To appear in the Proc. of *TGC'08*.

On the Relative Expressive Power of Ambient-based Calculi

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

Abstract. Nowadays, some of the most successful models for global computers are defined as ambient-based calculi; among them, the mainstream models are Mobile, Safe and Boxed Ambients. In this paper, we comparatively analyze most of their variants by comparing every variant against the language it comes from. In particular, we discuss and compare: objective moves in Mobile Ambients; passwords and a different semantics for the *out* primitive in Safe Ambients; different communication and mobility primitives in Boxed Ambients. By establishing the possibility/impossibility of encoding one language in another one, we discover which variants enhance the original language and which ones actually yield different formalisms.

1 Introduction

In the last few years, there has been a wide interest in the calculus of *Mobile Ambients* (MA) [4], an emerging model for global computing systems. As the name suggests, the key feature of MA is the notion of *ambient*, a bounded place where computations happen. Each ambient has a name, a collection of local processes and a collection of sub-ambients (each with its own name, processes and sub-ambients). The nesting of ambients forms a hierarchical (tree-like) structure that can be modified by the processes during a computation. A computation step (also called *reduction*) in MA may happen because an ambient, together with all its content, enters/exits another ambient, because an ambient is opened, so that its content is unleashed at its nesting level, or because co-located processes communicate.

The paradigm put forward by MA has stimulated the development of several ambient-based calculi, each modifying some feature of the original language. One of the first main developments is the calculus of *Safe Ambients* (SA) [11], obtained by adding *co-actions* to MA. The main difference is that ambient movements/openings in MA are initiated by only one ambient and the target ambient has no control over them; in contrast, in SA both participants must agree by using a matching action and co-action. A second major variation of MA is the calculus of *Boxed Ambients* (BA) [1], obtained by disallowing ambient openings and allowing a limited form of communication across boundaries (between parent and child in the tree-like nesting structure). After these two major variants, several minor variants of MA/SA/BA appeared in the literature.

A lot of efforts have been spent to define more and more sophisticated type theories to control and improve reliability of the global computing systems pro-

grammable in these models. However, a necessary (and more preliminary) requirement for establishing the trustworthiness of such systems is the deep understanding of all their distinctive language features. Moreover, a comparative analysis of the different proposals is desirable also for understanding the extent to which theories developed for one model can be transposed into another model.

In a companion paper [7], we approach these questions by comparing the language design issues of some calculi for mobility, including the three mainstream ambient-based calculi just mentioned; there, we prove that SA enhances the expressiveness of MA, whereas BA yields a language with an incomparable expressive power. In this paper, we go further and consider several variations of these three languages. We study the *relative expressive power* of each variation with respect to the calculus it comes from, i.e. we try to encode one language in the other, while respecting some reasonable properties. In principle, a good encoding should be "faithful", in that the encoding of a term must have the same functionalities as the original term. This notion can be formalized in different ways and a number of different proposals have been considered in the literature (e.g., sensitivity to barbs/divergence/deadlock, operational correspondence, full abstraction, ...).

Here, we take the proposal in [8] and consider only encodings that satisfy the following properties: *compositionality* (i.e., the encoding of a compound term must be expressed in terms of the encoding of its components), *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 sensitivity* (i.e., successful terms – for some notion of success – must be translated into successful terms, and vice versa). In [8] 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 set of criteria and the associated proof techniques developed in [8] are quickly recalled in Section 2. Then, we start our analysis in Section 3, where we present MA and compare it with two dialects that use objective moves. In particular, we show that MA is more expressive than MA_o, the *objective* variant of MA proposed in [4], whereas its expressive power is incomparable with the one of the *push and pull ambient calculus*, Pac [16].

In Section 4 we move to SA; we study how its expressiveness changes by introducing passwords and by changing the semantics for the *out* primitive, as suggested in [12]. We prove that passwords enhance the expressiveness of SA,

whereas the different modeling of the *out* primitive yields a language with an incomparable expressive power.

Finally, in Section 5 we consider BA and study the expressiveness of a few variations obtained by slightly changing its mobility and communication primitives. We first compare the expressiveness of *shared* and *localized* channels, and conclude that the two forms of communication are incomparable. Then, we extend BA with co-actions and passwords to obtain *safe boxed ambients* [13] and *new boxed ambients* [2]: we prove that both these variants enhance the expressiveness of BA and are incomparable.

Section 6 concludes the paper by mentioning some related work. Due to space limitations, no motivation on the languages considered is given; for this aspect and for a full explanation of their constructs, we refer the interested reader to their standard references. Moreover, we just quickly and intuitively sketch proofs; full details can be found in the on-line technical report [6].

2 The Formal Framework

2.1 Ambient-based Process Calculi

We assume a countable set of names, $\mathcal{N}$, ranged over by $m, n, \ldots, u, v, w, \ldots$, $x, y, z, \ldots$ and their decorated versions. To simplify reading, we shall use $m, n, \ldots$ to denote ambient names, $x, y, z, \ldots$ to denote input variables, and $u, v, w, \ldots$ to denote generic names.

A 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:

$$P \ ::= \ \mathbf{0} \ \mid M.P \ \mid n[P] \ \mid (\nu n)P \ \mid P_1|P_2 \ \mid !P \ \mid \surd$$

  As usual, $\mathbf{0}$ is the terminated process, whereas $\surd$ denotes success (see the discussion on Property 5 later on). In ambient-based calculi, $n[P]$ denotes process P running within ambient n; $M.P$ denotes process P prefixed by the sequence of actions M; $P_1|P_2$ denote 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*, defined by the following grammar:

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

  and $\mathcal{E}(P)$ denotes the process obtained by replacing the hole ' $\cdot$ ' with process P;
- $\equiv$ denotes *structural equivalence*, the least equivalence closed under alpha-renaming of bound names, under evaluation contexts and under the following axioms:

$$P \mid \mathbf{0} \equiv P \qquad P \mid Q \equiv Q \mid P \qquad P \mid (Q \mid R) \equiv (P \mid Q) \mid R$$

$$!P \equiv P \mid !P \qquad (\nu n)\mathbf{0} \equiv \mathbf{0} \qquad (\nu n)(\nu m)P \equiv (\nu m)(\nu n)P$$

$$P \,|\, (\nu n)Q \equiv (\nu n)(P \,|\, Q) \quad \text{if } n \notin \mathit{fn}(P)$$

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

  where $\mathit{fn}(P)$ denotes the *free names* (i.e., the names not bound) in P. Structural equivalence can be extended to evaluation contexts by getting rid of alpha-conversion.

Of course, the operational axioms are peculiar to every language, since they depend on the action prefixes. 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 with respect to closure under evaluation contexts (or, at the very least, with respect to parallel composition) and it is often defined in the form of a barbed equivalence [14], even though our results do not rely on any specific behavioural equivalence.

2.2 The Encodability Criteria

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 $\mathcal{L}_2$-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 $\mathit{fn}(S_1, \ldots, S_k) = N$.

Moreover, a good encoding should reflect in the encoded term all the name substitutions carried out in the source term.

Property 2. A translation $[\![\cdot]\!] : \mathcal{L}_1 \to \mathcal{L}_2$ is *name invariant* if, for every substitution σ, there exists a substitution σ' such that

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

In [8] we formally describe how the substitution in the target language (viz. σ') can be obtained from the substitution in the source (viz. σ). Notice that 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.

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

Another important semantic issue that an encoding should avoid is the introduction of infinite computations, written $\longmapsto^\omega$, when translating a terminating process.

Property 4. A translation $[\![\cdot]\!] : \mathcal{L}_1 \to \mathcal{L}_2$ is *divergence reflecting* whenever $[\![S]\!] \longmapsto^\omega$ implies that $S \longmapsto^\omega$, 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 [5, 19]. To formulate our property in a simpler way, we follow the approach in [19] 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, but all our results could be adapted to other modalities. 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 $[\![S]\!] \Downarrow$.

Definition 1 (Encoding). *An* encoding *of* $\mathcal{L}_1$ *into* $\mathcal{L}_2$ *is a translation* $[\![\cdot]\!] : \mathcal{L}_1 \to \mathcal{L}_2$ *that satisfies Properties 1–5.*

Finally, to prove a couple of results, we also need a further property, still very reasonable but not as basic as the previous five ones.

> *Property 6.* A translation $[\![\cdot]\!] : \mathcal{L}_1 \to \mathcal{L}_2$ is *adequate* if, for every S and S' such that $S \equiv S'$, it holds that $[\![S]\!] \simeq_2 [\![S']\!]$.

This property seems us quite acceptable, since the purpose of structural equivalence is relating different ways of writing the same process; thus, it is natural to require that the encodings of structurally equivalent processes behave in the same way. We could have asked for structural equivalence of the encoded terms, but, because of compositionality, this would have led to a too demanding property. It has to be said that Property 6 is quite close in spirit to the notion of *full abstraction*, whereas the proposal in [8] was defined as an alternative to such a notion. Thus, we would really like to avoid the use of Property 6; this leaves space for improving our results. Indeed, we believe that the impossibility results we are going to prove via Property 6 should also hold without it, but we have still not been able to prove them.

2.3 Derived Properties

In [8] we have shown that some separation results can be proved in the general framework we have just presented; however, to carry out more proofs, we have to slightly specialize the framework; this is mainly done by making some assumptions on the behavioural equivalence of the target language, viz. $\simeq_2$. In particular, in *loc.cit.* we have considered three alternative settings:

1. $\simeq_2$ is *exact*, i.e. $T \simeq_2 T'$ and T performs an action μ imply that T' (weakly) performs μ as well; moreover, parallel composition must be translated homomorphically, i.e. for every $N \subset \mathcal{N}$ it holds that $\mathcal{C}^N_{|}[\__1; \__2] = \__1 \mid \__2$;
2. $\simeq_2$ is *reduction sensitive*, i.e. $T \simeq_2 T'$ and $T' \longmapsto$ imply that $T \longmapsto$;
3. the occurrences of $\simeq_2$ in Property 3 are restricted to pairs of kind $(\mathcal{E}(T), T)$, for $\mathcal{E}(T) \simeq_2 T$.

All these assumptions are discussed and justified at length in [8]. By relying on them, we can prove a number of auxiliary results that will be useful in carrying out the main proofs of this paper.

Proposition 1. *Let* $[\![\cdot]\!]$ *be an encoding; then,* $S \mapsto\!\!\!\!\!\!/\;\;$ *implies that* $[\![S]\!] \mapsto\!\!\!\!\!\!/\;\;$.

Proposition 2. *Let* $[\![\cdot]\!]$ *be an encoding; 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* $[\![S_1 \mid S_2]\!] \longmapsto$.

Proposition 3. *Let* $[\![\cdot]\!] : \mathcal{L}_1 \to \mathcal{L}_2$ *be an encoding. If there exist two source terms* S_1 *and* S_2 *that do not reduce but such that* $[\![S_1 \mid S_2]\!] \longmapsto$, *then it can only be that* $\mathcal{C}_1([\![S_1]\!]) \mid \mathcal{C}_2([\![S_2]\!]) \longmapsto$, *where* $\mathcal{C}^{fn(S_1,S_2)}_{|}[\__1; \__2]$, *i.e. the context used to*

compositionally translate $S_1 \mid S_2$, is structurally equivalent to $\mathcal{E}(\mathcal{C}_1(\__1) \mid \mathcal{C}_2(\__2))$ 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., $m[\,\cdot\,]$, for some m).

Theorem 1. *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 $[\![\,\cdot\,]\!] : \mathcal{L}_1 \longrightarrow \mathcal{L}_2$.*

To state the following proof-technique, let us define the *matching degree* of a language $\mathcal{L}$, written $\textsc{Md}(\mathcal{L})$, as the least upper bound on the number of names that must be matched to yield a reduction in $\mathcal{L}$. For example, the matching degree of Mobile Ambients [4] is 1, whereas the matching degree of Safe Ambients with Passwords (SAP) [12] is 2.

Theorem 2. *If $\textsc{Md}(\mathcal{L}_1) > \textsc{Md}(\mathcal{L}_2)$, there exists no encoding $[\![\,\cdot\,]\!] : \mathcal{L}_1 \longrightarrow \mathcal{L}_2$.*

3 Mobile Ambients (MA) and its Variants

Definition 2 (MA processes and messages).

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

Intuitively, an ambient m enters into another ambient n via the in_n action, exits from another ambient n via the out_n action and is opened via the $open\_m$ action. Moreover, $\langle M \rangle$ represents message M ready to be consumed by a co-located input prefixed process $(x).P$ that, upon communication, replaces with M every occurrence of variable x in P. As usual, $(x).P$ binds x in P.

Definition 3 (MA reduction axioms).

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

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

MA, like all the following Ambient-based 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 [3, 11, 1].

3.1 Mobile Ambients with Objective Moves (MA_o)

The first variation of MA has been proposed in [4]: ambient movements, instead of being *subjective* (the moving ambient decides where and when moving), become *objective* (the moving ambient is stuck and moved from the outside). In

MA_o, actions in_n and out_n are replaced by $mv\, in\_n$ and $mv\, out\_n$, whose semantics is

$$mv\, in\_n.P_1 \mid n[P_2] \longmapsto n[P_1 \mid P_2] \qquad\qquad n[mv\, out\_n.P_1 \mid P_2] \longmapsto P_1 \mid n[P_2]$$

The following theorem proves that MA is strictly more expressive than MA_o.

Theorem 3. MA *is more expressive than* MA_o*: there exists an encoding of* MA_o *in* MA*; there exists no encoding of* MA *in* MA_o*.*

Proof. For the first part, it suffices to consider the translation of MA_o in MA provided in [4] and observe that it satisfies Properties 1–5. For the second part, we exploit Theorem 1 by noting that:

- $P \triangleq (\nu p)(open\_p.\surd \mid n[in\_n.p[out\_n.out\_n.\mathbf{0}]])$ is a MA process that does not reduce and does not report success but such that $P|P \Downarrow$;
- every MA_o-process T that does not reduce is such that $T \mid T \not\mapsto$. □

3.2 The Push and Pull Ambient Calculus (Pac)

The second variation of MA with objective moves is the so called *Push and Pull ambient calculus* (Pac) [16]. Now, actions in_n and out_n are replaced by $pull\_n$ and $push\_n$, whose semantics is

$$m[P_1] \mid n[pull\_m.P_2 \mid P_3] \longmapsto n[m[P_1] \mid P_2 \mid P_3]$$
$$n[m[P_1] \mid push\_m.P_2 \mid P_3] \longmapsto m[P_1] \mid n[P_2 \mid P_3]$$

Notice that the objective mobility in MA_o is much more controlled than that in Pac: at every moment, at most one movement for every ambient can happen in MA_o, since the moving ambient is blocked by the $mv\, in/mv\, out$ prefix. On the other hand, in Pac the same ambient can undergo different movements, because of execution of different parallel actions naming the same ambient. Indeed, MA can reasonably encode MA_o, whereas MA cannot encode Pac (nor vice versa), as the following theorem proves. We shall give full details for this proof, since the following ones will be similar or easier.

Theorem 4. MA *and* Pac *are incomparable: there exists no encoding satisfying Property 6 of* Pac *in* MA *and of* MA *in* Pac*.*

Proof. Let us work by contradiction and assume that such encodings do exist.

For the first claim, consider the Pac process $P \mid Q$, for $P \triangleq n[pull\_m.(\langle n \rangle \mid push\_p \mid p[\surd])]$, $Q \triangleq m[\mathbf{0}] \mid open\_p$ and $n \neq m$. By Proposition 2, its encoding must reduce; by Proposition 3 and by definition of MA reduction rules, this can happen in one of the following ways:

- $\mathcal{C}_1(\llbracket P \rrbracket)$ and $\mathcal{C}_2(\llbracket Q \rrbracket)$ communicate: if this were the case, then consider σ, the permutation that swaps n and m. By Property 2, we would have that $\llbracket P\sigma \mid Q \rrbracket \longmapsto$. This fact would falsify Proposition 1, thus implying that $\llbracket \cdot \rrbracket$ is not an encoding: contradiction.

– $\mathcal{C}_1(\llbracket P \rrbracket)$ contains an ambient that wants to enter into some ambient k and $\mathcal{C}_2(\llbracket Q \rrbracket)$ exhibits such an ambient at top-level:

If $\mathcal{C}_1(\cdot)$ was not empty, it must be that $\llbracket P \rrbracket$ contains a top-level in_k prefix, with $\llbracket P \rrbracket = \mathcal{C}^{\{n,m,p\}}_{n[\,]}(\llbracket pull\_n.(\langle m \rangle \mid push\_p \mid p[\surd]) \rrbracket)$ by compositionality. However, this is not possible, because otherwise either $\llbracket n[pull\_n.(\langle m \rangle \mid push\_p \mid p[\surd])] \mid Q \rrbracket \longmapsto$ or $\llbracket pull\_m.(\langle n \rangle \mid push\_p \mid p[\surd]) \mid Q \rrbracket \longmapsto$, according to whether $\mathcal{C}^{\{n,m\}}_{n[\,]}(\cdot)$ or $\llbracket pull\_m.(\langle n \rangle \mid push\_p \mid p[\surd]) \rrbracket$ contains the top-level in_k. Both these reductions would contradict Proposition 1.

So, it must be that $\mathcal{C}_1(\cdot)$ is empty and $\llbracket P \rrbracket$ contains the ambient that wants to enter into k; we now prove that this implies that either $\llbracket \cdot \rrbracket$ violates Proposition 1 or that $\llbracket !P \rrbracket$ can repeatedly provide an ambient that wants to enter into k (and so $\llbracket !P \mid Q \rrbracket$ diverges, in violation with Property 4). By Proposition 3, we know that $\mathcal{C}^{fn(P)}_{|}[\_1; \_2] \equiv \mathcal{E}(\_1 \mid \_2)$: indeed, also $\mathcal{C}_2(\cdot)$ must be empty, otherwise $\llbracket P \mid (\langle m \rangle \mid open\_m) \rrbracket \longmapsto$. Moreover, we can prove that the holes in $\mathcal{E}(\_1 \mid \_2)$ are not contained in any ambient, i.e. $\mathcal{E}(\cdot) \equiv (\nu \widetilde{n})(\ \_1 \mid \_2 \mid R)$.

If $k \notin \widetilde{n}$, then we can use Property 6 to state that $\llbracket !P \rrbracket \simeq \llbracket P \mid !P \rrbracket \equiv \mathcal{E}(\llbracket P \rrbracket \mid \llbracket !P \rrbracket) \equiv (\nu \widetilde{n})(\llbracket P \rrbracket \mid \llbracket !P \rrbracket \mid R)$; thus, $\llbracket !P \rrbracket$ contains a top-level ambient that wants to enter into k. But then $(\nu \widetilde{n})(\llbracket P \rrbracket \mid \llbracket !P \rrbracket \mid R) \simeq (\nu \widetilde{n})(\llbracket P \rrbracket \mid \llbracket P \mid !P \rrbracket \mid R) \equiv (\nu \widetilde{n})(\llbracket P \rrbracket \mid (\nu \widetilde{n})(\llbracket P \rrbracket \mid \llbracket !P \rrbracket \mid R) \mid R)$, and so on; hence, $\llbracket !P \rrbracket$ can exhibit as many top-level ambients that want to enter into k as desired.

We now prove that $k \in \widetilde{n}$ implies that there must exists another pair of complementary actions produced by $\llbracket P \rrbracket$ and $\llbracket Q \rrbracket$ such that either they fall in a different case of this Theorem (and, hence, $\llbracket \cdot \rrbracket$ would violate Proposition 1) or one of them is produced by an ambient that wants to enter into some ambient $h \notin \widetilde{n}$ (and we can then conclude by the previous reasoning). It it was not the case, then $\llbracket P \mid ((\nu b) in\_b.P \mid Q) \rrbracket$, that is structurally equivalent to $\mathcal{E}(\llbracket P \rrbracket \mid \mathcal{E}(\llbracket ((\nu b) in\_b.P \mid Q) \rrbracket))$, would not reduce, in contradiction with Proposition 2.

– $\mathcal{C}_1(\llbracket P \rrbracket)$ wants to open an ambient k and $\mathcal{C}_2(\llbracket Q \rrbracket)$ exhibits such an ambient at top-level: similarly, we can prove that $\llbracket \cdot \rrbracket$ is not an encoding by showing that either $\llbracket pull\_m.(\langle n \rangle \mid push\_p \mid p[\surd]) \mid Q \rrbracket \longmapsto$ or $\llbracket n[pull\_n.(\langle m \rangle \mid push\_p \mid p[\surd])] \mid Q \rrbracket \longmapsto$, against Proposition 1.

– $\mathcal{C}_1(\llbracket P \rrbracket)$ exhibits a top-level k ambient that $\mathcal{C}_2(\llbracket Q \rrbracket)$ wants to open/enter: similar to the previous case.

The second claim can be proved in a very similar way, by letting $P \triangleq n[in\_m.(\langle n \rangle \mid p[out\_n.out\_m.\surd])]$. Just notice that now $\llbracket !P \mid Q \rrbracket \longmapsto^{\omega}$ in the second item of the previous proof is replaced by $\llbracket n[!\, in\_m.(\langle n \rangle \mid p[out\_n.out\_m.\surd])] \mid !Q \rrbracket \longmapsto^{\omega}$ (and, again, this can be obtained thanks to Property 6). □

4 Safe Ambients (SA) and its Variants

The Safe Ambient calculus [11] extends MA by adding *co-actions* to make the execution of actions *in*/*out*/*open* more controlled.

Definition 4 (SA processes and messages). SA *extends Definition 2 with*

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

Definition 5 (SA reduction axioms).

$$(x).P \mid \langle M \rangle \longmapsto P\{^{M}/_{x}\}$$
$$open\_m.P_1 \mid m[\overline{open}\_m.P_2 | P_3] \longmapsto P_1 \mid P_2 \mid P_3$$
$$m[in\_n.P_1 | P_2] \mid n[\overline{in}\_n.P_3 | P_4] \longmapsto n[P_3 \mid P_4 \mid m[P_1 | P_2]]$$
$$n[m[out\_n.P_1 | P_2] \mid \overline{out}\_n.P_3 \mid P_4] \longmapsto m[P_1 | P_2] \mid n[P_3 | P_4]$$

4.1 Passwords and alternative modeling of the *out* action

In [12], SA has been enriched with passwords: an ambient n that aims at entering/exiting/opening another ambient m must not only be authorized by m via a corresponding co-action, but it must also exhibit some credential to perform the action (credentials are simply names and are called *passwords*). Intuitively, passwords are a way to better control ambient movements and openings: for example, in SA any ambient can open an ambient m that performs a $\overline{open}\_m$ action; with passwords, the action becomes $\overline{open}\_(m, p)$ and only the ambients knowing the password p can open m.

Let SA_p be the language where processes are defined like in Definition 2, whereas messages are defined as follows:

$$M ::= u \mid in\_(u, v) \mid out\_(u, v) \mid open\_(u, v) \mid \overline{in}\_(u, v) \mid \overline{out}\_(u, v) \mid \overline{open}\_(u, v) \mid M.M$$

The reductions rules extend the axioms in Definition 5 by also matching passwords.

Theorem 5. SA_p *is more expressive than* SA*: there exists an encoding of* SA *in* SA_p*; there exists no encoding of* SA_p *in* SA*.*

Proof. SA is trivially encodable in SA_p by letting each action and co-action naming n use n also as password. The converse cannot hold because in SA_p every reduction requires to atomically match two names (the name of the ambient target of the action and the password); SA reductions, instead, can match at most one name. This suffices to conclude, because of Theorem 2. □

The language proposed in [12] (called SAP) differs from SA_p in the semantics of the *out* action: in SAP, the co-action is not in the ambient left (like in SA and

SA_p) but is in the receiving ambient. Formally, the axiom to exit an ambient now becomes:

$$n[m[out\_(n,p).P_1|P_2] \mid P_3] \mid \overline{out}\_(n,p).P_4 \longmapsto m[P_1|P_2] \mid n[P_3] \mid P_4$$

We now prove that this slight modification makes SAP incomparable with both SA and SA_p; thanks to Theorem 5, it suffices to prove the following two results.

Theorem 6. *There exists no encoding of* SAP *in* SA_p *(and hence in* SA*).*

Proof. Consider the processes $P_1^1 \triangleq m[n[out\_(m,p)]]$, $P_2^1 \triangleq \overline{out}\_(m,p).\surd$, $P_1^2 \triangleq n[in\_(m,p)]$, $P_2^2 \triangleq m[\overline{in}\_(m,p).q[out\_m.\overline{open}\_q]] \mid \overline{out}\_m.open\_q.\surd$, $P_1^3 \triangleq m[\overline{open}\_(m,p).\langle n\rangle]$ and $P_2^3 \triangleq open\_(m,p).\surd$. The fact that $[\![P_1^i \mid P_2^i]\!]$ must evolve, for every $i = 1,2,3$, leads us to conclude that at least one of the following encodings must reduce: $[\![P_1^i\sigma \mid P_2^i]\!]$, for $i \in \{1,2,3\}$ and some non-trivial permutation σ of $\{n,m,p\}$ (that we assume pairwise distinct), $[\![P_1^1 \mid P_1^2]\!]$, $[\![P_1^1 \mid n[in\_(p,m)]]\!]$, $[\![P_1^1 \mid P_2^3]\!]$, $[\![P_1^1 \mid open\_(p,m).\surd]\!]$, $[\![P_1^2 \mid P_2^3]\!]$ or $[\![P_1^2 \mid open\_(p,m).\surd]\!]$. A reduction of any of these encodings would contradict Proposition 1. □

Theorem 7. *There exists no encoding of* SA *(and hence of* SA_p*) in* SAP*.*

Proof. Consider $m[n[out\_m.\overline{open}\_n] \mid \overline{out}\_m] \mid open\_n.\surd$, for $n \neq m$; because of Proposition 2, it encoding must reduce. The proof consists in showing that every possible kind of reduction implies that at least one of the following encodings reduce (in contradiction with Proposition 1): $[\![m[out\_m.\overline{open}\_n \mid \overline{out}\_m] \mid open\_n.\surd]\!]$, $[\![m[\mathbf{0}] \mid open\_n.\surd]\!]$ or $[\![m[n[\mathbf{0}] \mid \overline{out}\_m] \mid open\_n.\surd]\!]$. □

5 Boxed Ambients (BA) and its Variants

The Boxed Ambient calculus [1] evolves MA by removing the *open* action (that is considered too powerful) and by allowing 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 $\hat{}$) or towards an enclosed ambient n (if tagged with direction n).

Definition 6 (BA processes, messages and directions).

$$P ::= \ldots \mid (x)^\eta.P \mid \langle M\rangle^\eta.P$$

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

Definition 7 (BA reduction axioms).

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

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

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

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

5.1 Shared vs Localized Channels in BA: BA_s and BA

The operational rules for parent-child communications given in Definition 7 exploits *localized channels*: the communication channel is owned either by the parent (4th and 6th rule) or by the child (5th and 7th rule). Another possible way to model parent-child communications in BA exploits *shared channels*: the communication channel is shared by the parent and its child. Formally, BA_s is the calculus derived from BA by letting the last four rules of Definition 7 be replaced by:

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

BA_s provides a more controlled form of communication, since it rules out the interferences that can arise, e.g., in $(x)^n \mid n[\langle M\rangle^\star \mid (y)^\star \mid m[(z)\hat{}]]$, where message M can be consumed by three different input actions placed in different ambients. However, as we now prove, the two forms of communication are incomparable.

Theorem 8. *BA_s and BA are incomparable: there exists no encoding of BA_s in BA; there exists no converse encoding that also satisfies Property 6.*

Proof. For the second claim, we notice that BA has more kinds of remote reductions than BA_s; the main idea underlying the proof is to show that this higher flexibility in the source language cannot be reflected in the target language. In particular, consider $P_1 \triangleq (x)^n.(b[\mathbf{0}] \mid \surd) \mid n[\langle b\rangle^\star]$, $P_2 \triangleq \langle b\rangle^n.\surd \mid n[(x)^\star.b[\mathbf{0}]]$, $P_3 \triangleq (x)^\star.(n[b[\mathbf{0}]] \mid \surd) \mid n[\langle b\rangle\hat{}]$ and $P_4 \triangleq \langle b\rangle^\star.(n[\mathbf{0}] \mid \surd) \mid n[(x)\hat{}.b[\mathbf{0}]]$; by Proposition 2, their encodings must evolve. Again, this can be used to show that the encoding either violates Proposition 1 or introduces divergence (and to this aim Property 6 is needed).

For the first claim, the problem is that it is impossible in BA to rule out the interferences excluded by the tighter control on communications of BA_s. The proof proceeds like for the previous claim (but now Property 6 is not needed anymore), by considering processes $P_1 \triangleq (x)^n.(b[\mathbf{0}] \mid \surd) \mid n[\langle b\rangle\hat{}]$, $P_2 \triangleq \langle b\rangle^n.\surd \mid n[(x)\hat{}.b[\mathbf{0}]]$ and $P_3 \triangleq (x)^\star.(n[b[\mathbf{0}]] \mid \surd) \mid \langle b\rangle^\star.n[\mathbf{0}]$. □

5.2 Alternative Mobility Primitives in BA: SBA and NBA

We first consider *Safe Boxed Ambient* (SBA) [13]: it is BA extended with co-actions to better control ambient movements, in the same spirit as SA. However, in SBA co-actions can either allow any ambient enter/exit a given ambient n

(and this is similar to SA), or can selectively allow movements (this resembles SAP, though no password appears in SBA). Formally, the reductions for ambient movements are:

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

for $\delta \in \{*, n\}$. Also notice that the $\overline{out}$ is placed outside the ambient left, like in SAP. We now prove that SBA enhances the expressiveness of BA.

Theorem 9. SBA *is more expressive than* BA*: there exists an encoding of* BA *in* SBA*; there is no encoding of* SBA *in* BA*.*

Proof. It is easy to prove that SBA can encode BA: it suffices to translate every operator homomorphically, except for $[\![\,\mathbf{0}\,]\!] \triangleq\ !\,\overline{out}\_*$ and $[\![\,u[P]\,]\!] \triangleq\ !\,\overline{out}\_* \mid u[!\,\overline{in}\_* \mid [\![\,P\,]\!]]$. For the converse, notice that $\text{MD}(\text{SBA}) = 2 > \text{MD}(\text{BA}) = 1$ and apply Theorem 2. □

Another variation on BA is *New Boxed Ambients* (NBA) [2]: it adopts the shared-channel form of communication of BA_s, it introduces passwords in mobility actions (similarly to SAP) and let co-actions dynamically learn the name of the ambient that performed the corresponding action. As we have shown in Theorem 8, localized channels cannot be reasonably encoded in shared channels nor vice versa; thus, to compare NBA with BA and SBA, we define the variant of NBA with localized channels that we call NBA_l. Formally, its distinctive reduction rules are:

$$n[in\_(m,p).P_1 \mid P_2] \mid m[\overline{in}\_(x,p).P_3 \mid P_4] \longmapsto m[n[P_1 \mid P_2] \mid P_3\{^n/_x\} \mid P_4]$$
$$m[n[out\_(m,p).P_1 \mid P_2] \mid P_3] \mid \overline{out}\_(x,p).P_4 \longmapsto n[P_1 \mid P_2] \mid m[P_3] \mid P_4\{^n/_x\}$$

Theorem 10. NBA_l *is more expressive than* BA*: there exists an encoding of* BA *in* NBA_l*; there is no encoding of* NBA_l *in* BA*.*

Proof. NBA_l can encode BA: it suffices to translate every operator homomorphically, except for

$$[\![\,\mathbf{0}\,]\!] \triangleq\ !\,\overline{out}\_(x,p) \qquad [\![\,u[P]\,]\!] \triangleq\ !\,\overline{out}\_(x,p)\ \mid u[!\,\overline{in}\_(x,p)\ \mid [\![\,P\,]\!]]$$
$$[\![\,in\_u.P\,]\!] \triangleq in\_(u,p).[\![\,P\,]\!] \qquad [\![\,out\_u.P\,]\!] \triangleq out\_(u,p).[\![\,P\,]\!]$$

for some predefined and fixed (constant) password p. For the converse, notice that $\text{MD}(\text{NBA}_l) = 2 > \text{MD}(\text{BA}) = 1$ and apply Theorem 2. □

We have shown that both SBA and NBA_l are more expressive than BA; it remains to understand the relationships between NBA_l and SBA. We now prove that the two languages are incomparable.

Theorem 11. NBA_l *and* SBA *are incomparable: there exists no encoding of* NBA_l *in* SBA*, nor vice versa.*

Proof. For the first claim, consider processes $P \triangleq n[in\_(m,p).\langle q\rangle^\star]$ and $Q \triangleq m[\overline{in}\_(x,p).\langle\,\rangle^\star] \mid (\,)^m.\surd$, for n, m, p and q pairwise distinct. For the second claim, consider processes $P \triangleq n[in\_m]$ and $Q \triangleq m[\overline{in}\_n.\langle\,\rangle^\star] \mid (\,)^m.\surd$, for $n \neq m$. In both cases, we can prove like before that $[\![\, P \mid Q \,]\!] \longmapsto$, that holds because of Proposition 2, leads to contradict Proposition 1 for some minor variations of P and /or Q. □

6 Conclusions and Related Work

We have studied three ambient-based calculi and some of their variants, namely Mobile Ambients (compared with two dialects with objective moves), Safe Ambients (compared with its dialect with passwords) and Boxed Ambients (compared with the dialects resulting from some variations of its communication and mobility primitives). To this aim, we have exploited the set of criteria presented and discussed in [8]. However, we believe that all our impossibility results should hold under different 'reasonable' sets of properties.

Our results carry a two-fold contribution: on one hand, they should help in better clarifying the peculiarities of the languages studied and their distinctive programming features; on the other hand, they allow us to formally compare the expressive power of the languages and their inter-relationships. In some cases, we have discovered that the dialect proposed is comparable, in terms of expressive power, with the language it comes from: for example, MA_o reduces the expressiveness of MA, whereas SBA and NBA enhance the expressiveness of BA. In other cases, we have discovered that the dialect and its original language are incomparable, i.e. no relative encoding exists: the most remarkable cases are Pac vs MA, SAP vs SA and BA_s vs BA. In these cases, we must be aware that the dialect is not a different presentation of the original language nor a minor variation on it, as it is sometimes believed. Indeed, the distinguishing features added to (or modified in) the original language can have advantages (e.g., in terms of ease-of-programming or of controlling interferences) that make the dialect non-encodable in the original language; the price to be paid is that some computational feature of the original language gets lost, thus making also the converse encoding impossible.

To conclude, we want to mention a few strictly related results. First, [10] provides an encoding of BA_s in a variant of SA that exploits mobility primitives similar to those in SBA. The encoding respects all our criteria but the target language is still another variant of the languages we have presented. Second, [17, 18] are inspired by Palamidessi's work on electoral systems [15], where separation results are formulated according to the possibility/impossibility of electing a leader in a symmetric system (i.e. a set of parallel processes programmed in the same way, modulo renamings). Though their approach is different from ours, our results confirm theirs. For example, [17] proves that there is no encoding of MA into MA_o, as we have also shown. However, our approach is more discriminating: for example, we have proved that Pac and MA are incomparable, whereas leader eligibility does not say much on the relative expressive power of

such languages, since leader election is possible in both of them. Actually, by using leader election, we can only say that MA_o is less expressive than all the other languages presented in this paper, because it is the only one where leader election is impossible. This is an evidence of the fact that our comparison is more structured and informative.

References

1. M. Bugliesi, G. Castagna, and S. Crafa. Access control for mobile agents: the calculus of Boxed Ambients. *Trans. on Progr. Lang. and Syst.*, 26(1):57–124, 2004.
2. M. Bugliesi, S. Crafa, M. Merro, and V. Sassone. Communication and mobility control in Boxed Ambients. *Information and Computation*, 202(1):39–86, 2005.
3. L. Cardelli, G. Ghelli and A. D. Gordon. Types for the Ambient Calculus. *Information and Computation*, 177(2):160–194, 2002.
4. L. Cardelli and A. D. Gordon. Mobile ambients. *Theoretical Computer Science*, 240(1):177–213, 2000.
5. R. De Nicola and M. Hennessy. Testing equivalence for processes. *Theoretical Computer Science*, 34:83–133, 1984.
6. D. Gorla. Comparing calculi for mobility via their relative expressive power. Tech. Rep. 09/2006, Dip. di Informatica, Università di Roma "La Sapienza".
7. D. Gorla. On the relative expressive power of calculi for mobility. Extended abstract of the first part of [6], unpublished.
8. D. Gorla. Towards a unified approach to encodability and separation results for process calculi. Proc. of *CONCUR*, volume 5201 of *LNCS*, pages 492–507, 2008.
9. N. Kobayashi. A partially deadlock-free typed process calculus. *ACM Transactions on Programming Languages and Systems*, 20(2):436–482, 1998.
10. F. Levi. A typed encoding of boxed into safe ambients. *Acta Informatica*, 42(6):429–500, 2006.
11. F. Levi and D. Sangiorgi. Mobile safe ambients. *ACM Transactions on Programming Languages and Systems*, 25(1):1–69, 2003.
12. M. Merro and M. Hennessy. A bisimulation-based semantic theory of Safe Ambients. *ACM Trans. on Programming Languages and Systems*, 28(2):290–330, 2006.
13. M. Merro and V. Sassone. Typing and subtyping mobility in boxed ambients. In *Proc. of CONCUR'02*, volume 2421 of *LNCS*, pages 304–320. Springer, 2002.
14. R. Milner and D. Sangiorgi. Barbed bisimulation. In *Proc. of ICALP '92*, volume 623 of *LNCS*, pages 685–695. Springer, 1992.
15. C. Palamidessi. Comparing the expressive power of the synchronous and the asynchronous π-calculi. *Mathem. Structures in Computer Science*, 13(5):685–719, 2003.
16. I. Phillips and M. Vigliotti. On reduction semantics for the push and pull ambient calculus. *IFIP Conf. on Theoretical Comp. Sci.*, pages 550-562. Kluwer, 2002.
17. I. Phillips and M. Vigliotti. Electoral systems in ambient calculi. In *Proc. of FoSSaCS*, volume 2987 of *LNCS*, pages 408–422. Springer, 2004.
18. I. Phillips and M. Vigliotti. Leader election in rings of ambient processes. *Theoretical Computer Science*, 356(3):468–494, 2006.
19. J. Rathke, V. Sassone and P. Sobocinski. Semantic barbs and biorthogonality. In *Proc. of FoSSaCS*, volume 4423 of *LNCS*, pages 302–316. Springer, 2007.