\ldots,S_k)\,]\!] = \mathcal{C}^N_{\mathtt{op}}[[\![\,S_1\,]\!];\ldots;[\![\,S_k\,]\!]]$*, for every* $S_1,\ldots,S_k$ *with* $fn(S_1,\ldots,S_k) = N$*.*

Moreover, a good encoding should reflect in the encoded term all the name substitutions carried out in the source term. However, it is possible that an encoding fixes some names to play a precise rôle or it can map a single name into a tuple of names. In general, every encoding assumes a *renaming policy* $\varphi_{[\![\,]\!]} : \mathcal{N} \longrightarrow \mathcal{N}^k$ that is a function such that $\forall u, v \in \mathcal{N}$ with $u \neq v$, it holds that $\varphi_{[\![\,]\!]}(u) \cap \varphi_{[\![\,]\!]}(v) = \emptyset$ (where $\varphi_{[\![\,]\!]}(\cdot)$ is simply considered a set here). We extend the application of a substitution to sequences of names in the expected way.

Property 2 *A translation* $[\![\cdot]\!] : \mathcal{L}_1 \to \mathcal{L}_2$ *is* name invariant *if, for every substitution* σ*, it holds that*

$$[\![\,S\sigma\,]\!] \begin{cases} = \; [\![\,S\,]\!]\sigma' & \text{if } \sigma \text{ is injective} \\ \simeq_2 \; [\![\,S\,]\!]\sigma' & \text{otherwise} \end{cases}$$

where σ' *is the substitution such that* $\varphi_{[\![\,]\!]}(\sigma(a)) = \sigma'(\varphi_{[\![\,]\!]}(a))$*.*

Injectivity of σ must be taken into account because non-injective substitutions can fuse two distinct names, and this matters because compositionality also depends on the free names occurring in the encoded terms. For more discussion, see [13].

A source term and its encoding should have the same operational behaviour, i.e. all the computations of the source term must be preserved by the encoding without introducing "new" computations. This intuition is formalized as follows.

Property 3 *A translation* $[\![\cdot]\!] : \mathcal{L}_1 \to \mathcal{L}_2$ *is* operationally corresponding *if*

- *for every* S *and* S' *such that* $S \Longmapsto_1 S'$*, it holds that* $[\![\,S\,]\!] \Longmapsto_2 \simeq_2 [\![S']\!]$*;*
- *for every* S *and* T *such that* $[\![\,S\,]\!] \Longmapsto_2 T$*, there exists* S' *such that* $S \Longmapsto_1 S'$ *and* $T \Longmapsto_2 \simeq_2 [\![\,S'\,]\!]$*.*

An important semantic issue that an encoding should avoid is the introduction of infinite computations, written $\longmapsto^\omega$.

Property 4 *A translation* $[\![\cdot]\!] : \mathcal{L}_1 \to \mathcal{L}_2$ *is* divergence reflecting *whenever* $[\![\,S\,]\!] \longmapsto^\omega_2$ *implies that* $S \longmapsto^\omega_1$*, for every* S*.*

Finally, we require that the source and the translated term behave in the same way with respect to success, a notion that can be used to define sensible semantic theories [9,25]. To formulate our property in a simple way, we follow the approach in [25] and assume that all the languages contain the same success process $\surd$; then, we define the predicate $\Downarrow$, meaning reducibility (in some modality, e.g. may/must/fair-must) to a process containing a top-level unguarded occurrence of $\surd$. Clearly,

different modalities in general lead to different results; in this paper, proofs will be carried out in a 'may' modality. Finally, for the sake of coherence, we require the notion of success be caught by the semantic theory underlying the calculi, viz. $\simeq$; in particular, we assume that $\simeq$ never relates two processes P and Q such that $P \Downarrow$ and $Q \not\Downarrow$.

Property 5 *A translation* $[\![\cdot]\!] : \mathcal{L}_1 \to \mathcal{L}_2$ *is* success sensitive *if, for every* S*, it holds that* $S \Downarrow$ *iff* $[\![S]\!] \Downarrow$.

3.1 Derived Properties

The results we are going to prove (both negative and positive) rely on the choice of equivalence/preorder in the target language, viz. '$\simeq_2$'. Some translations become encodings (i.e., they satisfy all the properties just listed) only under some choices of '$\simeq_2$': a representative sample is given in Theorem 4.5. Similarly, some separation results can be proved under specific assumptions on '$\simeq_2$', in the sense that the proof we provide only works under such assumptions. Indeed, in [13] we show that some separation results can be proved in the general framework but, to carry out more proofs, we have to slightly specialize the framework. In particular, in *loc.cit.* we have considered three alternative settings:

(i) $\simeq_2$ is *exact*, i.e. $T \simeq_2 T'$ and T performs an action imply that T' can (weakly) perform the same action as well; moreover, parallel composition must be translated homomorphically, i.e. for every $N \subset \mathcal{N}$ it holds that $\mathcal{C}_{|}^{N}[\_1; \_2] = \_1 \mid \_2$;

(ii) $\simeq_2$ is *reduction sensitive*, i.e. $T \simeq_2 T'$ and $T' \longmapsto_2$ imply that $T \longmapsto_2$;

(iii) the occurrences of $\simeq_2$ in Property 3 are restricted to pairs of kind $((\nu\widetilde{n})([\![S']\!] \mid T'), [\![S']\!])$, for any $\widetilde{n}$ and T' such that $(\nu\widetilde{n})([\![S']\!] \mid T') \simeq_2 [\![S']\!]$.

All these assumptions are discussed and justified at length in [13]. By relying on them, we can prove a number of auxiliary results that will be useful in carrying out the main proofs of the paper.

Proposition 3.1 *Let* $[\![\cdot]\!]$ *be an encoding and assume that we are in setting (ii) or (iii); then,* $S \not\longmapsto_1$ *implies that* $[\![S]\!] \not\longmapsto_2$.

Proposition 3.2 *Let* $[\![\cdot]\!]$ *be an encoding and assume that we are in setting (ii) or (iii); if there exist two source terms* S_1 *and* S_2 *such that* $S_1 \mid S_2 \Downarrow$, $S_1 \not\Downarrow$ *and* $S_2 \not\Downarrow$, *then* $[\![S_1 \mid S_2]\!] \longmapsto_2$.

Proposition 3.3 *Let* $[\![\cdot]\!] : \mathcal{L}_1 \to \mathcal{L}_2$ *be an encoding and assume that we are in setting (ii) or (iii). If there exists two source terms* S_1 *and* S_2 *that do not reduce but such that* $[\![S_1 \mid S_2]\!] \longmapsto_2$, *then*

(i) *if* $\mathcal{L}_2 \in \{\pi_a, \mathrm{D}\pi\}$, *it can only be that* $[\![S_1]\!] \mid [\![S_2]\!] \longmapsto_2$;

(ii) *if* $\mathcal{L}_2 \in \{\mathrm{MA}, \mathrm{BA}, \mathrm{SA}\}$, *it can only be that* $\mathcal{C}_1([\![S_1]\!]) \mid \mathcal{C}_2([\![S_2]\!]) \longmapsto_2$, *where* $\mathcal{C}_{|}^{fn(S_1,S_2)}[\_1; \_2]$, *i.e. the context used to compositionally translate* $S_1 \mid S_2$, *is of the form* $\mathcal{E}(\mathcal{C}_1(\_1) \mid \mathcal{C}_2(\_2))$ *for some evaluation context* $\mathcal{E}(\cdot)$ *and two contexts* $\mathcal{C}_1(\cdot)$ *and* $\mathcal{C}_2(\cdot)$ *that are either empty (viz.,* $\cdot$*) or a single top-level ambient*

containing a top-level hole (viz., $m[\cdot \mid R]$).

Theorem 3.4 *Assume that there is a $\mathcal{L}_1$-process S such that $S \not\longmapsto_1$, $S \not\Downarrow$ and $S \mid S \Downarrow$; moreover, assume that every $\mathcal{L}_2$-process T that does not reduce is such that $T \mid T \not\longmapsto_2$. Then, there cannot exist any encoding $[\![\cdot]\!] : \mathcal{L}_1 \to \mathcal{L}_2$ under none of the settings (i), (ii) and (iii).*

To state the following proof-technique, let us define the *matching degree* of a language $\mathcal{L}$, written $\text{MD}(\mathcal{L})$, as the greatest number of names that must be matched to yield a reduction in $\mathcal{L}$. For example, the matching degree of MA is 1, whereas the matching degree of Dπ is 2.

Theorem 3.5 *If $\text{MD}(\mathcal{L}_1) > \text{MD}(\mathcal{L}_2)$, then there exists no encoding $[\![\cdot]\!] : \mathcal{L}_1 \to \mathcal{L}_2$ under settings (ii) nor (iii).*

A new derived property, not needed for the results in [13], is the following one.

Proposition 3.6 *Let $[\![\cdot]\!] : \mathcal{L}_1 \to \mathcal{L}_2$ be a translation that satisfies Property 2; for every S and $n \notin fn(S)$, it holds that $\varphi_{[\![\,]\!]}(n) \cap fn([\![S]\!]) = \emptyset$.*

Proof. By contradiction, let $n' \in \varphi_{[\![\,]\!]}(n) \cap fn([\![S]\!])$. Let m be such that $m \notin fn(S)$ and $\varphi_{[\![\,]\!]}(m) \cap fn([\![S]\!]) = \emptyset$; moreover, let σ be the permutation that swaps m and n. Trivially, $S = S\sigma$ and, hence, $[\![S]\!] = [\![S\sigma]\!]$. However, by Property 2, $[\![S\sigma]\!] = [\![S]\!]\sigma'$, for σ' that swaps $\varphi_{[\![\,]\!]}(m)$ and $\varphi_{[\![\,]\!]}(n)$ component-wise. The only possible way to have that $[\![S]\!] = [\![S]\!]\sigma'$ (that holds because of transitivity) is to have $dom(\sigma') \cap fn([\![S]\!]) = \emptyset$ that, however, does not hold, because $dom(\sigma') = \varphi_{[\![\,]\!]}(n) \cup \varphi_{[\![\,]\!]}(m)$ and $n' \in \varphi_{[\![\,]\!]}(n) \cap fn([\![S]\!])$: contradiction. □

It is worth noting that all negative results we are going to prove hold under settings (ii) and (iii); on the contrary, only some of them also hold in setting (i). All the three settings have advantages and disadvantages. In our opinion, the better compromise is setting (iii) that, in spite of being very 'syntactical', exactly captures the intuition underlying the occurrences of '$\simeq_2$' in Property 3: to garbage collect junk processes left by the encodings. Indeed, it is the only setting that is satisfied by all the encodings we are aware of. Clearly, a challenging direction for future work consists in finding other settings that enable proofs but are more liberal than the present ones.

4 The Hierarchy, bottom-up

We now give the results underlying the hierarchy of calculi for mobility depicted in Figure 1.

Theorem 4.1 *There exists no encoding of* MA, SA *and* BA *in* Dπ *and* π_a.

Proof. Corollary of Theorem 3.4:

- On one hand, notice that

- if T is a π_a-process such that $T \mid T \longmapsto_2$, then $T \equiv (\nu\widetilde{n})(a(x).P \mid \bar{a}\langle b\rangle \mid T')$, for some a not in $\widetilde{n}$. Thus, trivially, $T \longmapsto_2$; hence, every π_a-process T that does not reduce is such that $T \mid T \not\longmapsto_2$.
- if T is a Dπ-net such that $T \mid T \longmapsto_2$, then either $T \equiv (\nu\widetilde{n})(l : a(x).P \mid l : \bar{a}\langle b\rangle.Q \mid T')$ or $T \equiv (\nu\widetilde{n})(l : go\_l'.P \mid l' : \mathbf{0} \mid T')$, for some a, l and l' not in $\widetilde{n}$. Thus, trivially, $T \longmapsto_2$; hence, every Dπ-net T that does not reduce is such that $T \mid T \not\longmapsto_2$.

• On the other hand, we can find in MA, SA and BA a process S that does not reduce and does not report success, but such that $S \mid S$ reports success: it suffices to let S be
- $(\nu p)(open\_p.\surd \mid n[in\_n.p[out\_n.out\_n]])$. in MA and BA;
- $(\nu p)(open\_p.\surd \mid n[in\_n.p[out\_n.out\_n.\overline{open}\_p] \mid \overline{in}\_n \mid \overline{out}\_n])$ in SA.

□

Theorem 4.2 *There exists no encoding of* Dπ *in* SA, BA, MA *and* π_a.

Proof. Corollary of Theorem 3.5, since $\text{MD}(\text{D}\pi) = 2$ whereas $\text{MD}(\text{MA}) = \text{MD}(\text{SA}) = \text{MD}(\text{BA}) = \text{MD}(\pi_a) = 1$. □

Theorem 4.3 Dπ *and* BA *are more expressive than* π_a.

Proof. Because of Theorems 4.1 and 4.2, it suffices to encode π_a in Dπ and BA.

Clearly, π_a can be trivially encoded in Dπ: it suffices to co-locate the π_a process in a pre-defined locality hosting all the channels needed. This translation is an encoding under any choice of $\simeq_2$.

BA can encode π_a: [2] provides an encoding of π_a in BA that satisfies all the properties of Section 3, with operational correspondence that holds up to any (pre)congruence that satisfies the law $(\nu n)n[P] \simeq \mathbf{0}$ whenever $k \notin fn(P)$ (e.g., strong barbed equivalence). It is defined as a homomorphism w.r.t. all the operators, except for

$$[\![\, u(x).P \,]\!] \triangleq (x)^u.[\![\, P \,]\!] \qquad [\![\, \bar{u}\langle v\rangle \,]\!] \triangleq (\nu k)(u[\langle v\rangle^\star.in\_k] \mid k[\mathbf{0}]) \quad \text{for } k \text{ fresh}$$

□

Theorem 4.4 MA *is more expressive than* π_a.

Proof. Because of Theorem 4.1, MA cannot be encoded in π_a. We are left with proving that π_a can be encoded in MA; this is not a trivial task, if we want to satisfy all the properties in Section 3 with operational correspondence that holds up to strong barbed equivalence. Indeed, in several papers [7,6,5] there are attempts to encode π_a in MA, but none of them satisfies Property 3.

The encoding relies on a renaming policy that maps every name a to a triple of pairwise distinct names (a_1, a_2, a_3); it is a homomorphism w.r.t. all the operators,

except for restrictions, inputs and outputs, that are translated as follows:

$$[\![(\nu a)P]\!] \triangleq (\nu\, a_1, a_2, a_3)[\![P]\!]$$

$$[\![\bar{a}\langle b\rangle]\!] \triangleq a_1[a_2[open\_a_3.\langle b_1, b_2, b_3\rangle]]$$

$$\begin{aligned}[\![a(x).P]\!] \triangleq\ & open\_a_1.(\nu p, q)(open\_p \\ & \mid\ a_3[in\_a_2.open\_\texttt{rest}\ \mid\ (x_1, x_2, x_3).in\_q.p[out\_q.[\![P]\!]]] \\ & \mid\ q[open\_a_2.\texttt{rest}[!\,\texttt{rest}[in\_a_3.out\_q.in\_a_2.open\_\texttt{rest}]]])\end{aligned}$$

for p and q fresh

where (x_1, x_2, x_3) is a shortcut for $(x_1).open\_\texttt{pol}.(x_2).open\_\texttt{pol}.(x_3)$ and $\langle b_1, b_2, b_3\rangle$ is a shortcut for $\langle b_1\rangle \mid \texttt{pol}[\langle b_2\rangle \mid \texttt{pol}[\langle b_3\rangle]]$, with `pol` a reserved name for implementing polyadic communications. The proof that the translation just presented satisfies all the properties listed in Section 3 is omitted for space reasons; the interested reader can find it in [10]. □

Our encoding follows the philosophy underlying the encoding of π_a in BA; however, MA misses the parent-child communication of BA, used to translate an input action. Thus, for every communication along a, the ambient named a_3 is used as a 'pilot' ambient to enter a_2 and consume the datum associated to b. To reflect the fact that an output along a can be consumed only once, we exploit the outer ambient a_1 and the corresponding $open\_a_1$ action. However, interferences can still arise from independent communications along channel a: several a_3-ambients can enter into the same a_2-ambient. In this case, only one of them is opened and the remaining ones must be rolled back, i.e. reappear at top-level, ready to enter another a_2-ambient. Such a rolling back is done by opening a_2 in a restricted ambient q and by leading all the not consumed a_3-ambients out from q via the reserved ambient `rest`, that also restores the in_a_2 capability.

Theorem 4.5 SA *is more expressive than* MA*:* SA *can be encoded in* MA*, whereas there exists no encoding of* SA *in* MA*.*

Proof. In [17] MA is translated into SA by mapping all the operators homomorphically, except for $[\![u[P]]\!] \triangleq u[!\,\overline{in}\_u \mid !\,\overline{out}\_u \mid !\,\overline{open}\_u \mid [\![P]\!]]$. Such an encoding enjoys operational correspondence up to (strong/weak) barbed equivalence restricted to translated contexts (written $\simeq^{tr}$). Indeed, the MA process $open\_n \mid n[\mathbf{0}]$ reduces to $\mathbf{0}$, whereas its encoding can only reduce to $!\,\overline{in}\_n \mid !\,\overline{out}\_n \mid !\,\overline{open}\_n$ and the latter process is only *translated* barbed equivalent [1] to the encoding of $\mathbf{0}$ (viz.,

[1] The idea is that, if placed within an ambient different from n, $!\,\overline{in}\_n \mid !\,\overline{out}\_n \mid !\,\overline{open}\_n$ has the same effect as $\mathbf{0}$. If placed (possibly, after some reduction steps) within an ambient n, $!\,\overline{in}\_n \mid !\,\overline{out}\_n \mid !\,\overline{open}\_n$ is useless (and, again, has the same effect as $\mathbf{0}$): since we are working in a translated setting (and hence every ambient has replicated occurrences of all the co-capabilities involving its name), an identical process is already present within n and can be absorbed thanks to Milner's law $!P \simeq^{tr} !P \mid !P$.

0 itself).

On the contrary, SA cannot be encoded in MA; this is proved by contradiction. Consider the pair of SA processes $P \triangleq n[in\_n.\langle m\rangle]$ and $Q \triangleq n[\overline{in}\_n.(m[out\_n.\overline{open}\_m.\surd] \mid \overline{out}\_n)] \mid open\_m$, for $n \neq m$; by Proposition 3.2, $[\![P \mid Q]\!]$ must reduce and, because of Proposition 3.3 and the definition of the reduction rules for MA, it can only be that

(i) either $\mathcal{C}_1([\![P]\!])$ exhibits a top-level ambient n' and $\mathcal{C}_2([\![Q]\!])$ wants to enter/open it;

(ii) or $\mathcal{C}_2([\![Q]\!])$ exhibits a top-level ambient n' and $\mathcal{C}_1([\![P]\!])$ wants to enter/open it.

for some context $\mathcal{C}_1(\cdot)$ and $\mathcal{C}_2(\cdot)$ that are empty or have a single top-level ambient containing a top-level hole. Notice that the reduction cannot happen because of a communication otherwise, by Property 2, $[\![m[in\_m.\langle n\rangle] \mid Q]\!]$ would reduce, against Proposition 3.1. For the same reason, it must be that $n' \in \varphi_{[\![\ ]\!]}(n)$.

We now prove that both cases are impossible; without loss of generality, assume that we fall in case (i), since case (ii) is similar. First, notice that $\mathcal{C}_1(\cdot)$ must be empty: if it was not, we would have that $[\![n[out\_n.\langle m\rangle] \mid Q]\!] \longmapsto$ (recall that $\mathcal{C}_1(\cdot)$ is part of $\mathcal{C}_{|}^{\{n,m\}}[\_1;\_2]$, the context used to encode parallel composition of processes with free names $\{n, m\}$; so, it only depends on parallel composition and such names). Thus, we have that $[\![P]\!]$ exhibits the top-level ambient n'; but also this leads to a contradiction. Indeed, by Property 1, it holds that $[\![P]\!] \triangleq \mathcal{C}_{n[\ ]}^{\{n,m\}}([\![in\_n.\langle m\rangle]\!])$; so, the ambient named n' can be exhibited either by $\mathcal{C}_{n[\ ]}^{\{n,m\}}(\cdot)$ or by $[\![in\_n.\langle m\rangle]\!]$ (and, hence, $\mathcal{C}_{n[\ ]}^{\{n,m\}}(\cdot)$ has a top-level hole). In both cases, we can contradict Proposition 3.1: in the first case, we would have that $[\![n[out\_n.\langle m\rangle] \mid Q]\!] \longmapsto$; in the second case, we would have that $[\![in\_n.\langle m\rangle \mid Q]\!] \longmapsto$. □

We want to remark that SA can encode MA also in a different way, that allows us to formulate operational correspondence up to standard (i.e., not just translated) barbed equivalence. To this aim, we have to consider a *family* of encodings $[\![\cdot]\!]_N$, for $N \subset \mathcal{N}$, with the idea that a MA process P can be encoded via $[\![\cdot]\!]_N$ only if $fn(P) \subseteq N$. For every N, $[\![\cdot]\!]_N$ is a homomorphism for all operators, except for

$$[\![\mathbf{0}]\!]_N \triangleq P_N \qquad [\![u[P]]\!]_N \triangleq u[P_N \mid [\![P]\!]_N]$$

$$[\![(\nu n)P]\!]_N \triangleq (\nu n)[\![P]\!]_{N\cup\{n\}} \qquad [\![(x).P]\!]_N \triangleq (x).[\![P]\!]_{N\cup\{x\}}$$

where $P_N \triangleq \prod_{n\in N} !\,\overline{in}\_n \mid !\,\overline{out}\_n \mid !\,\overline{open}\_n$. By exploiting the equivalence $!P \simeq !P \mid !P$, it is easy to check that now operational correspondence holds up to $\simeq$.

It is however worth noting that such a kind of *parameterized* encoding (and, similarly, those proposed by [19,16]) is not considered in our framework because

If we did not restrict our attention to translated contexts, we could use $n[\cdot]$ as a distinguishing context in SA for the two processes **0** and $!\,\overline{in}\_n \mid !\,\overline{out}\_n \mid !\,\overline{open}\_n$.

it would make difficult to formulate our properties and carry out proofs without knowing what the index represents. For example, having a parameterized encoding $[\![\cdot]\!]_\Xi$, which is the initial (i.e., top-level) value of Ξ in $[\![\cdot]\!]_\Xi$? Moreover, even assuming that Ξ are names (as in the translation of SA in MA just presented), are they names in the source or in the target language? The latter question is very delicate: in the first case, Property 2 should be adapted by requiring that $[\![S\sigma]\!]_{\Xi\sigma}$ is equal/equivalent to $([\![S]\!]_\Xi)\sigma'$; in the second case, we have that $[\![S\sigma]\!]_{\Xi\sigma'}$ must be equal/equivalent to $([\![S]\!]_\Xi)\sigma'$. Thus, even if we believe that such an enhanced form of encoding is reasonable, we have problems in adapting our framework without specifying anything on the index. On the contrary, we would find it very strange to make too many assumptions on the index, in general.

Nevertheless, we believe that both the argument shown in the proof of Theorem 4.5 and this further possible encoding give the feeling that SA can be encoded in MA. This fact justifies the dashed arrow placed in Figure 1 between SA and MA: there should be some kind of arrow from MA into SA, but it seemed us fair to distinguish such an arrow from the other ones in the picture.

Theorem 4.6 *There exists no encoding of* BA *in* SA *and* MA.

Proof. We show the non-encodability of BA in SA, since the result for BA in MA is simpler. Consider the processes $(x)^n.\surd$ and $n[\langle b\rangle^\star]$, for $n \neq b$. Because of Proposition 3.2, $[\![(x)^n.\surd \mid n[\langle b\rangle^\star]]\!]$ must reduce and, because of Propositions 3.3 and of the reduction rules for SA, this can only happen because:

(i) either $\mathcal{C}_1([\![(x)^n.\surd]\!])$ wants to enter into an ambient named n' and $\mathcal{C}_2([\![n[\langle b\rangle^\star]]\!])$ allows such an entrance;

(ii) or $\mathcal{C}_2([\![n[\langle b\rangle^\star]]\!])$ wants to enter into an ambient named n' and $\mathcal{C}_1([\![(x)^n.\surd]\!])$ allows such an entrance;

(iii) or $\mathcal{C}_1([\![(x)^n.\surd]\!])$ wants to open an ambient named n' and $\mathcal{C}_2([\![n[\langle b\rangle^\star]]\!])$ allows such an opening;

(iv) or $\mathcal{C}_2([\![n[\langle b\rangle^\star]]\!])$ wants to open an ambient named n' and $\mathcal{C}_1([\![(x)^n.\surd]\!])$ allows such an opening.

Indeed, $\mathcal{C}_1([\![(x)^n.\surd]\!])$ and $\mathcal{C}_2([\![n[\langle b\rangle^\star]]\!])$ cannot perform a communication, otherwise, by Property 2, $[\![(x)^n.\surd \mid b[\langle n\rangle^\star]]\!]$ would reduce; for the same reason, it must be that $n' \in \varphi_{[\![\,]\!]}(n)$.

However, we now prove that all the cases depicted above lead to contradict Proposition 3.1. Let us consider $\mathcal{C}_2([\![n[\langle b\rangle^\star]]\!])$ in all cases. If $\mathcal{C}_2(\cdot)$ is empty we can work as follows. First, observe that $[\![n[\langle b\rangle^\star]]\!] \triangleq \mathcal{C}^{\{b\}}_{n[\,]}([\![\langle b\rangle^\star]\!])$; if the action is produced by $\mathcal{C}^{\{b\}}_{n[\,]}(\cdot)$ alone, also $[\![n[\langle b\rangle^{\uparrow\uparrow}]]\!]$ would produce the same action and so $[\![(x)^n.\surd \mid n[\langle b\rangle^{\uparrow\uparrow}]]\!]$ would reduce; similarly, if the action is produced by $[\![\langle b\rangle^\star]\!]$ alone, $[\![(x)^n.\surd \mid \langle b\rangle^\star]\!]$ would reduce. Hence, the action is produced both by $\mathcal{C}^{\{b\}}_{n[\,]}(\cdot)$ and $[\![\langle b\rangle^\star]\!]$; this means that $\mathcal{C}^{\{b\}}_{n[\,]}(\cdot)$ provides an ambient and $[\![\langle b\rangle^\star]\!]$ performs the action $\overline{in}\_n'$, in_n' or $\overline{open}\_n'$. However, the latter fact is not possible because otherwise $n' \in fn([\![\langle b\rangle^\star]\!])$, in contradiction with Proposition 3.6. So, it must be

that $\mathcal{C}_2(\cdot)$ is not empty; this rules out case (iv) above and imposes that $[\![n[\langle b\rangle^\star]]\!]$ performs the action $\overline{in}\_n'$, in_n' or $\overline{open}\_n'$, respectively. We then work like in the case in which $\mathcal{C}_2(\cdot)$ is empty to prove that there is no way for $[\![n[\langle b\rangle^\star]]\!]$ to perform such an action without contradicting Proposition 3.1. □

Theorem 4.7 *There exists no encoding of* SA *in* BA.

Proof. By contradiction. First, consider the pair of SA processes $P \triangleq n[in\_n.\langle m\rangle]$ and $Q \triangleq n[\overline{in}\_n.(m[out\_n.\overline{open}\_m] \mid \overline{out}\_n)] \mid open\_m.\surd$, for $n \neq m$; by Propositions 3.2 and 3.3, and by definition of the reduction rules for BA, it must be that

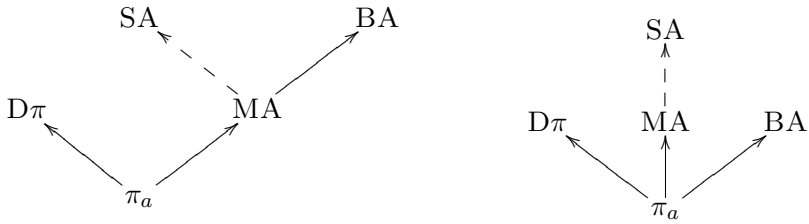
(i) either $\mathcal{C}_1([\![P]\!])$ exhibits an ambient n' and $\mathcal{C}_2([\![Q]\!])$ wants to enter in it, or vice versa;

(ii) or $\mathcal{C}_1([\![P]\!])$ sends a message to a sibling ambient n' and $\mathcal{C}_2([\![Q]\!])$ is ready to locally receive a message, or vice versa;

(iii) or $\mathcal{C}_1([\![P]\!])$ waits for a message from a sibling ambient n' and $\mathcal{C}_2([\![Q]\!])$ locally sends some message, or vice versa.

In all cases, $n' \in \varphi_{[\![\,]\!]}(n)$. We now prove that the three cases above all lead to a contradiction: the first case is formally identical to the proof of Theorem 4.5; the second and the third case are similar, so we only work out case (ii). First, notice that $\mathcal{C}_1(\cdot)$ must be empty: if it were not, it must be a single ambient and, hence, it could not send/receive any message. So, the action is produced either by $\mathcal{C}^{\{n,m\}}_{n[\,]}(\cdot)$ or by $[\![in\_n.\langle m\rangle]\!]$. In both cases, we can contradict Proposition 3.1: in the first case, we would have that $[\![n[in\_m.\langle n\rangle] \mid Q]\!] \longmapsto$; in the second case, we would have that $[\![in\_n.\langle m\rangle \mid Q]\!] \longmapsto$. □

To complete the hierarchy of Figure 1, it would suffice to prove that there exists no encoding of MA in BA. Surprisingly, we have not been able to prove such a (quite expectable) result; we leave it open as a conjecture. Thus, there are only two possibilities for resolving the '??' in Figure 1:



according to whether MA is encodable in BA or not. We strongly believe that the right one should hold, even if we still have not been able to prove it.

5 Conclusions and Related Work

We have comparatively studied some mainstream calculi for mobility, namely the asynchronous π-calculus, a distributed π-calculus and Mobile/Safe/Boxed Ambi-

Encoding	Encodability	In this paper	In other papers
π_a in Dπ	✓	Thm 4.3	? (though common sense)
π_a in MA	✓	Thm 4.4	[7,6,5]: not satisfactory
π_a in SA	✓	Thms 4.4 + 4.5	[17]
π_a in BA	✓	Thm 4.3	[2]
Dπ in π_a	×	Thm 4.2	[4]
Dπ in MA	×	Thm 4.2	?
Dπ in SA	×	Thm 4.2	?
Dπ in BA	×	Thm 4.2	?
MA in π_a	×	Thm 4.1	[22]
MA in Dπ	×	Thm 4.1	?
MA in SA	✓	Thm 4.5	[17]: not satisfactory
MA in BA	??	?	?
SA in π_a	×	Thm 4.1	[22]
SA in Dπ	×	Thm 4.1	?
SA in MA	×	Thm 4.5	?
SA in BA	×	Thm 4.7	?
BA in π_a	×	Thm 4.1	[22]
BA in Dπ	×	Thm 4.1	?
BA in MA	×	Thm 4.6	?
BA in SA	×	Thm 4.6	?

Table 1
Overview of all our results: '✓' stands for a possibility result, '×' stands for an impossibility result, '??' stands for an open question.

ents. We have organized all these languages in a clear hierarchy based on their relative expressive power. In [11], we also extend our analysis to various dialects of MA/BA/SA appeared in the literature in the last years and compare every dialect with the language it comes from.

The results of this paper are summed up in Table 1 and are compared with analogous results from the literature ('?' stands for absence – to the best of our knowledge – of any formal claim already appeared). To obtain such results, we have exploited the set of criteria presented and discussed in [13].

To the best of our knowledge, our encoding of π_a in MA is the first one that satisfies operational correspondence. It is quite complex (the encoding of a single communication in π_a requires 14 reduction steps in MA) because some ingenuity is needed to handle the possible interferences that can arise between the encoding of different actions on the same channel. Notice that the encoding of π_a in SA [17] is simpler (just 5 reductions to mimic a single communication), since co-actions can be exploited to reduce such interferences; this is a further evidence of SA's expressive power. Moreover, the encoding of π_a in BA [2] is even simpler: thanks to parent-child communications, just 2 reductions are needed to mimic a single communication. These facts suggest that co-actions and, even more, remote communications are more suitable to implement channel-based communications in ambient-based languages.

To conclude, we want to mention some strictly related results. First, [26] provides an encoding of the synchronous (choice-free) π-calculus in 'pure' SA, i.e. SA without communications. The encoding enjoys all our properties, with homomorphism w.r.t. '|' that holds only for processes without free names; otherwise, it is simply compositional, because it introduces a channel-handler for every free name of the translated process. Second, [16] provides an encoding of (a variant of) BA in (a variant of) SA; the encoding respects all our criteria but the languages considered differ from the ones we have presented. Third, the results in [4] entail that Dπ cannot be encoded in π_a, under properties similar to ours; notably, they need homomorphism w.r.t. parallel composition whereas we just rely on compositionality. Fourth, [22,23] are inspired by Palamidessi's work on electoral systems [21] and separate several calculi for mobility according to the possibility of solving the problem of leader election. Though their approach is different from ours, our results confirm theirs: for example, they prove that π_a cannot encode MA, SA and BA. However, we want to remark that our approach is more informative than theirs, since we are able to compare pairs of languages in which leader election is either possible or impossible (e.g., SA and MA, or π_a and Dπ).

Finally, calculi for mobility have been a workbench for investigations on the expressiveness of operators like restriction, communication primitives, nondeterministic choice and replication ([3,18,21,20,12,8], just to cite some samples). These works are quite orthogonal to ours, since they compare different sub-calculi of the same language, whereas we compare different programming paradigms.

References

[1] G. Boudol. Asynchrony and the π-calculus (note). Rapport de Recherche 1702, INRIA Sophia-Antipolis, May 1992.

[2] M. Bugliesi, G. Castagna, and S. Crafa. Access Control for Mobile Agents: the Calculus of Boxed Ambients. *ACM Transactions on Programming Languages and Systems*, 26(1):57–124, 2004.

[3] N. Busi and G. Zavattaro. On the expressive power of movement and restriction in pure mobile ambients. *Theoretical Computer Science*, 322(3):477–515, 2004.

[4] M. Carbone and S. Maffeis. On the expressive power of polyadic synchronisation in π-calculus. *Nordic Journal of Computing*, 10(2):70–98, 2003.

[5] L. Cardelli, G. Ghelli, and A. D. Gordon. Mobility types for mobile ambients. In *Proc. of ICALP*, volume 1644 of *LNCS*, pages 230–239. Springer, 1999.

[6] L. Cardelli and A. D. Gordon. Types for mobile ambients. In *Proc. of POPL*, pages 79–92. ACM, 1999.

[7] L. Cardelli and A. D. Gordon. Mobile ambients. *Theoretical Computer Science*, 240(1):177–213, 2000.

[8] R. De Nicola, D. Gorla, and R. Pugliese. On the Expressive Power of KLAIM-based Calculi. *Theoretical Computer Science*, 356(3):387–421, 2006.

[9] R. De Nicola and M. Hennessy. Testing equivalence for processes. *Theoretical Computer Science*, 34:83–133, 1984.

[10] D. Gorla. Comparing calculi for mobility via their relative expressive power. Technical Report 09/2006, Dip. di Informatica, Università di Roma "La Sapienza".

[11] D. Gorla. On the relative expressive power of ambient-based calculi. To appear in the *Proc. of TGC*. Springer, 2008.

[12] D. Gorla. On the relative expressive power of asynchronous communication primitives. *Proc. of FoSSaCS*, volume 3921 of *LNCS*, pages 47–62. Springer, 2006.

[13] D. Gorla. Towards a Unified Approach to Encodability and Separation Results for Process Calculi. In *Proc. of CONCUR'08*, volume 5201, pages 492–507. Springer, 2008.

[14] M. Hennessy and J. Riely. Resource Access Control in Systems of Mobile Agents. *Information and Computation*, 173:82–120, 2002.

[15] K. Honda and M. Tokoro. An object calculus for asynchronous communication. In *Proc. of ECOOP*, volume 512 of *LNCS*, pages 133–147. Springer, 1991.

[16] F. Levi. A typed encoding of boxed into safe ambients. *Acta Informatica*, 42(6):429–500, 2006.

[17] F. Levi and D. Sangiorgi. Mobile safe ambients. *ACM Transactions on Programming Languages and Systems*, 25(1):1–69, 2003.

[18] S. Maffeis and I. Phillips. On the computational strength of pure ambient calculi. *Theoretical Computer Science*, 330(3):501–551, 2005.

[19] R. Milner. Functions as processes. *Journal of Mathematical Structures in Computer Science*, 2(2):119–141, 1992.

[20] U. Nestmann and B. C. Pierce. Decoding choice encodings. *Information and Computation*, 163:1–59, 2000.

[21] C. Palamidessi. Comparing the expressive power of the synchronous and the asynchronous π-calculi. *Mathematical Structures in Computer Science*, 13(5):685–719, 2003.

[22] I. Phillips and M. Vigliotti. Electoral systems in ambient calculi. In *Proc. of FoSSaCS*, volume 2987 of *LNCS*, pages 408–422. Springer, 2004.

[23] I. Phillips and M. Vigliotti. Leader election in rings of ambient processes. *Theoretical Computer Science*, 356(3):468–494, 2006.

[24] B. C. Pierce and D. N. Turner. Pict: A programming language based on the pi-calculus. In *Proof, Language and Interaction: Essays in Honour of Robin Milner*. MIT Press, 2000.

[25] J. Rathke, V. Sassone and P. Sobocinski. Semantic Barbs and Biorthogonality. In *Proc. of FoSSaCS*, volume 4423 of *LNCS*, pages 302–316. Springer, 2007.

[26] P. Zimmer. On the Expressiveness of Pure Safe Ambients. *Mathematical Structures in Computer Science*, 13:721–770, 2003.