

Communication and Mobility Control in Boxed Ambients*

Michele Bugliesi
Università “Ca’ Foscari”, Venezia

Silvia Crafa
Università “Ca’ Foscari”, Venezia

Massimo Merro
Università di Verona

Vladimiro Sassone
University of Sussex

Abstract

Boxed Ambients (BA) replace Mobile Ambients’ open capability with communication primitives acting across ambient boundaries. The expressiveness of the new model of communication is achieved at the price of interferences that affect message reception and whose resolution requires synchronisation of activities at multiple, distributed locations. We study a variant of BA aimed at controlling communication interferences as well as mobility ones. Our calculus modifies the communication mechanism of BA, and introduces a new form of co-capability, inspired from Safe Ambients (SA) (with passwords), that registers incoming agents with the receiver ambient while at the same time performing access control. We prove that new calculus has a rich semantics theory, including a sound and complete coinductive characterisation, and an expressive, yet simple type system. Through a set of examples, and an encoding, we characterise its expressiveness with respect to both BA and SA.

Introduction

The calculus of Mobile Ambients [5] (MA) introduced the notion of ambient acting at the same time as administrative domain and computational environment. Processes live inside ambients, and inside ambients compute and interact. Ambients relocate themselves, carrying along all their contents: their migration, triggered by the processes they enclose, models mobility of entire domains and active computational loci. Two capabilities control ambient

*Work supported by ‘MyThS: Models and Types for Security in Mobile Distributed Systems’, EU FET-GC IST-2001-32617; ‘Mikado: Mobile Calculi based on Domains’ EU FET-GC IST-2001-32222; and by MIUR Project ‘Mefisto: Modelli Formali per la Sicurezza’.

movements: in and out. These are performed by processes wishing their enclosing ambient to move to a sibling and, respectively, out of its parent. The corresponding reductions are shown below, where P , Q and R are processes, m and n ambient names, $|$ is parallel composition, and square brackets delimit ambients' contents:

$$n[\text{in } m.P \mid Q] \mid m[R] \longrightarrow m[n[P \mid Q] \mid R], \quad m[n[\text{out } m.P \mid Q] \mid R] \longrightarrow n[P \mid Q] \mid m[R].$$

A third capability, open, can be used to dissolve ambients, as expressed by the reduction $\text{open } n.P \mid n[Q] \longrightarrow P \mid Q$. Process interaction is by anonymous message exchanges confined inside ambients, as in

$$n[\langle M \rangle.P \mid (x).Q] \longrightarrow n[P \mid Q\{x := M\}],$$

where brackets represent outputs, curly brackets substitutions, and round parentheses bind input variables.

These ideas have given rise to an innovative calculus capturing several aspects of current real-world distributed systems, and have posed some new hard problems. Paper [11] unveiled a set of so-called grave interferences, i.e. situations where the inherent nondeterminism of movement goes wild. For instance, in

$$k[n[\text{in } m.P \mid \text{out } k.R] \mid m[Q]]$$

although n 's next hop is beyond n 's control, the difference that such a choice brings about is so big that it is difficult to see how such a situation could have been purposely programmed. Levi and Sangiorgi's proposal of Safe Ambients (SA) in [11] counters the problem by using 'co-actions' to grant ambients a form of control over other ambients' access. A process willing to be entered will manifest that explicitly, as e.g. in

$$n[\text{in } m.P \mid Q] \mid m[\overline{\text{in}} m.R \mid S] \longrightarrow m[n[P \mid Q] \mid R \mid S],$$

and similarly for out and open. Building on such infrastructure, a type-system enforced notion of *single-threadedness* ensures that at any time ambients are willing to engage in at most one activity (across boundaries) that may lead to grave interferences.

Recently, Merro and Hennessy [12] found it useful to work with a version of SA called SAP, where incoming ambients must be able to present a suitable password in order to cross ambients' boundaries. Paper [12] develops a treatable semantic theory for SAP in the form of a labelled transition system (LTS) based characterisation of its (reduction) barbed congruence. We will find use for some of these ideas in the present paper too.

Another source of potential problems is open in its own nature as ambient dissolver. A process exercising such a capability will embody all the contents of the dissolved ambient, including its capabilities and migration strategies. Of course there is nothing inherently wrong with that and indeed it is from open that MA gain part of their expressiveness in systems with dynamic topology. However, despite its usefulness, from the system designer's point of view open must be handled with the greatest care.

The calculus of Boxed Ambients [3] (BA) was born out of the observation that, after all, there is an alternative way to yield expressiveness: namely, by direct communication

across boundaries, as in the Seal calculus [21]. As shown below, in BA it is possible to draw an input from a sub-ambient n 's local channel (viz. $(x)^n$) as well as from the parent's local channel (viz. $(x)^\dagger$), and dually with the roles of input and output swapped.

$$\begin{aligned} (x)^n.P \mid n[\langle M \rangle.Q \mid R] &\longrightarrow P\{x := M\} \mid n[Q \mid R] \\ \langle M \rangle.P \mid n[(x)^\dagger.Q \mid R] &\longrightarrow P \mid n[Q\{x := M\} \mid R]. \end{aligned}$$

Although remarkable in many respects (cf. [3]), such design choices, have the drawback of introducing a great amount of non-local nondeterminism and communication interference. This is exemplified perfectly by the term below, where a single message issued in n unleashes a nondeterministic race among three potential receivers located in three different ambients:

$$m[(x)^n.P \mid n[\langle M \rangle \mid (x).Q \mid k[(x)^\dagger.R]]]$$

This raises difficulties for a distributed implementation of BA, as there is a hidden, non trivial distributed consensus problem to address at each communication. These forms of interference are as grave as those that led to the definition of SA, and they should be regarded as programming errors too.

In this paper we propose a variant of BA aimed at controlling such interferences and at providing a fresh foundation for the ideas behind BA. Our proposal, NBA, takes inspiration from [9], and is based on the idea that each ambient comes equipped with two mutually non-interfering channels, respectively for local and upward communications.

$$\begin{aligned} (x)^n.P \mid n[\langle M \rangle^\wedge.Q \mid R] &\longrightarrow P\{x := M\} \mid n[Q \mid R] \\ \langle M \rangle^n.P \mid n[(x)^\wedge.Q \mid R] &\longrightarrow P \mid n[Q\{x := M\} \mid R] \end{aligned}$$

Hierarchical communication, whose new rules are shown above, is indicated by a pair of distinct constructors, simultaneously on input and output, so that no communication interference is possible. The upward channel can be thought of as a gateway between parent and child, located at the child's and travelling with it, and poses no particular implementation challenges.

From the theoretical viewpoint, a first consequence of the elimination of unwanted interferences is a set of good, expected algebraic laws for NBA, as illustrated in §4. Also, the type system for BA results considerably simplified. In particular, the types of ambients and capabilities need only record upward exchanges, while processes are characterised by their local and hierarchical exchanges. The details are discussed in §5.

Unfortunately, limiting ourselves to banning communication interferences as above would result in a poorly expressive calculus (although some of its good properties have been underlined in [9]). For instance, in the system $n[P]$ there would be no way for P to communicate with its sub-ambients, unless their names were statically known. In our effort to tackle interference we seem to have killed hierarchical communication at all. Far from that, in order to regain expressive power we only need to reinstate a mechanism for an ambient to learn dynamically the names of incoming ambients. Essentially, our idea is to introduce co-actions of the form $\overline{m}(x)$ that have the effect of binding such names to the

variable x . Similarly to SA, co-actions provide a mechanism for expressing a general willingness to accept incoming ambients; in addition to that, the receiving ambient learns the incoming ambient's name. It can thus be thought as (an abstraction of) an access protocol as it would actually take place in real computational domains, where newly arrived agents would have to register themselves in order to be granted access to local resources.

Observe, however, that a purely binding mechanism such as this would not in itself be able to control access, but only to register it. In order to provide ambients with a finer mechanism of access control, we add a second component to our (co-)capabilities and write rules as the one below.

$$a[\text{in}\langle b, k \rangle . P_1 \mid P_2] \mid b[\overline{\text{in}}(x, k) . Q_1 \mid Q_2] \longrightarrow b[a[P_1 \mid P_2] \mid Q_1\{x := a\} \mid Q_2]$$

In practical terms, this enhances our access protocol with a form of control over the credentials of incoming processes (k in the rule above), as a preliminary step to the registration protocol. An example for all of the practical relevance and naturality of this mechanism, is the negotiation of credential that takes place when connecting to a wireless LAN or to a LAN using DHCP or to a ISP using PPP.

Remarkably, our admission mechanism resembles quite closely the notion of passwords as developed in [12], which thus arises yet again as a natural notion to consider. As a consequence, we benefit from results similar to those in *loc. cit.* In particular, we devise a labelled transition semantics for NBA that yields a bisimulation congruence sound with respect to (reduction) barbed congruence, and we use it to prove a number of laws. Passwords also have a relevant role in the type system, where their types keep track of the type of (upward exchanges of) incoming ambients, so contributing effectively to a clean and easy type system.

As the paper will show, besides having practical, implementation-oriented features and enjoying good theoretical properties, such as a rich and tractable algebraic theory and a simple type system, at the same time NBA remains expressive enough. In particular, by means of examples and encodings in §7 we show that the expressive power we loose with respect to BA is, as expected and planned, essentially that directly related to communication interferences.

Structure of the paper. §1 introduces the calculus, presents the reduction semantics and the associated notion of behavioural equivalence. §2 and §3 develop an alternative semantics based on an LTS, that yields a bisimilarity that is proved to be sound with respect to the reduction barbed congruence of §1. In §4 we use this relation to prove a number of algebraic laws for the calculus. The type system of NBA is illustrated and discussed in §5. §6, §7 and §8 focus on expressiveness issues in relation to BA and SA, including several examples, an encoding of the π calculus, and an encoding of BA into (an extension of) NBA. §9 shows an alternative LTS, whose associated bisimilarity fully characterises barbed congruence at the price of introducing additional higher order labels. Finally, §10 is dedicated to conclusions.

A preliminary version of this paper appeared in [4].

1 The NBA Calculus

The syntax includes two syntactic categories, *messages* and *processes*, summarised in Table 1. Messages (or expressions) are ranged over by M, N and include *names*, *variables* and *capabilities*. We presuppose two mutually disjoint sets: \mathbf{N} of names, and \mathbf{V} of variables. The set \mathbf{V} is ranged over by letters toward the end of the alphabet, typically x, y, z , while the remaining letters $a, b, \dots, m, n, \dots, q, r$ are reserved for the names in the set \mathbf{N} .

Processes, ranged over by P, Q, R, S , are built from the constructors of *inactivity*, *parallel composition*, *replication* and *restriction*, *prefix*, anonymous (polyadic) *input*, *output*, and *ambient*. The syntactic structure is similar to that of the original calculus BA [3]. The main differences are in the constructs for mobility: the movement capabilities now have two arguments – the name of the target ambient, and the password to be provided along with the name – and they are matched by co-actions $\overline{\text{in}}(x, N)$ and $\overline{\text{out}}(x, N)$ built around a variable x and an expression (typically, a name) N . Also, the calculus has replicated prefixing, rather than full replication: this will result in an image-finite labelled transition system.

The input operator $(\tilde{x} : \tilde{W}).P$ is a binder for the *variables* \tilde{x} , and so are the two co-actions $\overline{\text{in}}(x, M).P$ and $\overline{\text{out}}(x, M).P$, whereas the restriction operator $(\nu n : W)P$ binds the *name* n : in all cases the scope of the binder is P . As it is customary, terms that are α -convertible are considered identical. The notions of *free names* and *free variables* of a process, noted $\text{fn}(P)$ and $\text{fv}(P)$ respectively, arise as expected, and so does the definition of *capture free* substitution $P\{\tilde{x} := \tilde{M}\}$. We sometime use the notation $\text{fn}(P, Q)$ as a shorthand for $\text{fn}(P) \cup \text{fn}(Q)$, and similarly $\text{fv}(P, Q)$. A name (variable) is *fresh* in a term if it is different from any other free name (variable) in that term. A process is *closed* if has no free variables (though it may have free names). We use a number of notational conventions. Parallel composition has the lowest precedence among the operators. The process $M.N.P$ is read as $M.(N.P)$. We write $\langle \tilde{M} \rangle^\eta$, and (\tilde{x}) for $\langle M_1, \dots, M_k \rangle^\eta$ and (x_1, \dots, x_k) respectively. Similarly, we write $(\nu \tilde{n})$ for $(\nu n_1) \dots (\nu n_k)$, and define term equality up to rearrangements of adjacent restrictions. We omit trailing dead processes, writing M for $M.0$, $\langle \tilde{M} \rangle$ for $\langle \tilde{M} \rangle.0$, and $n[]$ for $n[0]$.

1.1 Reduction and Behavioural Semantics

The dynamics of the calculus is defined in Table 1 and, as usual, is up to structural congruence. The definition of structural congruence, noted \equiv , is standard (cf. [5]).

The *mobility rules* require as in [12] that the ambients involved in the move to agree on some password k ; in addition the target of the move gets to know the name of the moving ambient as a result of synchronisation. Also differently from *loc. cit.*, the co-out action in rule (EXIT) does not mention the name of moving ambient, and so it provides for lesser control over ambient movement.

The *communication rules* are explained and motivated in the introduction. As usual, in all communication rules we assume that tuples have the same arity, a condition that will be enforced by the type system.

As to behavioural equivalence, we rely on *reduction barbed congruence* [10], defined in terms of reduction and observability, which appears appropriate to capture the dynamics

<i>Locations:</i>			<i>Messages:</i>		
η	$::=$	a	M, N	$::=$	a
		\hat{a}			$\text{in}\langle M, N \rangle$
		\star			$\text{out}\langle M, N \rangle$
					$M.N$
					name
					enter
					exit
					path
<i>Processes:</i>			<i>Prefixes:</i>		
P	$::=$	$\mathbf{0}$	π	$::=$	M
		$P_1 P_2$			$(x_1, \dots, x_k)^\eta$
		$(\nu n)P$			$\langle M_1, \dots, M_k \rangle^\eta$
		$! \pi.P$			$\overline{\text{in}}(x, M)$
		$M[P]$			$\overline{\text{out}}(x, M)$
		$\pi.P$			
		nil process			capability
		composition			input
		restriction			output
		replication			allow enter
		ambient			allow exit
		prefixing			

mobility

(ENTER) $n[\text{in}\langle m, k \rangle.P_1 | P_2] \mid m[\overline{\text{in}}(x, k).Q_1 | Q_2] \longrightarrow m[n[P_1 | P_2] | Q_1\{x := n\} | Q_2]$

(EXIT) $n[m[\text{out}\langle n, k \rangle.P_1 | P_2] | Q] \mid \overline{\text{out}}(x, k).R \longrightarrow m[P_1 | P_2] \mid n[Q] | R\{x := m\}$

communication

(LOCAL) $(\tilde{x}).P \mid \langle \tilde{M} \rangle.Q \longrightarrow P\{\tilde{x} := \tilde{M}\} \mid Q$

(INPUT n) $(\tilde{x})^n.P \mid n[\langle \tilde{M} \rangle^\wedge.Q \mid R] \longrightarrow P\{\tilde{x} := \tilde{M}\} \mid n[Q \mid R]$

(OUTPUT n) $\langle \tilde{M} \rangle^n.P \mid n[(\tilde{x})^\wedge.Q \mid R] \longrightarrow P \mid n[Q\{\tilde{x} := \tilde{M}\} \mid R]$

structural rules

(STRUCT)
$$\frac{P \equiv P', \quad P' \longrightarrow Q', \quad Q' \equiv Q}{P \longrightarrow Q}$$

(CONTEXT)
$$P \longrightarrow Q \Rightarrow \mathbf{E}\{P\} \longrightarrow \mathbf{E}\{Q\}$$

Evaluation context
$$\mathbf{E} ::= \{ \cdot \} \mid \mathbf{E} | P \mid P | \mathbf{E} \mid (\nu n)\mathbf{E} \mid n[\mathbf{E}]$$

Table 1: Syntax and Reduction Rules

of the calculus, and its behavioural theory, given the presence of the newly introduced synchronisation mechanisms based on binding and passwords. The observation predicate $P \downarrow_n$, and the resulting notion of observational congruence are defined below.

Definition 1.1 (Barbs). Given a process P , we write $P \downarrow_n$ if $P \equiv (\nu \tilde{m})(n[\overline{\text{in}}(x, k).Q \mid R] \mid S)$ for $\{n, k\} \cap \{\tilde{m}\} = \emptyset$. We write $P \Downarrow_n$ if $P \Longrightarrow P'$ and $P' \downarrow_n$, where \Longrightarrow is the reflexive and transitive closure of \longrightarrow . \square

Definition 1.2. A relation \mathcal{R} is *reduction closed* if $P \mathcal{R} Q$ and $P \longrightarrow P'$ imply the existence of some Q' such that $Q \Longrightarrow Q'$ and $P' \mathcal{R} Q'$. \mathcal{R} is *barb preserving* if $P \mathcal{R} Q$ and $P \downarrow_n$ imply $Q \downarrow_n$. \square

Definition 1.3 (Reduction Barbed Congruence). Reduction barbed congruence, written \cong , is the largest equivalence relation that is preserved by contexts (i.e. is a congruence) and, when restricted to closed processes, is reduction closed and barb preserving. \square

Notice that the choice of barb is different from the one we used in [9], reflecting here the new observable interactions that an ambient may engage with the context, via mobility. Indeed, we could still rely on our original definition of observation: as we shall prove, the barbed congruence relation we just defined has the extensional property we expect, namely it is independent of the particular choice of the barb (cf. Theorem 2.7).

2 Labelled Transition Semantics

We now prepare the ground for a characterisation of reduction barbed congruence in terms of a labelled bisimilarity. Because of its co-inductive nature, the latter will provide powerful proof techniques for establishing equivalences [16, 18, 17].

The labelled transition semantics is given in terms of the reductions collected in Tables 3–5. To ease the notation, we present the transitions the monadic version of the calculus; the case of polyadic NBA is straightforward. The transitions are of the form $P \xrightarrow{\alpha} O$, where O is an “outcome.” The label α , defined in Table 2, codifies the context with which P may interact, as usual.

<i>Prefixes</i>	$\mu ::= \text{in}\langle n, k \rangle \mid \text{out}\langle n, k \rangle \mid (M)^\eta \mid \langle - \rangle^\eta \mid \overline{\text{in}}(m, k) \mid \overline{\text{out}}(m, k)$
<i>Labels</i>	$\alpha ::= \tau \mid \mu \mid \text{enter}\langle n, k \rangle \mid m \overline{\text{enter}}(n, k) \mid \text{exit}\langle n, k \rangle \mid \text{pop}\langle k \rangle \mid m \text{ get } M \mid m \text{ put } \langle - \rangle$
<i>Concretions</i>	$K ::= (\nu \tilde{m})\langle P \rangle Q \mid (\nu \tilde{m})\langle M \rangle P$
<i>Outcomes</i>	$O ::= P \mid K$

Table 2: Labels, concretions and outcomes

The outcome O is either a process Q , when α is a prefix or the silent action, or a *concretion* of the forms $(\nu \tilde{p})\langle P \rangle Q$ and $(\nu \tilde{p})\langle M \rangle Q$, with P and Q processes, and M an expression. Intuitively, in $(\nu \tilde{p})\langle P \rangle Q$ process P , the *prime*, represents the sub-component of the system that interacts with the environment, while in $(\nu \tilde{p})\langle M \rangle Q$, the expression M represents a piece of information that is transmitted to the environment. In both cases the process Q represents the remaining components of the process that are not affected by the interaction with the environment, and \tilde{p} is the set of private names shared by P (or M) and Q .

Although our bisimilarity will consider only transitions from process to process, the transitions having concretions as derivatives are useful to formally define the τ -transitions of the system. More precisely, concretions represent partial derivatives which need a contribution from the environment to be completed (such contribution is modelled, in §3, via corresponding higher-order transitions). We use the following conventions.

▷ if O is the concretion $(\nu \tilde{p})\langle P \rangle Q$, then:

▷ $(\nu r)O = (\nu \tilde{p})\langle P \rangle (\nu r)Q$, if $r \notin \text{fn}(P)$, and $(\nu r)O = (\nu r, \tilde{p})\langle P \rangle Q$ otherwise;

<p>(CAP)</p> $\frac{M \in \{\text{in}\langle n, k \rangle, \text{out}\langle n, k \rangle\}}{M.P \xrightarrow{M} P}$	<p>(CO-CAP)</p> $\frac{\pi(x) \in \{\overline{\text{in}}(x, k), \overline{\text{out}}(x, k)\}}{\pi(x).P \xrightarrow{\pi(n)} P\{x := n\}}$	<p>(PATH)</p> $\frac{M_1.(M_2.P) \xrightarrow{\alpha} P'}{(M_1.M_2).P \xrightarrow{\alpha} P'}$
<p>(INPUT)</p> $\frac{}{(x)^\eta.P \xrightarrow{(M)^\eta} P\{x := M\}}$	<p>(OUTPUT)</p> $\frac{}{\langle M \rangle^\eta.P \xrightarrow{\langle - \rangle^\eta} (\nu)\langle M \rangle P}$	
<p>(GET)</p> $\frac{P \xrightarrow{(M)^\wedge} P_1}{m[P] \xrightarrow{m \text{ get } M} m[P_1]}$	<p>(PUT)</p> $\frac{P \xrightarrow{\langle - \rangle^\wedge} (\nu \tilde{p})\langle M \rangle P_1 \quad (m \notin \{\tilde{p}\})}{m[P] \xrightarrow{m \text{ put } \langle - \rangle} (\nu \tilde{p})\langle M \rangle m[P_1]}$	
<p>(ENTER)</p> $\frac{P \xrightarrow{\text{in}\langle n, k \rangle} P'}{m[P] \xrightarrow{\text{enter}\langle n, k \rangle} (\nu)\langle m[P'] \rangle \mathbf{0}}$	<p>(CO-ENTER)</p> $\frac{P \xrightarrow{\overline{\text{in}}(n, k)} P'}{m[P] \xrightarrow{m \overline{\text{enter}}(n, k)} (\nu)\langle P' \rangle \mathbf{0}}$	
<p>(EXIT)</p> $\frac{P \xrightarrow{\text{out}\langle n, k \rangle} P'}{m[P] \xrightarrow{\text{exit}\langle n, k \rangle} (\nu)\langle m[P'] \rangle \mathbf{0}}$	<p>(POP)</p> $\frac{P \xrightarrow{\text{exit}\langle n, k \rangle} (\nu \tilde{p})\langle m[P_1] \rangle P_2}{n[P] \xrightarrow{\text{pop}\langle k \rangle} (\nu \tilde{p})\langle m \rangle (m[P_1] \mid n[P_2])}$	

Table 3: Commitments: Visible transitions

$$\triangleright O \mid R = (\nu \tilde{p})\langle P \rangle (Q \mid R),$$

where \tilde{p} are chosen so that $r \notin \{\tilde{p}\}$ and $\text{fn}(R) \cap \{\tilde{p}\} = \emptyset$.

\triangleright if O is the concretion $(\nu \tilde{p})\langle M \rangle P$, then:

$$\triangleright (\nu r)O \text{ is } (\nu \tilde{p})\langle M \rangle ((\nu r)P), \text{ if } r \notin \text{fn}(M), \text{ and } (\nu r, \tilde{p})\langle M \rangle Q \text{ otherwise;}$$

$$\triangleright O \mid R = (\nu \tilde{p})\langle M \rangle (P \mid R),$$

where again \tilde{p} are chosen so that $r \notin \{\tilde{p}\}$ and $\text{fn}(R) \cap \{\tilde{p}\} = \emptyset$.

The labelled transition system builds on those in [11, 12]. The main differences are in the transitions for hierarchical communications, distinctive of NBA, and in the transitions for mobility, as in the latter need to account for the binding of names that arises upon mobility. A further difference is in our use of a standard structural rule for parallel composition, as opposed to the ad-hoc rule (PAR EXIT) in [12].

$$P \xrightarrow{\text{enter}\langle m,k \rangle} (\nu \tilde{p})\langle n[P_1] \rangle P_2$$

$$Q \xrightarrow{m \overline{\text{enter}}(n,k)} (\nu \tilde{q})\langle Q_1 \rangle Q_2$$

$$\begin{aligned} (\text{fn}(P_1) \cup \text{fn}(P_2)) \cap \{\tilde{q}\} &= \emptyset \\ \text{fn}(Q) \cap \{\tilde{p}\} &= \emptyset \end{aligned}$$

$$P \mid Q \xrightarrow{\tau} (\nu \tilde{p}, \tilde{q})(m[Q_1 \mid n[P_1]] \mid P_2 \mid Q_2)$$

$$(\tau\text{-EXIT})$$

$$\frac{P \xrightarrow{\text{pop}\langle k \rangle} (\nu \tilde{p})\langle m \rangle P_1 \quad Q \xrightarrow{\overline{\text{out}}(m,k)} Q_1 \quad \tilde{p} \cap \text{fn}(Q) = \emptyset}{P \mid Q \xrightarrow{\tau} (\nu \tilde{p})(P_1 \mid Q_1)}$$

$$(\tau\text{-EXCHANGE})$$

$$\frac{P \xrightarrow{\langle M \rangle} P_1 \quad Q \xrightarrow{\langle - \rangle} (\nu \tilde{q})\langle M \rangle Q_1 \quad \text{fn}(P) \cap \{\tilde{q}\} = \emptyset}{P \mid Q \xrightarrow{\tau} (\nu \tilde{q})(P_1 \mid Q_1)}$$

$$(\tau\text{-PUT})$$

$$\frac{P \xrightarrow{\langle - \rangle^n} (\nu \tilde{p})\langle M \rangle P_1 \quad Q \xrightarrow{n \text{ get } M} Q_1 \quad \text{fn}(Q) \cap \{\tilde{p}\} = \emptyset}{P \mid Q \xrightarrow{\tau} (\nu \tilde{p})(P_1 \mid Q_1)}$$

$$(\tau\text{-GET})$$

$$\frac{P \xrightarrow{\langle M \rangle^n} P_1 \quad Q \xrightarrow{n \text{ put } \langle - \rangle} (\nu \tilde{q})\langle M \rangle Q_1 \quad \text{fn}(P) \cap \{\tilde{q}\} = \emptyset}{P \mid Q \xrightarrow{\tau} (\nu \tilde{q})(P_1 \mid Q_1)}$$

Table 4: Commitments: τ transitions

(PAR)	(RES)	(τ-AMB)	(REPL)
$P \xrightarrow{\alpha} O$	$P \xrightarrow{\alpha} O \quad n \notin \text{fn}(\alpha)$	$P \xrightarrow{\tau} P'$	$\pi.P \xrightarrow{\alpha} O$
$P \mid Q \xrightarrow{\alpha} O \mid Q$	$(\nu n)P \xrightarrow{\alpha} (\nu n)O$	$n[P] \xrightarrow{\tau} n[P']$	$!\pi.P \xrightarrow{\alpha} !\pi.P \mid O$

Table 5: Commitments: Structural transitions

The transitions for non-local exchanges are defined by the rules (PUT n), (GET n) and their τ -counterparts (τ -PUT), (τ -EXCHANGE) and (τ -GET): they all should be self-explanatory. A few remarks are in order for the movement transitions. The rule (CO-ENTER) says that ambient $m[P]$ is willing to accept an incoming ambient n exhibiting the password k . Dually, the rule (ENTER) leaves in the prime position the ambient involved in the move. The two rules synchronise in the rule (τ ENTER). The treatment of out moves is more complex, and requires three steps. Rule (EXIT) isolates the exiting ambient in the prime of the concretion, leaving the process that will not move in the residual. Then, the (EXIT) rule completes the move by leaving the name m of the exiting ambient in a buffer. This name should then match the name that is expected by the accepting context, as required in the rule (τ -EXIT).

Next, we show that the labelled transition semantics coincides with the reduction semantics. The proof is not difficult, but long. We first need to extend the definition of structural congruence to concretions. That can be accomplished as follows:

- $\triangleright (\nu \tilde{p})\langle P \rangle Q \equiv (\nu \tilde{p})\langle P' \rangle Q'$ if $P \equiv P'$ and $Q \equiv Q'$
- $\triangleright (\nu \tilde{p})\langle M \rangle P \equiv (\nu \tilde{p})\langle M \rangle P'$ if $P \equiv P'$.

Then we prove the following two preliminary lemmas. The first describes the structure of processes and outcomes involved in the labelled transitions (we only give the cases that involve ‘in’ moves: the other cases are similar). The second relates labelled transitions and structural congruence.

Lemma 2.1.

1. If $P \xrightarrow{\text{in}\langle m,k \rangle} P'$ then there exist names \tilde{p} , with $\{m, k\} \cap \{\tilde{p}\} = \emptyset$, and processes P_1, P_2 such that $P \equiv (\nu \tilde{p})(\text{in}\langle m, k \rangle.P_1 \mid P_2)$ and $P' \equiv (\nu \tilde{p})(P_1 \mid P_2)$.
2. If $P \xrightarrow{\overline{\text{in}}\langle m,k \rangle} P'$ then there exist names \tilde{p} , with $\{m, k\} \cap \{\tilde{p}\} = \emptyset$, and processes P_1, P_2 such that $P \equiv (\nu \tilde{p})(\overline{\text{in}}\langle x, k \rangle.P_1 \mid P_2)$ and $P' \equiv (\nu \tilde{p})(P_1 \{x := m\} \mid P_2)$.
3. If $P \xrightarrow{\text{enter}\langle m,k \rangle} O$ then there exist names \tilde{p}, n , with $\{m, k\} \cap \{\tilde{p}\} = \emptyset$, and processes P_1, P'_1, P_2 such that $P \equiv (\nu \tilde{p})(n[P_1] \mid P_2)$, $P_1 \xrightarrow{\text{in}\langle m,k \rangle} P'_1$, and $O \equiv (\nu \tilde{p})(n[P'_1] \mid P_2)$.
4. If $P \xrightarrow{m \overline{\text{enter}}\langle n,k \rangle} O$ then there exist names \tilde{p} , with $\{m, n, k\} \cap \{\tilde{p}\} = \emptyset$, and processes P_1, P'_1, P_2 such that $P \equiv (\nu \tilde{p})(m[P_1] \mid P_2)$, $P_1 \xrightarrow{\overline{\text{in}}\langle n,k \rangle} P'_1$, and $O \equiv (\nu \tilde{p})(P'_1 \mid P_2)$.

Proof. By transition induction. □

Lemma 2.2. If $P \xrightarrow{\alpha} O$ and $P \equiv Q$, then there exists O' such that $Q \xrightarrow{\alpha} O'$ and $O \equiv O'$.

Proof. By induction on the derivation of $P \equiv Q$. As it often happens in proofs involving structural congruence, to handle the law of symmetry we prove the following two statements, by simultaneous induction on the derivations of $P \equiv Q$ ($Q \equiv P$).

1. If $P \xrightarrow{\alpha} O$ and $P \equiv Q$, then there exists O' such that $Q \xrightarrow{\alpha} O'$ and $O \equiv O'$.
2. If $P \xrightarrow{\alpha} O$ and $Q \equiv P$, then there exists O' such that $Q \xrightarrow{\alpha} O'$ and $O \equiv Q'$.

The inductive cases are standard. There are a multitude of base cases, which also are rather standard. We give just one case to illustrate the role of the side-conditions on the τ -transitions of Table 4. Note, to this regard, that all the τ -transitions have the side condition $\tilde{p} \cap \text{fn}(Q) = \emptyset$ (or dually $\tilde{q} \cap \text{fn}(P) = \emptyset$): this condition is needed to capture the effect of scope extrusion, as all such transition involve the transmission of possibly private names (the name of the moving ambient for the transitions (τ -ENTER) and (τ -EXIT)).

To illustrate, in case (1), take the sub-case¹ when $P \equiv Q$ is $(\nu l)(P \mid Q) \equiv (\nu l)P \mid Q$, for $l \notin \text{fn}(Q)$. Then the labelled transition must be of the form $(\nu l)(P \mid Q) \xrightarrow{\alpha} (\nu l)O$, derived by (RES) from $P \mid Q \xrightarrow{\alpha} O$ for $l \notin \text{fn}(\alpha)$. Of the many possible cases to analyse, let us focus the one where α is the silent action and the last transition is derived by (τ -ENTER) from $P \xrightarrow{\text{enter}\langle m, k \rangle} (\nu \tilde{p})\langle n[P_1] \rangle P_2$ and $Q \xrightarrow{m \text{ enter}\langle n, k \rangle} (\nu \tilde{q})\langle Q_1 \rangle Q_2$, where $\{\tilde{q}\} \cap \text{fn}(P_1, P_2) = \{\tilde{p}\} \cap \text{fn}(Q) = \emptyset$ and $O \equiv (\nu \tilde{p}, \tilde{q})(m[Q_1 \mid n[P_1]] \mid P_2 \mid Q_2)$.

We need to show that $(\nu l)P \mid Q \xrightarrow{\tau} (\nu l)O$. To see that, we first observe that $l \neq m, k$, as $l \notin \text{fn}(Q)$ by hypothesis, and $\{m, k\} \subseteq \text{fn}(Q)$ as it can be shown by transition induction. Thus from $P \xrightarrow{\text{enter}\langle m, k \rangle} (\nu \tilde{p})\langle n[P_1] \rangle P_2$, we derive $(\nu l)P \xrightarrow{\text{enter}\langle m, k \rangle} (\nu l)((\nu \tilde{p})\langle n[P_1] \rangle P_2)$ by (RES). Now we distinguish the two cases that arise from two possible formats of the outcome of this last transition.

In the first case we have $(\nu l)P \xrightarrow{\text{enter}\langle m, k \rangle} (\nu l, \tilde{p})\langle n[P_1] \rangle P_2$. This, together with the transition from Q , yields $(\nu l)P \mid Q \xrightarrow{\tau} (\nu l, \tilde{p}, \tilde{q})(m[Q_1 \mid n[P_1]] \mid P_2 \mid Q_2) \equiv (\nu l)O$, by (τ -ENTER). The side conditions to the rule are satisfied thanks to the hypotheses on \tilde{p} and \tilde{q} and to the additional condition $l \notin \text{fn}(Q)$. Note that the proof would *not* go through had we replaced the side condition $\{\tilde{p}\} \cap \text{fn}(Q) = \emptyset$ in rule (τ -ENTER) with $\{\tilde{p}\} \cap \text{fn}(Q_1, Q_2) = \emptyset$ from [11, 12]. In particular, the latter condition could be violated by l , as $l \notin \text{fn}(Q)$ does not imply that $l \notin \text{fn}(Q_1, Q_2)$, for l could be n , which may occur free in Q_1 .

Otherwise the transition in question is $(\nu l)P \xrightarrow{\text{enter}\langle m, k \rangle} (\nu \tilde{p})\langle n[P_1] \rangle (\nu l)P_2$, which implies that $l \neq n$. From this, and from the transition from Q , we derive $(\nu l)P \mid Q \xrightarrow{\tau} \tau(\nu \tilde{p}, \tilde{q})(m[Q_1 \mid n[P_1]] \mid (\nu l)P_2 \mid Q_2)$. Finally, from the hypothesis $l \notin \text{fn}(Q)$ and the fact that $l \neq n, m$, it follows that $(\nu l)O \equiv (\nu \tilde{p}, \tilde{q})(m[Q_1 \mid n[P_1]] \mid (\nu l)P_2 \mid Q_2)$. \square

We are finally ready to establish the desired connection between the reduction and the labelled transition semantics.

Theorem 2.3.

1. If $P \xrightarrow{\tau} P'$ then $P \longrightarrow P'$
2. If $P \longrightarrow P'$ then $P \xrightarrow{\tau} P'$

¹ This sub-case should rather be written as $(\nu l)(P \mid Q) \equiv P \mid (\nu l)Q$, but the equivalence as given is consistent with the format of the (τ -ENTER) rule displayed in Table 4, where it is P that contains the moving ambient, whose name is transmitted with the move.

Proof. By transition induction, and a case analysis on the last rule applied in the derivation of the hypothesis. The proof of (1) appeals to Lemma 2.1 to reconstruct the structure of P and P' . We give the case (τ -ENTER) as representative. In this case, the transition in question is $P \mid Q \xrightarrow{\tau} (\nu \tilde{p}, \tilde{q})(m[Q_1 \mid n[P_1]] \mid P_2 \mid Q_2)$, derived from

$$P \xrightarrow{\text{enter}\langle m, k \rangle} (\nu \tilde{p})(n[P_1])P_2, \quad Q \xrightarrow{m \overline{\text{enter}}(n, k)} (\nu \tilde{q})(Q_1)Q_2$$

with $\text{fn}(P_1, P_2) \cap \tilde{q} = \text{fn}(Q) \cap \tilde{p} = \emptyset$. By (repeated applications of) Lemma 2.1 there exist $\tilde{r}, \tilde{s}, R_1, R_2, S_1, S_2$ such that

$$P \equiv (\nu \tilde{p})(n[(\nu \tilde{r})\text{in}\langle m, k \rangle.R_1 \mid R_2] \mid P_2) \mid (\nu \tilde{q})(m[(\nu \tilde{s})\overline{\text{in}}(x, k).S_1 \mid S_2] \mid Q_2)$$

with $P_1 = (\nu \tilde{r})(R_1 \mid R_2)$ and $Q_1 = (\nu \tilde{s})(S_1 \{x := n\} \mid S_2)$. By choosing the bound names \tilde{r} and \tilde{s} appropriately, we may rearrange P by structural congruence, as in

$$P \equiv (\nu \tilde{p}, \tilde{q})(\nu \tilde{r}, \tilde{s})(n[\text{in}\langle m, k \rangle.R_1 \mid R_2] \mid m[\overline{\text{in}}(x, k)S_1 \mid S_2] \mid P_2 \mid Q_2).$$

Then

$$P \longrightarrow (\nu \tilde{p}, \tilde{q})(m[(\nu \tilde{s})(S_1 \{x := n\} \mid S_2) \mid n[(\nu \tilde{r})(R_1 \mid R_2)]] \mid P_2 \mid Q_2)$$

by an (ENTER) reduction followed by rearrangements via structural congruence.

The proof of (2) is also by transition induction. It needs Lemma 2.2, with O a process, to handle the case when $P \longrightarrow P'$ by (STRUCT). \square

We now re-examine our definition of barbed congruence \cong in the light of the new labelled transition semantics. As already mentioned, the predicate $P \downarrow_n$ detects the ability of the process P to interact with its environment via the ambient n . We start by noting that our definition of barb coincides with the choice of one particular action.

Lemma 2.4. $P \downarrow_n$ if and only if $P \xrightarrow{n \overline{\text{enter}}(m, k)}$ for some m, k .

Proof. Directly by the definition of $P \downarrow_n$ and an inspection of the transition rules. \square

We now study how the definition of barbed congruence is affected by inheriting the definition of barb from the labelled transition system. More precisely, we show that for all possible labels generated by the labelled transitions, the corresponding definitions of barbed congruence collapse, and coincide with \cong . We write $P \xrightarrow{\alpha}$ to say that $P \xrightarrow{\alpha} P'$ for some P' . In force of Theorem 2.3, in the following we confuse \Longrightarrow and $\xrightarrow{\tau}^*$.

Definition 2.5. For $\alpha \in \text{Labels}$ we write $P \downarrow_\alpha$ if $P \xrightarrow{\alpha}$, and $P \Downarrow_\alpha$ if $P \Longrightarrow \xrightarrow{\alpha}$. Let then $\alpha \in \text{Labels} \setminus \{\tau\}$, and define \cong_α to be the largest congruence that, when restricted to closed processes, is reduction closed and preserves α -barbs, i.e. $P \cong_\alpha Q$ and $P \downarrow_\alpha$ implies $Q \downarrow_\alpha$. \square

Proposition 2.6. Assume $P \cong_\alpha Q$. Then

1. $P \Longrightarrow P'$ implies $Q \Longrightarrow Q'$ for some Q' such that $P' \cong_\alpha Q'$;

2. $P \Downarrow_\alpha$ if and only if $Q \Downarrow_\alpha$.

Proof. Part (1) is proved by induction on the number of steps in $P \Longrightarrow P'$. If $P' = P$, then choose $Q' = Q$. Otherwise, assume $P \longrightarrow P^* \Longrightarrow P'$ in $n + 1$ steps. Since $P \cong_\alpha Q$, there exists Q^* such that $Q \Longrightarrow Q^*$ and $P^* \cong_\alpha Q^*$. Now the proof follows by the induction hypothesis.

For part (2), assume $P \Downarrow_\alpha$. By definition, $P \Longrightarrow P' \downarrow_\alpha$ for some P' . Since $P \cong_\alpha Q$, by part (1) there exists Q' such that $Q \Longrightarrow Q'$ and $P' \cong_\alpha Q'$. Thus $Q \Longrightarrow Q' \downarrow_\alpha$. \square

Theorem 2.7. For all $\alpha \in \text{Labels} \setminus \{\tau\}$, $P \cong Q$ if and only if $P \cong_\alpha Q$.

Proof. Since the definitions of \cong and \cong_α differ only in the notion of barb, it is enough to show that the two barbs imply each other.

- ▷ $\alpha = n \text{ put } \langle - \rangle$. Consider the implication from left to right first. Let $P \cong Q$ and $P \downarrow_{n \text{ put } \langle - \rangle}$: we want to show that $Q \downarrow_{n \text{ put } \langle - \rangle}$. Consider the following context, where ℓ is fresh in P and Q :

$$C[\cdot] \triangleq [\cdot] \mid (x)^n \ell[\overline{\text{in}}(x, k). \mathbf{0}].$$

Given any R with ℓ fresh in R , it is easy to show that $R \downarrow_{n \text{ put } \langle - \rangle}$ if and only if $C[R] \downarrow_\ell$. This is enough to complete the proof, for $P \downarrow_{n \text{ put } \langle - \rangle}$ implies $C[P] \downarrow_{n \text{ put } \langle - \rangle}$, and since $P \cong Q$, one has $C[Q] \downarrow_{n \text{ put } \langle - \rangle}$ which implies $Q \downarrow_{n \text{ put } \langle - \rangle}$.

For the reverse implication, let $P \cong_{n \text{ put } \langle - \rangle} Q$, and $P \downarrow_n$. Consider the context defined as follows:

$$C^k[\cdot] \triangleq [\cdot] \mid \ell[\text{in}(n, k). \text{out}(n, \ell). \langle \cdot \rangle^\wedge] \mid \overline{\text{out}}(x, \ell). \mathbf{0}.$$

Given any R with ℓ fresh in R , it is easily shown that

- ▷ if $R \downarrow_n$ then there exists k such that $C^k[R] \downarrow_{\ell \text{ put } \langle - \rangle}$;
- ▷ $C^k[R] \downarrow_{\ell \text{ put } \langle - \rangle}$ implies $R \downarrow_n$.

Now, $P \downarrow_n$ implies that there exists k such that $C^k[P] \downarrow_{\ell \text{ put } \langle - \rangle}$. Thus we have $C^k[Q] \downarrow_{\ell \text{ put } \langle - \rangle}$, and then $Q \downarrow_n$ as desired.

- ▷ $\alpha = \text{pop}(k)$. For the implication from left to right, choose the following context, with ℓ fresh in P and Q :

$$C[\cdot] \triangleq [\cdot] \mid \overline{\text{out}}(-, k). \ell[\overline{\text{in}}(-, h)].$$

The proof proceeds as in the previous case as for all R with $\ell \notin \text{fn}(R)$, we have $R \downarrow_{\text{pop}(k)}$ if and only if $C[R] \downarrow_\ell$. For the reverse implication, choose the context:

$$C^k[\cdot] \triangleq [\cdot] \mid \ell[\text{in}(n, k). \text{out}(n, h)].$$

with h fresh. For each R with $h \notin \text{fn}(R)$, we have (i) $R \downarrow_n$ implies that $C^k[R] \downarrow_{\text{pop}(h)}$ for a suitable k , and (ii) $C^k[R] \downarrow_{\text{pop}(h)}$ implies $R \downarrow_n$. From this, we conclude as in the previous cases.

▷ $\alpha = \text{exit}\langle n, k \rangle$. For the implication from left to right, choose the context

$$C[\cdot] \triangleq n[[\cdot]] \mid \overline{\text{out}}(-, k). \ell[\overline{\text{in}}(-, h)].$$

Again, if $\ell \notin \text{fn}(R)$, one has $R \Downarrow_{\text{exit}\langle n, k \rangle}$ if and only if $C[R] \Downarrow_\ell$. For the reverse implication, choose the context:

$$C^k[\cdot] \triangleq [\cdot] \mid \ell[\text{in}\langle n, k \rangle. \text{out}\langle n, h \rangle. \text{out}\langle \ell, h \rangle] \mid \overline{\text{out}}(-, h)$$

with h fresh, and verify that $R \Downarrow_n$ if and only if $C^k[R] \Downarrow_{\text{exit}\langle \ell, h \rangle}$.

▷ $\alpha = \text{in}\langle n, k \rangle$. For the implication from left to right, choose the context

$$C[\cdot] \triangleq a[[\cdot]] \mid n[\overline{\text{in}}(-, k). b[\text{out}\langle n, h \rangle. \overline{\text{in}}(-, k)]] \mid \overline{\text{out}}(-, h)$$

with a, b, h fresh, and verify that $R \Downarrow_{\text{in}\langle n, k \rangle}$ if and only if $R \Downarrow_b$. For the reverse implication, choose

$$C^k[\cdot] \triangleq [\cdot] \mid a[\text{in}\langle n, k \rangle. \text{out}\langle n, h \rangle] \mid \overline{\text{out}}(-, h). \text{in}\langle a, h \rangle$$

with a, h fresh, and verify that $P \Downarrow_n$ if and only if $C^k[P] \Downarrow_{\text{in}\langle a, h \rangle}$.

▷ The other cases are handled similarly. □

Notice that in the proof above we have used $_$ to denote a “dummy” bound variable. By that we mean that $_$ appears only in binding occurrences. We will use such notation again.

3 Labelled Bisimilarity

In this section we provide a sound characterisation of barbed congruence in terms of (weak) labelled bisimilarity. To define the latter, we need a way to test the equivalence of processes after any (number of τ transition following any) visible transition. To account for that, we introduce a new, higher-order, transition for each of the first-order transitions whose outcome is a concretion, rather than a process.

The new transitions are collected in Table 6. The higher-order labels occurring in these transitions encode the minimal contribution by the environment needed by the process to complete a transition. Thus, in (PUT HO) and (OUTPUT HO) the process Q represents the context receiving the value M output by P , and the variable x is a placeholder for that value. The rule (OUTPUT $\hat{\text{HO}}$) is similar, but more complex because the value output by P will be received at a different nesting level. In particular, to complete its output, P needs to be placed into an ambient n (possibly containing a sibling process Q) and the value M output by P will be received at the enclosing nesting level.

The higher-order transitions for mobility have the same rationale. Thus, for instance, in the rule (CO-ENTER HO) the environment provides an ambient $n[Q]$ moving into m . In the rule (EXIT HO) we can imagine the environment wrapping the process P with an ambient $n[Q]$, and receiving the name m of the exiting ambient at R .

(OUTPUT HO)	
$\frac{P \xrightarrow{\langle - \rangle^\eta} (\nu \tilde{p}) \langle M \rangle P' \quad \text{fv}(Q) \subseteq \{x\}, \tilde{p} \cap \text{fn}(Q) = \emptyset, \eta \neq \hat{\cdot}}{P \xrightarrow{\langle - \rangle^\eta Q} (\nu \tilde{p}) (P' \mid Q\{x := M\})}$	
(OUTPUT $\hat{\cdot}$ HO)	(PUT HO)
$\frac{P \xrightarrow{\langle - \rangle^{\hat{\cdot}}} (\nu \tilde{p}) \langle M \rangle P' \quad \text{fv}(R) \subseteq \{x\}, \tilde{p} \cap \text{fn}(n[Q], R) = \emptyset}{P \xrightarrow{\langle - \rangle^{\hat{\cdot}} n[Q] R} (\nu \tilde{p}) (n[P' \mid Q] \mid R\{x := M\})}$	$\frac{P \xrightarrow{m \text{ put } \langle - \rangle} (\nu \tilde{p}) \langle M \rangle P' \quad \text{fv}(Q) \subseteq \{x\}, \tilde{p} \cap \text{fn}(Q) = \emptyset}{P \xrightarrow{m \text{ put } \langle - \rangle Q} (\nu \tilde{p}) (P' \mid Q\{x := M\})}$
(ENTER HO)	(CO-ENTER HO)
$\frac{P \xrightarrow{\text{enter} \langle n, k \rangle} (\nu \tilde{p}) \langle m[P_1] \rangle P_2 \quad \text{fv}(Q) \subseteq \{x\}, \tilde{p} \cap \text{fn}(Q) = \emptyset}{P \xrightarrow{\text{enter} \langle n, k \rangle Q} (\nu \tilde{p}) (n[m[P_1] \mid Q\{x := m\}] \mid P_2)}$	$\frac{P \xrightarrow{m \text{ enter} \langle n, k \rangle} (\nu \tilde{p}) \langle P_1 \rangle P_2 \quad \tilde{p} \cap \text{fn}(Q) = \emptyset}{P \xrightarrow{m \text{ enter} \langle n, k \rangle Q} (\nu \tilde{p}) (m[n[Q] \mid P_1] \mid P_2)}$
(EXIT HO)	(POP HO)
$\frac{P \xrightarrow{\text{exit} \langle n, k \rangle} (\nu \tilde{p}) \langle m[P_1] \rangle P_2 \quad \text{fv}(R) \subseteq \{x\}, \tilde{p} \cap \text{fn}(Q, R) = \emptyset}{P \xrightarrow{\text{exit} \langle n, k \rangle QR} (\nu \tilde{p}) (m[P_1] \mid n[P_2 \mid Q] \mid R\{x := m\})}$	$\frac{P \xrightarrow{\text{pop} \langle k \rangle} (\nu \tilde{p}) \langle m \rangle P' \quad \text{fv}(Q) \subseteq \{x\}, \tilde{p} \cap \text{fn}(Q) = \emptyset}{P \xrightarrow{\text{pop} \langle k \rangle Q} (\nu \tilde{p}) (P' \mid Q\{x := m\})}$

Table 6: Commitments: Higher-Order Transitions

Having defined the new higher-order transitions, we are now ready to give the relation of labelled bisimilarity. Let Λ be the set of all labels including the first-order labels of Table 2 as well as the higher-order labels determined by the transitions in Table 6. We denote with λ any label in the set Λ . As usual, we focus on weak bisimilarities based on weak transitions, and use the following notation:

- i) $\xRightarrow{\lambda}$ denotes $\Longrightarrow \xrightarrow{\lambda} \Longrightarrow$
- ii) $\xRightarrow{\hat{\lambda}}$ denotes \Longrightarrow if $\lambda = \tau$ and $\xRightarrow{\lambda}$ otherwise.

Definition 3.1 (Bisimilarity). A symmetric relation \mathcal{R} over closed processes is a *bisimulation* if $P \mathcal{R} Q$ and $P \xrightarrow{\lambda} P'$ imply that there exists Q' such that $Q \xRightarrow{\hat{\lambda}} Q'$ and $P' \mathcal{R} Q'$. Two processes P and Q are bisimilar, written $P \approx Q$, if $P \mathcal{R} Q$ for some bisimulation \mathcal{R} . \square

This definition of bisimilarity is only defined over closed processes. We generalise it to arbitrary processes as follows:

Definition 3.2 (Full bisimilarity). Two processes P and Q are *full bisimilar*, $P \approx_c Q$, if $P\sigma \approx Q\sigma$ for every closing substitution σ . \square

Note that the definition of bisimilarity only tests transitions from processes to processes, which typically involve higher-order actions. To this regard, it is important to point out that the structural rules of Table 5 only apply when $\lambda \in \text{Labels}$: in other words, there are no structural rules associated with higher-order transitions. (Observe though that α -conversion and, as a consequence, rearrangement of the order of adjacent restrictions still applies.) We will return to this observation in the proof of Theorem 3.4, where we show that full bisimilarity is a congruence.

Lemma 3.3.

1. If $P \xrightarrow{\text{exit}\langle n,k \rangle 0R} P'$ then $n[P] \xrightarrow{\text{pop}\langle k \rangle R} P'$.
2. If $P \xrightarrow{\langle - \rangle^{\wedge} n[0]R} P'$ then $n[P] \xrightarrow{n \text{ put } \langle - \rangle R} P'$.

Proof. By transition induction. □

Theorem 3.4. *Full bisimilarity is a congruence*

Proof. It is easy to show that \approx_c is preserved by input prefixes (these include, proper input prefixes and co-capability prefixes). For instance, assuming $P \approx_c Q$, we need to show that $(x)^\eta.P\sigma \approx (x)^\eta.Q\sigma$ for all closing substitutions σ . By definition, one has $((x)^\eta.P)\sigma = (x)^\eta.(P\sigma)$ (with σ capture free). The only moves from $(x)^\eta.(P\sigma)$ are of the form $(x)^\eta.(P\sigma) \xrightarrow{(M)} P\sigma\{x := M\}$ for an arbitrary expression (message) M . Since also $(x)^\eta.(Q\sigma) \xrightarrow{(M)} Q\sigma\{x := M\}$, it remains to show that $P\sigma\{x := M\} \approx Q\sigma\{x := M\}$. But this follows directly from the assumption $P \approx_c Q$.

For the remaining constructs we can safely restrict to closed processes in the language, and prove that \approx is a congruence. We treat all the constructs simultaneously, as follows. Let \mathcal{S} be the least equivalence relation that contains \approx and is closed by prefix, parallel composition, restriction and ambient, i.e.:

- ▷ $\approx \subseteq \mathcal{S}$
- ▷ $P \mathcal{S} Q$ implies $\pi.P \mathcal{S} \pi.Q$
- ▷ $P \mathcal{S} Q$ implies $P \mid R \mathcal{S} Q \mid R$ for all processes R
- ▷ $P \mathcal{S} Q$ implies $n[P] \mathcal{S} n[Q]$, $(\nu n)P \mathcal{S} (\nu n)Q$ and $!P \mathcal{S} !Q$.

We show that \mathcal{S} is a bisimulation up to \equiv (cf. [19]). The theorem follows directly from this fact (for, then, \mathcal{S} is itself a bisimulation, hence $\mathcal{S} \subseteq \approx$, which implies $\mathcal{S} = \approx$). The proof is by induction on the formation of \mathcal{S} .

- ▷ $P \mathcal{S} Q$ because $P \approx Q$. This case follows by definition.
- ▷ $\pi.P \mathcal{S} \pi.Q$ because $P \mathcal{S} Q$. There are five sub-cases to consider. If π is a capability, say M , the only move from $M.P$ is of the form $M.P \xrightarrow{M} P$. Then $M.Q \xrightarrow{M} Q$, and this concludes the proof because $P \mathcal{S} Q$ by hypothesis.

The case when π is an input prefix has already been worked out above. There are two more sub-cases for output prefixes.

- ▷ $\pi.P \xrightarrow{\lambda} P'$ because $\pi = \langle M \rangle^\eta$ with $\eta \neq \hat{\cdot}$, $\lambda = \langle - \rangle^\eta R$ and P' is structurally equivalent to $P \mid R\{x := M\}$. The same move is also available to $\langle M \rangle^\eta.Q$, hence one has $\langle M \rangle^\eta \xrightarrow{\lambda} Q \mid R\{x := M\}$. Since $P \mathcal{S} Q$ by hypothesis, and since \mathcal{S} is closed by parallel composition, we conclude $P \mid R\{x := M\} \mathcal{S} Q \mid R\{x := M\}$, as desired.
- ▷ The case when $\pi = \langle M \rangle^{\hat{\cdot}}$ is similar: it also requires the closure of \mathcal{S} by the ambient constructor.
- ▷ $P \mid R \mathcal{S} Q \mid R$ because $P \mathcal{S} Q$. We proceed by a case analysis of why $P \mid R \xrightarrow{\lambda} O$, with O a process (not a concretion). There thirteen cases in all to consider, plus their symmetric cases. We start with the structural case, below.
 - ▷ $P \mid R \xrightarrow{\lambda} P' \mid R$ because $P \xrightarrow{\lambda} P'$. Since $P \mathcal{S} Q$, by induction hypothesis we find a weak transition $Q \xRightarrow{\lambda} Q'$ with $P' \mathcal{S} Q'$. Thus, we also have a weak transition $Q \mid R \xRightarrow{\lambda} Q' \mid R$, and since \mathcal{S} is closed by parallel composition, $P' \mid R \mathcal{S} Q' \mid R$ as desired.

Then there are six cases of τ -transitions, plus their symmetric cases.

- ▷ $P \mid R \xrightarrow{\tau} O$ as $P \xrightarrow{\text{enter}\langle m,k \rangle} (\nu \tilde{p})\langle n[P_1] \rangle P_2$ and $R \xrightarrow{m \overline{\text{enter}}(n,k)} (\nu \tilde{r})\langle R_1 \rangle R_2$, with $O \equiv (\nu \tilde{r})(\nu \tilde{p})(m[R_1 \mid n[P_1]] \mid P_2 \mid R_2)$, and $R_1 \equiv R_x\{x := n\}$ for a suitable R_x . We must find a matching move $Q \mid R \Rightarrow O'$ with $O \mathcal{S} O'$.
 By rule (ENTER HO) one has $P \xrightarrow{\text{enter}\langle m,k \rangle R_x} P' \equiv (\nu \tilde{p})(m[n[P_1] \mid R_1] \mid P_2)$. Since $P \mathcal{S} Q$, by induction hypothesis there exists Q' such that $P' \mathcal{S} Q'$, for which $Q \xRightarrow{\text{enter}\langle m,k \rangle R_x} Q'$. Thus $Q \Rightarrow V \xrightarrow{\text{enter}\langle m,k \rangle R_x} Z \Rightarrow Q'$ for appropriate V and Z . An inspection of the transition rules shows that Z must be of the form $(\nu \tilde{q})(m[l[Q_1] \mid R_x\{x := l\}] \mid Q_2)$ for suitable names l, \tilde{q} and processes Q_1 and Q_2 . Furthermore, the transition $V \xrightarrow{\text{enter}\langle m,k \rangle R_x} Z$ must have been derived from $V \xrightarrow{\text{enter}\langle m,k \rangle} (\nu \tilde{q})(l[Q_1])Q_2$. From $R \xrightarrow{m \overline{\text{enter}}(n,k)} (\nu \tilde{r})\langle R_1 \rangle R_2$, it follows that $R \xrightarrow{m \overline{\text{enter}}(l,k)} (\nu \tilde{r})\langle R_x\{x := l\} \rangle R_2$. Hence by an application of the rule (τ ENTER), we have $Q \mid R \Rightarrow V \mid R \xrightarrow{\tau} (\nu \tilde{r})(Z \mid R_2) \Rightarrow (\nu \tilde{r})(Q' \mid R_2)$. From $P' \mathcal{S} Q'$, since \mathcal{S} is closed by restriction and parallel composition, it follows that $O \equiv (\nu \tilde{r})(P' \mid R_2) \mathcal{S} (\nu \tilde{r})(Q' \mid R_2) \equiv O'$, as desired.
- ▷ $P \mid R \xrightarrow{\tau} O$ because $P \xrightarrow{m \overline{\text{enter}}(n,k)} (\nu \tilde{p})\langle P_1 \rangle P_2$ and $R \xrightarrow{\text{enter}\langle m,k \rangle} (\nu \tilde{r})\langle n[R_1] \rangle R_2$, with $O \equiv (\nu \tilde{r})(\nu \tilde{p})(m[P_1 \mid n[R_1]] \mid R_2 \mid P_2)$. We must find a matching move $Q \mid R \Rightarrow O'$ with $O \mathcal{S} O'$. By an application of rule (CO-ENTER HO) one has $P \xrightarrow{m \overline{\text{enter}}(n,k)R_1} P' \equiv (\nu \tilde{p})(m[n[R_1] \mid P_1] \mid P_2)$, with $\tilde{p} \cap \text{fn}(R_1) = \emptyset$. Since $P \mathcal{S} Q$, there exists Q' such that $Q \Rightarrow V \xrightarrow{m \overline{\text{enter}}(n,k)R_1} Z \Rightarrow Q'$ with $P' \mathcal{S} Q'$.

An inspection of the transition rules shows that $Z \equiv (\nu \tilde{q})(m[n[R_1] \mid Q_1] \mid Q_2)$ for suitable names \tilde{q} , and processes Q_1 and Q_2 . In particular, the transition $V \xrightarrow{m \text{ enter}(n,k)R_1} Z$ must have been derived from $V \xrightarrow{m \overline{\text{enter}}(n,k)} (\nu \tilde{q})\langle Q_1 \rangle Q_2$. Thus, by an application of $(\tau \text{ ENTER})$

$$\begin{aligned} Q \mid R &\Longrightarrow V \mid R \\ &\xrightarrow{\tau} (\nu \tilde{r})(\nu \tilde{q})(m[n[R_1] \mid Q_1] \mid R_2 \mid Q_2) \\ &\equiv (\nu \tilde{r})(Z \mid R_2) \\ &\Longrightarrow (\nu \tilde{r})(Q' \mid R_2) \end{aligned}$$

From $P' \mathcal{S} Q'$, since \mathcal{S} is closed by restriction and parallel composition, it follows that $O \equiv (\nu \tilde{r})(P' \mid R_2) \mathcal{S} (\nu \tilde{r})(Q' \mid R_2) \equiv O'$, as desired.

- ▷ $P \mid R \xrightarrow{\tau} O$ because $P \xrightarrow{\text{pop}(k)} (\nu \tilde{p})\langle m \rangle P'$ and $R \xrightarrow{\overline{\text{out}}(m,k)} R'$, where O structurally equivalent to $(\nu \tilde{p})(P' \mid R')$ and R' is of the form $R_x\{x := m\}$ for a suitable R_x .

By the rule (POP HO), we derive $P \xrightarrow{\text{pop}(k)R_x} O$. Since $P \mathcal{S} Q$, by the induction hypothesis we find a transition $Q \Longrightarrow V \xrightarrow{\text{pop}(k)R_x} Z \Longrightarrow O'$ with $O \mathcal{S} O'$. An inspection of the transition rules shows that $V \xrightarrow{\text{pop}(k)} Z$ must derive from $V \xrightarrow{\text{pop}(k)} (\nu \tilde{r})\langle l \rangle V'$ for suitable V' and l , with $Z \equiv (\nu \tilde{r})(V' \mid R_x\{x := l\})$. Also, from $R \xrightarrow{\overline{\text{out}}(m,k)} R'$, it follows that $R \xrightarrow{\overline{\text{out}}(l,k)} R_x\{x := l\}$. Thus $V \mid R \xrightarrow{\tau} Z$, and we are done, since $Q \mid R \Longrightarrow V \mid R \xrightarrow{\tau} Z \Longrightarrow O'$

- ▷ $P \mid R \xrightarrow{\tau} O$ because $P \xrightarrow{\overline{\text{out}}(m,k)} P'$ and $R \xrightarrow{\text{pop}(k)} (\nu \tilde{r})\langle m \rangle R'$ with O structurally equivalent to $(\nu \tilde{r})(R' \mid P')$. Since $P \mathcal{S} Q$, by induction hypothesis, we know that $Q \Longrightarrow U \xrightarrow{\overline{\text{out}}(m,k)} Z \Longrightarrow Q'$. Thus

$$Q \mid R \Longrightarrow U \mid R \xrightarrow{\overline{\text{out}}(m,k)} (\nu \tilde{r})(R' \mid Z) \Longrightarrow (\nu \tilde{r})(R' \mid Q') \equiv O'.$$

Now, $O \mathcal{S} O'$ derives from $P' \mathcal{S} Q'$ because \mathcal{S} is closed by parallel composition and restriction.

- ▷ $P \mid R \xrightarrow{\tau} O$ because $P \xrightarrow{\langle - \rangle} (\nu \tilde{p})\langle M \rangle P'$, $R \xrightarrow{(M)} R'$ and O is structurally equivalent to $(\nu \tilde{p})(P' \mid R')$ and R' is of the form $R_x\{x := M\}$.

From $P \xrightarrow{\langle - \rangle} (\nu \tilde{p})\langle M \rangle P'$, by (OUTPUT HO) we derive $P \xrightarrow{\langle - \rangle R_x} O$. By induction hypothesis, since $P \mathcal{S} Q$, we have $Q \Longrightarrow U \xrightarrow{\langle - \rangle R_x} Z \Longrightarrow O'$ with $O \mathcal{S} O'$.

The previous higher-order transition must be derived from $U \xrightarrow{\langle - \rangle} (\nu \tilde{q})\langle N \rangle V$ with Z of the form $(\nu \tilde{q})(V \mid R_x\{x := N\})$. Thus, since $R \xrightarrow{(N)} R_x\{x := N\}$, we have $U \mid R \xrightarrow{\tau} Z$ and then $Q \mid R \Longrightarrow U \mid R \xrightarrow{\tau} Z \Longrightarrow O'$ as desired.

- ▷ The dual case of the previous transition, (τ -EXCHANGE), and the two cases of (τ -GET) and (τ -PUT) follow the same pattern outline in the previous cases.

Finally we have seven cases for the higher-order transitions: these need a special treatment because, as we noted, there are no structural rules associated with the higher-order transition. We give the case of ($\text{OUTPUT} \hat{=}$ HO), which is the most complex.

- ▷ $P \mid R \xrightarrow{\langle - \rangle^{\hat{n}[R_1]R_2}} O$, because $P \mid R \xrightarrow{\langle - \rangle^{\hat{}}} K_S \equiv (\mathbf{v}\tilde{s})\langle M \rangle S$, and O is structurally equivalent to $(\mathbf{v}\tilde{s})(n[S \mid R_1] \mid R_2\{x := M\})$. We have two possible sub-cases, depending on whether P or R move. We consider the second case first.

If $R \xrightarrow{\langle - \rangle^{\hat{}}} K_R$, then $K_S \equiv K_R \mid P$, which implies $K_R \equiv (\mathbf{v}\tilde{s})\langle M \rangle R'$ and $S \equiv R' \mid P$. Thus $O \equiv C[P]$ where $C[P] \equiv (\mathbf{v}\tilde{s})(n[R' \mid P \mid R_1] \mid R_2\{x := M\})$. Clearly,

$Q \mid R \xrightarrow{\langle - \rangle^{\hat{n}[R_1]R_2}} C[Q]$. By induction hypothesis $P \mathcal{S} Q$, and since \mathcal{S} is closed by all the operators in the context $C[\cdot]$, we have $C[P] \mathcal{S} C[Q]$ as desired.

If instead P moves, i.e. $P \xrightarrow{\langle - \rangle^{\hat{}}} K_P$, then $K_S \equiv K_P \mid R$, which implies $K_P \equiv (\mathbf{v}\tilde{s})\langle M \rangle P'$ and $S \equiv P' \mid R$. Thus $O \equiv (\mathbf{v}\tilde{s})(n[P' \mid R \mid R_1] \mid R_2\{x := M\})$. Now

from $P \xrightarrow{\langle - \rangle^{\hat{}}} K_P$, by ($\text{OUTPUT} \hat{=}$ HO), we derive $P \xrightarrow{\langle - \rangle^{\hat{n}[R \mid R_1]R_2}} O$. Since $P \mathcal{S} Q$, by the induction hypothesis there exist O' such that $O \mathcal{S} O'$ and a weak

transition of the form: $Q \Longrightarrow U \xrightarrow{\langle - \rangle^{\hat{n}[R \mid R_1]R_2}} Z \Longrightarrow O'$. By an inspection of the transition rules, $Z \equiv (\mathbf{v}\tilde{m})(n[Q' \mid R \mid R_1] \mid R_2\{x := N\})$. Furthermore, the

transition from U must derive from $U \xrightarrow{\langle - \rangle^{\hat{}}} (\mathbf{v}\tilde{m})\langle N \rangle Q'$. Then by rule (PAR)

$U \mid R \xrightarrow{\langle - \rangle^{\hat{}}} (\mathbf{v}\tilde{m})\langle N \rangle Q' \mid R$, from which $U \mid R \xrightarrow{\langle - \rangle^{\hat{n}[R_1]R_2}} Z$. We are done, since $Q \mid R \Longrightarrow U \mid R$ and $Z \Longrightarrow O'$.

- ▷ $n[P] \mathcal{S} n[Q]$ because $P \mathcal{S} Q$. There are again several sub-cases to consider, one for each possible transition. The first, and simplest, case is when $\lambda = \tau$, and the transition $n[P] \xrightarrow{\tau} O$ derives by (AMB). Then, O is the process $n[P']$ and the transition is derived by $P \xrightarrow{\tau} P'$. From the hypothesis $P \mathcal{S} Q$, we know that $Q \Longrightarrow Q'$ with $P' \mathcal{S} Q'$. Then the claim follows by the assumption that \mathcal{S} is closed by the ambient constructor. The remaining cases are as follows.

- ▷ $n[P] \xrightarrow{n \text{ get } M} O$ because $P \xrightarrow{(M)^{\hat{}}} P'$ and $O \equiv n[P']$. Since $P \mathcal{S} Q$, by the induction hypothesis, we know that $Q \xrightarrow{(M)^{\hat{}}} Q'$ with $P' \mathcal{S} Q'$. From this, we have $n[Q] \xrightarrow{n \text{ get } M} n[Q'] \equiv O'$, and $O \mathcal{S} O'$ because \mathcal{S} is closed by the ambient constructor.

- ▷ $n[P] \xrightarrow{\text{exit}(m,k)RS} O$ because $n[P] \xrightarrow{\text{exit}(m,k)} (\mathbf{v})\langle n[P'] \rangle \mathbf{0}$, where O is structurally equivalent to $n[P'] \mid m[R] \mid S\{x := n\}$. The latter transition must have been derived from $P \xrightarrow{\text{out}(m,k)} P'$. Since $P \mathcal{S} Q$, by the induction hypothesis there

exists Q' such that it follows by induction that $Q \Longrightarrow \xrightarrow{\text{out}\langle m,k \rangle} Q'$ and $P' \mathcal{S} Q'$. Then

$$n[Q] \xrightarrow{\text{exit}\langle m,k \rangle RS} n[Z] \mid m[R] \mid S\{x := n\}$$

That $O \mathcal{S} O'$ follows again from $P' \mathcal{S} Q'$ and from \mathcal{S} being closed under parallel composition and ambient construction.

- ▷ $n[P] \xrightarrow{\text{pop}\langle k \rangle R} O$ because $n[P] \xrightarrow{\text{pop}\langle k \rangle} (\nu \tilde{p})\langle m \rangle (m[P_1] \mid n[P_2])$, with O structurally equivalent to $(\nu \tilde{p})\langle m[P_1] \mid n[P_2] \mid R\{x := m\} \rangle$. The latter transition must be derived from $P \xrightarrow{\text{exit}\langle n,k \rangle} (\nu \tilde{p})\langle m[P_1] \rangle P_2$, from which $P \xrightarrow{\text{exit}\langle n,k \rangle \mathbf{0}R} O$. Since $P \mathcal{S} Q$, by induction hypothesis there exists O' s.t. $Q \xrightarrow{\text{exit}\langle n,k \rangle \mathbf{0}R} Z \Longrightarrow O'$ where $Z \equiv (\nu \tilde{q})(l[Q_1] \mid n[Q_2] \mid R\{x := l\})$ and $O \mathcal{S} O'$. By Lemma 3.3(1), we then have the desired $n[Q] \xrightarrow{\text{pop}\langle k \rangle R} (\nu \tilde{q})(l[Q_1] \mid n[Q_2] \mid R\{x := l\}) \Longrightarrow O'$.
- ▷ $n[P] \xrightarrow{n \text{ put } \langle - \rangle R} O$ because $n[P] \xrightarrow{n \text{ put } \langle - \rangle} (\nu \tilde{p})\langle M \rangle n[P']$, where O is structurally equivalent to $(\nu \tilde{p})(n[P'] \mid R\{x := M\})$. The last transition must derive from $P \xrightarrow{\langle - \rangle^\wedge} (\nu \tilde{p})\langle M \rangle P'$, from which $P \xrightarrow{\langle - \rangle^\wedge n[\mathbf{0}]R} (\nu \tilde{p})(n[P'] \mid R\{x := M\})$ derives by an application of (OUTPUT $\hat{=}$ HO). Since $P \mathcal{S} Q$, by induction hypothesis it follows that there exists O' such that $Q \xrightarrow{\langle - \rangle^\wedge n[\mathbf{0}]R} O'$ and $O \mathcal{S} O'$. An inspection of the transition rules shows that O' is of the form $(\nu \tilde{q})(n[Q'] \mid R\{x := N\})$ for suitable Q', R and N . By Lemma 3.3(2), we then have $n[Q] \xrightarrow{n \text{ put } \langle - \rangle R} O'$ as desired.
- ▷ The remaining cases, namely (ENTER HO) and (CO-ENTER HO) similar to and simpler than the previous ones.

- ▷ $(\nu n)P \mathcal{S} (\nu n)Q$ because $P \mathcal{S} Q$. Assume $(\nu n)P \xrightarrow{\lambda} P'$. The move may either derived by (RES), or else by one of the higher-order transitions. In the first case the proof follows directly by the induction hypothesis and the assumption that \mathcal{S} is closed by the restriction operator. For the remaining cases, the proof is by a case analysis of the higher-order transition involved in the move. We give one of these cases below, as representative.

Assume $(\nu n)P \mathcal{S} (\nu n)Q$, and $(\nu n)P \xrightarrow{\text{pop}\langle k \rangle R} O$, and let this transition be derived by (POP HO) from $(\nu n)P \xrightarrow{\text{pop}\langle k \rangle} (\nu n, \tilde{p})\langle m \rangle P'$, with $O \equiv (\nu n, \tilde{p})(P' \mid R\{x := m\})$ and $\{n, \tilde{p}\} \cap \text{fn}(R) = \emptyset$. We need to find a weak transition $(\nu n)Q \xrightarrow{\text{pop}\langle k \rangle R} O'$ with $O \mathcal{S} O'$.

The transition from $(\nu n)P$ must derive by (RES) from $P \xrightarrow{\text{pop}\langle k \rangle} (\nu \tilde{p})\langle m \rangle P'$. From this transition, we have $P \xrightarrow{\text{pop}\langle k \rangle R} P^*$ with $P^* \equiv (\nu \tilde{p})(P' \mid R\{x := m\})$ (and thus $O \equiv$

$(\nu n)P^*$). Since $P \mathcal{S} Q$ we find a weak transition of the form $Q \Longrightarrow V \xrightarrow{\text{pop}\langle k \rangle R} Z \Longrightarrow Q^*$ with $P^* \mathcal{S} Q^*$. By examining the transition $V \xrightarrow{\text{pop}\langle k \rangle R} Z$, we see that it must derive from $V \xrightarrow{\text{pop}\langle k \rangle} (\nu \tilde{q})\langle l \rangle V'$, for $Z \equiv (\nu \tilde{q})(V' \mid R\{x := l\})$ and a suitable l . Now, by (RES), $(\nu n)V \xrightarrow{\text{pop}\langle k \rangle} (\nu n, \tilde{q})\langle l \rangle V'$, and then by (POP HO), $(\nu n)V \xrightarrow{\text{pop}\langle k \rangle R} \equiv (\nu n)Z$. Since $(\nu n)Q \Longrightarrow (\nu n)V$ and $(\nu n)Z \Longrightarrow (\nu n)Q^*$, we have found a weak transition $(\nu n)Q \xrightarrow{\text{pop}\langle k \rangle R} (\nu n)Q^*$. We are done since $(\nu n)Q^* \mathcal{S} (\nu n)P^*$ follows by $P^* \mathcal{S} Q^*$ and the assumption that \mathcal{S} is closed by restriction.

- ▷ $!P \mathcal{S} !Q$ because $P \mathcal{S} Q$. Assume $!P \mathcal{S} !Q$, and let $!P \xrightarrow{\lambda} P'$. The move may either be of form $!P \xrightarrow{\lambda} !P \mid P'$, derived from $P \xrightarrow{\lambda} P'$ by (REPL), or else derived by one of the higher-order transitions. If it is derived by (REPL), give the assumption $P \mathcal{S} Q$, we may use induction to find a move $Q \xrightarrow{\lambda} Q'$ with $P' \mathcal{S} Q'$. Thus $!Q \xrightarrow{\lambda} !Q \mid Q'$ by an application of (REPL). Then we have $!P \mathcal{S} !Q$ and $P' \mathcal{S} Q'$. Since \mathcal{S} is closed by parallel composition, this implies $!P \mid P' \mathcal{S} !Q \mid Q'$, as desired. For the remaining cases, the proof is by a case analysis of the higher-order transition involved in the move. This analysis is similar to that carried out in the previous cases and thus omitted.

□

We conclude with the proof the \approx_c is contained in our relation of barbed congruence. An alternative notion of labelled bisimilarity that completely captures barbed congruence will be discussed in §9.

Theorem 3.5 (Soundness of full bisimilarity). *If $P \approx_c Q$ then $P \cong Q$.*

Proof. It is enough to show that \approx_c is a barbed bisimulation up to \equiv . Assume $P \approx_c Q$. If $P \Downarrow_n$ then, by Lemma 2.4, $P \xrightarrow{n \text{ get } M}$, and we know that $Q \xrightarrow{n \text{ get } M}$, from which $Q \Downarrow_n$. Now assume that $P \longrightarrow P'$. By Theorem 2.3 $P \xrightarrow{\tau} \equiv P'$. Since $P \approx_c Q$, there exists Q' such that $Q \Longrightarrow Q'$ and $P' \equiv \approx_c \equiv Q'$, as desired. □

4 Algebraic Laws

In this section we give some of the characterising algebraic laws for NBA. Some of these laws are inherited from the companion calculi, notably SA(P) and BA, while others are specific to the new calculus, and show the beneficial effects of the new primitives for communication and mobility.

Mobility. The first set of laws are related to mobility and inherited from Safe Ambients (with/out passwords). They show that there are two ways to equate a mobility redex and the

result of reduction: either by relying on secret passwords, or by having the move happen within a protected context (i.e. an ambient).

Theorem 4.1.

1. $(\nu p)(m[\text{in}\langle n, p \rangle.P] \mid n[\overline{\text{in}}(x, p).Q]) \cong (\nu p)(n[Q\{x := m\} \mid m[P]])$
2. $l[m[\text{in}\langle n, p \rangle.P] \mid n[\overline{\text{in}}(x, p).Q]] \cong l[n[Q\{x := m\} \mid m[P]]]$
3. $(\nu p)(n[m[\text{out}\langle n, p \rangle.P] \mid \overline{\text{out}}(x, p).Q]) \cong (\nu p)(m[P] \mid Q\{x := m\})$
4. $l[n[m[\text{in}\langle n, p \rangle.P] \mid \overline{\text{out}}(x, p).Q] \cong l[m[P] \mid Q\{x := m\}].$

Proof. By exhibiting the appropriate bisimulation. In all cases, the bisimulation has the form $\{(LHS, RHS)\} \cup \mathcal{I}$, where LHS and RHS denote, respectively, the left-hand side and the right-hand side of the equation, and \mathcal{I} is the identity. \square

Garbage Collection. The next set of laws provide useful ways to single out inert processes that can be safely garbage collected.

Theorem 4.2. *For any I, J, H finite:*

1. $l[\prod_{i \in I} (\tilde{x}_i)^{n_i}.P_i \mid \prod_{j \in J} (\tilde{x}_j).P_j \mid \prod_{h \in H} \langle \tilde{M}_h \rangle^{m_h}.P_h] \cong \mathbf{0}$
2. $l[\prod_{i \in I} (\tilde{x}_i)^{n_i}.P_i \mid \prod_{j \in J} \langle \tilde{M}_j \rangle.P_j \mid \prod_{h \in H} \langle \tilde{M}_h \rangle^{m_h}.P_h] \cong \mathbf{0}$

Proof. In both cases, the singleton set containing the pair of the two processes is a full bisimulation: this follows by observing that none of the processes in the two laws has any transition. \square

Taking $I = J = H = \emptyset$ in the previous theorem, one also derives $l[] \cong \mathbf{0}$, a very useful equation that allows empty ambients to be garbage collected. This equation holds in Safe Ambients (with/out passwords) as well, while it is not valid for Mobile Ambients, nor for the calculus BA studied in [3]. Notice, in particular, that in NBA the equation is the result of both the presence of co-capabilities *and* of the new semantics of parent-child communication.

Buffer Laws. A further set of laws shows how outputs distribute over the ambient constructor.

Theorem 4.3. *For any finite J :*

1. $l[\prod_{j \in J} \langle \tilde{M}_j \rangle.P_j] \cong \prod_{j \in J} l[\langle \tilde{M}_j \rangle.P_j].$
2. $l[\prod_{j \in J} \langle \tilde{M}_j \rangle^{\hat{}}] \cong \prod_{j \in J} l[\langle \tilde{M}_j \rangle^{\hat{}}].$

Proof. The first equation follows directly by Theorem 4.2(1), as both sides are equivalent to the null process. For (2), we reason by induction on the size of J . For the base case, when $J = \emptyset$, the equation follows by Theorem 4.2(1). For the inductive case, we first show that

$$l[\prod_{j \in J} \langle \tilde{M}_j \rangle^{\hat{}}] \approx_c l[\langle \tilde{M}_k \rangle] \mid l[\prod_{j \in J \setminus \{k\}} \langle \tilde{M}_j \rangle^{\hat{}}] \quad (1)$$

We give a direct proof, showing that the derivatives of the two terms are bisimilar. Assume $l[\Pi_{j \in J} \langle \tilde{M}_j \rangle^\wedge] \xrightarrow{\lambda} P'$. An inspection of the transition rules shows that $\lambda = l \text{ put } \langle - \rangle S$, and that $P' \equiv l[\Pi_{j \in J - \{k\}} \langle \tilde{M}_j \rangle^\wedge] \mid S\{\tilde{x} := \tilde{M}_k\}$, for some process S , and $k \in J$. On the other hand, first observe that $l[\langle \tilde{M}_k \rangle^\wedge] \xrightarrow{l \text{ put } \langle - \rangle} (\nu) \langle \tilde{M}_k \rangle l[]$. Then, an application of the (PAR) rule derives

$$l[\langle \tilde{M}_k \rangle^\wedge] \mid l[\Pi_{j \in J} \langle \tilde{M}_j \rangle^\wedge] \xrightarrow{l \text{ put } \langle - \rangle} (\nu) \langle \tilde{M}_k \rangle (l[] \mid \Pi_{j \in J - \{k\}} l[\langle \tilde{M}_j \rangle^\wedge])$$

Then, by (OUTPUT HO) we derive $l[\langle \tilde{M}_k \rangle^\wedge] \mid l[\Pi_{j \in J} \langle \tilde{M}_j \rangle^\wedge] \xrightarrow{\lambda} l[] \mid P'$, which is what we need, because $l[] \approx_c \mathbf{0}$. The reasoning for the symmetric case is essentially the same.

From (1), by Theorem 3.4, we have $l[\Pi_{j \in J} \langle \tilde{M}_j \rangle^\wedge] \cong l[\langle \tilde{M}_k \rangle] \mid l[\Pi_{j \in J \setminus \{k\}} \langle \tilde{M}_j \rangle^\wedge]$. Now we may use the induction hypothesis and conclude

$$l[\Pi_{j \in J} \langle \tilde{M}_j \rangle^\wedge] \cong l[\langle \tilde{M}_k \rangle] \mid l[\Pi_{j \in J \setminus \{k\}} \langle \tilde{M}_j \rangle^\wedge] \cong l[\langle \tilde{M}_k \rangle] \mid \Pi_{j \in J \setminus \{k\}} l[\langle \tilde{M}_j \rangle^\wedge] \equiv \Pi_{j \in J} l[\langle \tilde{M}_j \rangle^\wedge]$$

as desired. \square

The first equation is a consequence of the semantics of communication of NBA, which makes local communication not observable. This is not true of the semantics of communication in BA. To see that, take $P = l[\langle M_1 \rangle \mid \langle M_2 \rangle]$ and $Q = l[\langle M_1 \rangle] \mid l[\langle M_2 \rangle]$. Then the context $C[\cdot] = [\cdot] \mid n[\text{in} \langle l \rangle . (x)^\dagger . (x)^\dagger . \text{out} \langle l \rangle . \langle \cdot \rangle^\dagger]$ distinguishes them, as $C[P] \Downarrow_n$ while $C[Q] \not\Downarrow_n$, according to the semantics of BA (cf. Introduction, page 3).

The second equation, instead, holds with either semantics. In neither case it generalises to output prefixes with non-null continuation, as in general $n[P_1 \mid P_2] \not\approx n[P_1] \mid n[P_2]$. As a simple example, take $P_1 = () . \overline{\text{in}}(x, n) . \mathbf{0}$ and $P_2 = \langle \cdot \rangle$. Then, $n[P_1] \mid n[P_2] \cong \mathbf{0}$, by Theorem 4.2, while $n[P_1 \mid P_2] \Rightarrow n[\overline{\text{in}}(x, n)]$ which is active and observable.

Communication. The next block of equations gives further insight into the semantics of communication.

Theorem 4.4. *If $|\tilde{x}| = |\tilde{M}|$ then:*

1. $l[(\tilde{x}) . P \mid \langle \tilde{M} \rangle . Q] \cong l[P\{\tilde{x} := \tilde{M}\} \mid Q]$
2. $(\nu l)((\tilde{x})^l . P \mid l[\langle \tilde{M} \rangle^\wedge . Q]) \cong (\nu l)(P\{\tilde{x} := \tilde{M}\} \mid l[Q])$
3. $m[(\tilde{x})^l . P \mid l[\langle \tilde{M} \rangle^\wedge . Q]] \cong m[P\{\tilde{x} := \tilde{M}\} \mid l[Q]]$

The dual laws of 2 and 3 (resulting from exchanging input with output prefixes) hold as well.

Proof. Again, by exhibiting the appropriate bisimulation. In all cases, the bisimulation has the form $\{(LHS, RHS)\} \cup \mathcal{I}$, where LHS and RHS denote the left-hand side and the right-hand side of the equation, respectively. \square

The first equation, 4.4(1) shows again that NBA does not suffer from interferences on local communications: this law holds in Safe Ambients but not in Mobile Ambients, due to open, nor in Boxed Ambients. The remaining equations are distinctive of NBA.

Firewalls. As a further illustration of the algebraic properties of NBA, consider the *perfect firewall* equation from [6]: $(\nu n)n[P] \cong \mathbf{0}$, for $n \notin \text{fn}(P)$. This equation is not valid in NBA, nor in BA. In BA, ambients with secret names may exchange values with their parent. In NBA they can move, and reveal their name. For example, let $P = \text{out}\langle m, m \rangle$, for $m \neq n$, and consider the context $C[\cdot] = (\nu m)(m[[\cdot]] \mid \overline{\text{out}}(x, m).Q)$, where $m \notin \text{fn}(Q)$ and $Q \not\equiv \mathbf{0}$. Then $C[\mathbf{0}] \cong \mathbf{0}$, while $C[(\nu n)n[P]] \longrightarrow \cong (\nu n)(Q\{x := n\} \mid n[P])$.

Indeed, the law $(\nu n)n[P] \cong \mathbf{0}$ ($n \notin \text{fn}(P)$) is not valid in SA or SA(P) either, because the movement of secret ambients is observable in such calculi like in NBA (due to the presence of co-capabilities). In SAP, the equation is re-stated as $(\nu m)(\nu n)m[n[P]] \cong \mathbf{0}$, which holds thanks to the format of the $\overline{\text{out}}$ capability used in [12], which mentions the name of the moving ambient (m in this case). The different syntax for $\overline{\text{out}}$ we adopted in NBA yields yet another variant of the firewall equation.

Theorem 4.5 (Perfect Firewall). $m[n[P]] \cong \mathbf{0}$, for all m and P such that $m \notin \text{fn}(P)$.

Proof. The set $\mathcal{S} = \{ (m[n[P]], \mathbf{0}) \mid m \notin \text{fn}(P) \}$ is a bisimulation. To see that observe that the only visible transitions from $m[n[P]]$ must have a label $\text{pop}\langle k \rangle$ (or its higher-order counterpart) for some k , derived from a transition with label $\text{exit}\langle m, k \rangle$. But this is not possible, if $m \notin \text{fn}(P)$. \square

5 The Type System

We already remarked the effects of revised semantics of communication on the typing system. In this section we elaborate on those ideas, and show that the combination of such semantics with the movement co-capabilities distinctive of NBA can be accounted for at a low complexity cost in the type system, while allowing a degree of flexibility comparable with that of the moded types of [3].

We start our discussion by introducing the structure of types.

<i>Message Types</i>	W	$::=$		$N[E]$	ambient/password
				$C[E]$	capability
<i>Exchange Types</i>	E, F	$::=$		shh	no exchange
				$W_1 \times \dots \times W_k$	tuples ($k \geq 0$)
<i>Process Types</i>	T	$::=$		$[E, F]$	composite exchange

The types of ambients trace the upward exchanges of ambients with this type. In addition, in the present system the types of the form $N[E]$ also serve as the types of passwords: hence, $N[E]$ is indeed the class of *name* types. When used as a password type, $N[E]$ informs on the type E of the upward exchanges of any ambient whose movement is probed by a $N[E]$ password. There is no type confusion in this double role of name types, as different uses of a name have different, and orthogonal, imports in the typing rules. An alternative, perhaps more easily understood solution would be to use two different constructors for ambient and password names: however, this would also have the undesired effect of disallowing the

same name to be used in the two roles, a feature that is harmless, and rather convenient in many examples.

As for capability types, $C[E]$ is the type of capabilities exercised within ambients with upward exchange of type E . Perhaps unexpectedly, tracing the type E is necessary to provide static guarantees of type safety, even with the new semantics of communication. This is due to the dynamic binding of names that takes place upon ambient mobility. On one side, the target context relies on the type of the password presented by the incoming ambients to make assumptions on the upward exchange types of these ambients. Correspondingly, on the side of the moving ambients, the capability types guarantee the consistency between the upward exchanges of that ambient and the type of the passwords used to move.

Exchange and process types also have the same structure as in previous type systems for Ambient Calculi. Type *shh*, however, besides indicating the absence of exchanges, provides here for a *silent* mode for mobility similar to, but substantially simpler than, the *moded types* of [3]. Specifically, the typing rules guarantee that the name of an ambient, say n , crossing a boundary with a password of type $N[\text{shh}]$ will not be used by the receiving environment. Thus, unless the target ambient knows the name n , the use of a $N[\text{shh}]$ password guarantees safe mobility for regardless of the ambients' upward exchanges.

We proceed with the presentation of the typing rules. The rules for valid type environments are standard.

$$\begin{array}{c}
 \text{(ENV EMPTY)} \\
 \hline
 \emptyset \vdash \diamond
 \end{array}
 \qquad
 \begin{array}{c}
 \text{(ENV NAME)} \\
 \Gamma \vdash \diamond \quad a \notin \text{Dom}(\Gamma) \\
 \hline
 \Gamma, a : W \vdash \diamond
 \end{array}$$

Table 7 gives the typing rules for messages. The notation $F \leq G$, with F and G exchange types, is short for $F \in \{\text{shh}, G\}$; operator \sqcup is the (partial) lub operator associated with \leq . Rule (PROJECTION) is standard. Rules (IN) and (OUT) define the types of capabilities in terms of the type of the component passwords: together with the typing rules for the process constructs for ambients in Table 9, they construe the types of passwords as *interfaces* for mobility. In particular, if the type F associated with the password N is a message type W (equivalently, a tuple), then N requires any ambient relying upon N for mobility to have upward exchanges of type W (cf. rules (PREFIX) and (AMB) in Table 9). If, instead, $F = \text{shh}$, then the type G of the upward exchanges can be any type: this is sound, because a move based on an $N[\text{shh}]$ password is guarantee to not reveal the name of the incoming ambient to the target context (cf. rules (CO-IN/OUT-SILENT) in Table 9). Rule (PATH) follows the same intuition: it is applicable only when $E_1 \sqcup E_2$ is defined.

Tables 8 to 10 define the typing of processes. The rules in Table 8 are standard. The rules in Table 9 complement those in Table 7 in governing mobility. Rule (AMB) is standard, and construes the type $N[E]$ as the interface of the ambient M for any process that knows the name M : any such process may have sound E exchanges with M , as the process enclosed within M has upward exchanges of this type. The rules for the mobility co-actions provide similar guarantees for the exchanges a process may have with ambients whose name the process gets to know by exercising the co-capability. In this case, it is the type of the password M that acts as interface: if M has a type $N[\tilde{W}]$ as in rules (CO-IN) and (CO-OUT),

<p>(PROJECTION)</p> $\frac{\Gamma, a : W, \Gamma' \vdash \diamond}{\Gamma, a : W, \Gamma' \vdash a : W}$	<p>(PATH)</p> $\frac{\Gamma \vdash M_1 : C[E_1] \quad \Gamma \vdash M_2 : C[E_2]}{\Gamma \vdash M_1.M_2 : C[E_1 \sqcup E_2]}$
<p>(IN)</p> $\frac{\Gamma \vdash M : N[E] \quad \Gamma \vdash N : N[F] \quad (F \leq G)}{\Gamma \vdash \text{in}\langle M, N \rangle : C[G]}$	<p>(OUT)</p> $\frac{\Gamma \vdash M : N[E] \quad \Gamma \vdash N : N[F] \quad (F \leq G)}{\Gamma \vdash \text{out}\langle M, N \rangle : C[G]}$

Table 7: Good Messages: $\Gamma \vdash M : W$

<p>(PAR)</p> $\frac{\Gamma \vdash P : [E, F] \quad \Gamma \vdash Q : [E, F]}{\Gamma \vdash P \mid Q : [E, F]}$	<p>(REPL)</p> $\frac{\Gamma \vdash P : [E, F]}{\Gamma \vdash !P : [E, F]}$	<p>(DEAD)</p> $\frac{\Gamma \vdash \diamond}{\Gamma \vdash \mathbf{0} : [E, F]}$	<p>(NEW)</p> $\frac{\Gamma, n : N[G] \vdash P : [E, F]}{\Gamma \vdash (\nu n : N[G])P : [E, F]}$
--	--	--	--

Table 8: Good processes I: $\Gamma \vdash P : [E, F]$

we are guaranteed that \tilde{W} is indeed the type of the exchanges of the incoming ambient. If instead the password type is $N[\text{shh}]$, no such guarantee can be made, as easily verified inspecting (PREFIX) and the communication rules in Table 10). Accordingly, rules (CO-IN-SILENT) and (CO-OUT-SILENT) require that the continuation process P makes no use of the variable x and, hence, of the name of the incoming ambient (unless that is already known to P). An alternative, and still sound solution, would be to generalise the (CO-IN) and (CO-OUT) rules by (systematically) replacing the type \tilde{W} with a generic exchange type G . Following this, rules (CO-IN-SILENT) and (CO-OUT-SILENT) could be dispensed with. On the other hand, the resulting system would be less general than the present one, in that any ambient using a silent password for mobility would be required to be upward silent. The current solution, instead, has no such constraint: the typing rules only prevent upward exchanges with the processes enclosed into ambients reached by the use of a silent password. The last set of rules, in Table 10, are those for input output and contain no surprise. In rules for output the judgement $\Gamma \vdash \tilde{M} : \tilde{W}$ stands for the judgements $\Gamma \vdash M_i : W_i$ for $i = 1, \dots, n$ when $\tilde{W} = W_1 \times \dots \times W_n$.

Proposition 5.1 (Subject Reduction). *If $\Gamma \vdash P : T$, and $P \longrightarrow Q$, then $\Gamma \vdash Q : T$.*

Proof. A rather standard proof. The only novelties are the presence of substitutions in the reductions for mobility, and the use of passwords. For the latter, the essence of the proof is in the following observation: if $n[\text{in}\langle m, k \rangle.P_1 \mid P_2]$ (similarly $n[\text{out}\langle m, k \rangle.P_1 \mid P_2]$) is well typed for $n : N[E]$, then $k : N[F]$ for $F \leq E$, and $P_1 \mid P_2 : [G, E]$. Perhaps interestingly, it need not be the case that $F = E$. In particular, it could be that $F = \text{shh}$, in which case the context probing n with k must know the name n , hence its type $N[E]$, to have exchanges with $n[P_1 \mid P_2]$. \square

<p>(AMB)</p> $\frac{\Gamma \vdash M : \mathbb{N}[E] \quad \Gamma \vdash P : [F, E]}{\Gamma \vdash M[P] : [G, H]}$	<p>(PREFIX)</p> $\frac{\Gamma \vdash M : \mathbb{C}[F] \quad \Gamma \vdash P : [E, G] \quad (F \leq G)}{\Gamma \vdash M.P : [E, G]}$
<p>(CO-IN)</p> $\frac{\Gamma \vdash M : \mathbb{N}[\tilde{W}] \quad \Gamma, x : \mathbb{N}[\tilde{W}] \vdash P : [E, F]}{\Gamma \vdash \overline{\text{in}}(x, M).P : [E, F]}$	<p>(CO-OUT)</p> $\frac{\Gamma \vdash M : \mathbb{N}[\tilde{W}] \quad \Gamma, x : \mathbb{N}[\tilde{W}] \vdash P : [E, F]}{\Gamma \vdash \overline{\text{out}}(x, M).P : [E, F]}$
<p>(CO-IN-SILENT)</p> $\frac{\Gamma \vdash M : \mathbb{N}[\text{shh}] \quad \Gamma \vdash P : [E, F] \quad (x \notin \text{fv}(P))}{\Gamma \vdash \overline{\text{in}}(x, M).P : [E, F]}$	<p>(CO-OUT-SILENT)</p> $\frac{\Gamma \vdash M : \mathbb{N}[\text{shh}] \quad \Gamma \vdash P : [E, F] \quad (x \notin \text{fv}(P))}{\Gamma \vdash \overline{\text{out}}(x, M).P : [E, F]}$

Table 9: Good Processes II (mobility)

6 Encoding the π calculus

As a standard test of expressive power for NBA, we give an encoding of the following, choice-free fragment of the synchronous π -calculus [15].

$$P \in \pi ::= \bar{a}(\tilde{b}).P \mid a(\tilde{x}).P \mid P \mid P \mid (\nu a)P$$

There are several choices for the encoding. One solution is obtained directly from the channel encoding of [3] now tailored to the new semantics of communication.

$$\begin{aligned} \llbracket \bar{a}(\tilde{b}).P \rrbracket &= (\nu r) a[\langle \tilde{b}, r \rangle] \mid (r[\langle \cdot \rangle^{\hat{\cdot}}] \mid \langle \cdot \rangle^r. \llbracket P \rrbracket) \quad r \notin \text{fn}(P) \\ \llbracket a(\tilde{x}).Q \rrbracket &= (\tilde{x}, y)^a \langle \cdot \rangle^y. \llbracket Q \rrbracket \quad y \notin \text{fv}(Q) \end{aligned}$$

A different, somewhat more compact encoding, illustrates the power of the binding mechanisms associated with NBA's co-actions. We only show the encoding of channels, the remaining clauses are defined compositionally.

$$\begin{aligned} \llbracket \bar{a}(\tilde{b}).P \rrbracket &\triangleq (\nu p) (a[\langle \tilde{b} \rangle^{\hat{\cdot}}. a[\text{out}(a, p)]] \mid \overline{\text{out}}(-, p). \llbracket P \rrbracket) \quad (p \notin \text{fn}(P)) \\ \llbracket a(\tilde{x}).P \rrbracket &\triangleq (\tilde{x})^a. \llbracket P \rrbracket \end{aligned}$$

Given the direct nature of the encoding, its operational correctness is simple to prove. We do need, however, some preliminary definitions. First, we rely on the commitment semantics of the π calculus given in Table 11. The definition is adapted from [14]: it uses concretions of the form $(\nu \tilde{p})\langle \tilde{q} \rangle P$ with $\{\tilde{p}\} \subseteq \{\tilde{q}\}$, and relies on the same conventions for the notation $(\nu n)O$ and $O \mid Q$ defined in §2 (on page 7).

Then we introduce an *expansion* relation [1] for NBA, which is the standard asymmetric variant of the reduction barbed congruence \cong . The formal definition is as follows, where indicates *one* or more reduction steps .

<p>(INPUT)</p> $\frac{\Gamma, \tilde{x}:\tilde{W} \vdash P : [\tilde{W}, E]}{\Gamma \vdash (\tilde{x}:\tilde{W}).P : [\tilde{W}, E]}$ <p>(INPUT $\hat{}$)</p> $\frac{\Gamma, \tilde{x}:\tilde{W} \vdash P : [E, \tilde{W}]}{\Gamma \vdash (\tilde{x}:\tilde{W})^\wedge.P : [E, \tilde{W}]}$ <p>(INPUT M)</p> $\frac{\Gamma \vdash M : N[\tilde{W}] \quad \Gamma, \tilde{x}:\tilde{W} \vdash P : [G, H]}{\Gamma \vdash (\tilde{x}:\tilde{W})^M.P : [G, H]}$	<p>(OUTPUT)</p> $\frac{\Gamma \vdash \tilde{M} : \tilde{W} \quad \Gamma \vdash P : [\tilde{W}, E]}{\Gamma \vdash \langle \tilde{M} \rangle.P : [\tilde{W}, E]}$ <p>(OUTPUT $\hat{}$)</p> $\frac{\Gamma \vdash \tilde{M} : \tilde{W} \quad \Gamma \vdash P : [E, \tilde{W}]}{\Gamma \vdash \langle \tilde{M} \rangle^\wedge.P : [E, \tilde{W}]}$ <p>(OUTPUT N)</p> $\frac{\Gamma \vdash N : N[\tilde{W}] \quad \Gamma \vdash \tilde{M} : \tilde{W} \quad \Gamma \vdash P : [G, H]}{\Gamma \vdash \langle \tilde{M} \rangle^N.P : [G, H]}$
---	---

Table 10: Good Processes III (input/output)

Definition 6.1 (Expansion [20]). A relation \mathcal{R} is an *expansion* if whenever $P \mathcal{R} Q$,

- i) for each name n , $P \downarrow_n$ implies $Q \downarrow_n$, and $Q \downarrow_n$ implies $P \downarrow_n$.
- ii) $P \longrightarrow P'$ implies $Q \Longrightarrow \longrightarrow Q'$ with $P' \mathcal{R} Q'$
- iii) $Q \longrightarrow Q'$ implies $P \mathcal{R} Q'$ or $P \longrightarrow P'$ with $P' \mathcal{R} Q'$

We note by \lesssim the largest expansion relation preserved by contexts, and say that Q *expands* P if $P \lesssim Q$, that is if $P \mathcal{R} Q$ for some expansion \mathcal{R} . \square

We give a simple, but useful version of one of the algebraic laws given in the previous section, now stated in terms of the expansion relation. We write $Q \gtrsim P$ whenever $P \lesssim Q$.

Lemma 6.2. $(\nu p)(n[m[\text{out}\langle n, p \rangle].P] \mid \overline{\text{out}}(x, p).Q) \gtrsim (\nu p)(m[P] \mid Q\{x := m\})$

Proof. Let LHS and RHS denote the left-hand and right-hand sides, respectively. First observe that $n[\mathbf{0}] \gtrsim \mathbf{0}$. Also, it is easy to see that $LHS \longrightarrow RHS \mid n[\mathbf{0}]$ is the only reduction for LHS . Now assume $LHS \gtrsim RHS$. Clearly, if $RHS \downarrow_l$ then $LHS \downarrow_l$; furthermore, LHS exhibits no barbs, hence the second part of condition (i) holds trivially. For the remaining conditions, if RHS moves, as in $RHS \longrightarrow P$, we have $LHS \longrightarrow RHS \mid n[\mathbf{0}] \longrightarrow P \mid n[\mathbf{0}]$, and $P \mid n[\mathbf{0}] \gtrsim P$ because \gtrsim is closed by parallel composition. If LHS moves, as in $LHS \longrightarrow P$, then $P \equiv RHS \mid n[\mathbf{0}]$, and we know that $RHS \mid n[\mathbf{0}] \gtrsim RHS$. This line of reasoning applies unchanged when we close by contexts, as $C[LHS] \longrightarrow R$ implies that $R \equiv C'[LHS]$, with $C[\mathbf{0}] \longrightarrow C'[\mathbf{0}]$, or $R \equiv C[RHS]$. \square

As an immediate corollary, we have $(\nu p)(n[m[\text{out}\langle n, p \rangle]] \mid \overline{\text{out}}(x, p).Q) \gtrsim (\nu p)Q\{x := m\}$. We will use this latter relation in the proof of the following result.

Lemma 6.3 (Operational Correspondence). Let $P \in \pi$.

- 1. Assume $P \xrightarrow{\alpha} O$. Then the following cases arise:

(INPUT)	(OUTPUT)
$\frac{}{a(\tilde{x}).P \xrightarrow{a(\tilde{b})} P\{\tilde{x} := \tilde{b}\}}$	$\frac{}{\bar{a}(\tilde{b}).P \xrightarrow{\bar{a}} (\nu)(\tilde{b})P}$
(COMM)	
$\frac{P \xrightarrow{\bar{a}} (\nu\tilde{c})(\tilde{b})P' \quad Q \xrightarrow{a(\tilde{b})} Q' \quad \text{fn}(Q) \cap \{\tilde{c}\} = \emptyset}{P \mid Q \xrightarrow{\tau} (\nu\tilde{c})(P' \mid Q')}$	
(RES)	(PAR)
$\frac{P \xrightarrow{\alpha} O \quad a \notin \text{fn}(\alpha)}{(\nu a)P \xrightarrow{\alpha} (\nu a)O}$	$\frac{P \xrightarrow{\alpha} O}{P \mid Q \xrightarrow{\alpha} O \mid Q}$

Table 11: Commitments for the pi-calculus

- (a) $\alpha = a(\tilde{b})$, O is a process and $\langle P \rangle \xrightarrow{(\tilde{b})^a} \gtrsim \langle O \rangle$
- (b) $\alpha = \bar{a}$, $O \equiv (\nu\tilde{c})(\tilde{b})P'$ and $\langle P \rangle \xrightarrow{a \text{ put } \langle - \rangle} (\nu\tilde{c})(\tilde{b})P^*$ with $P^* \gtrsim \langle P' \rangle$
- (c) $\alpha = \tau$, O is a process and $\langle P \rangle \xrightarrow{\tau} \gtrsim \langle O \rangle$

2. Assume $\langle P \rangle \xrightarrow{\alpha} O$. Then the following cases arise:

- (a) $\alpha = (\tilde{b})^a$, O is a process and $\exists P' \in \pi$ such that $P \xrightarrow{a(\tilde{b})} P'$ with $O \gtrsim \langle P' \rangle$.
- (b) $\alpha = a \text{ put } \langle - \rangle$, $O \equiv (\nu\tilde{c})(\tilde{b})P_1$ and $\exists P' \in \pi$ s.t. $P \xrightarrow{\bar{a}} (\nu\tilde{c})(\tilde{b})P'$ and $P_1 \gtrsim \langle P' \rangle$
- (c) $\alpha = \tau$, O is a process, and $\exists P' \in \pi$ such that $P \xrightarrow{\tau} P'$ and $O \gtrsim \langle P' \rangle$

Proof. Part 1 is proved by transition induction. We distinguish the following cases.

- ▷ $P \xrightarrow{\alpha} O$ is $a(\tilde{x}).P_1 \xrightarrow{a(\tilde{b})} P_1\{\tilde{x} := \tilde{b}\}$. By definition, $\langle P \rangle = (x)^a.\langle P_1 \rangle$, and then $\langle P \rangle \xrightarrow{(\tilde{b})^a} \langle P_1 \rangle\{\tilde{x} := \tilde{b}\}$. We are done since $\langle P_1 \rangle\{\tilde{x} := \tilde{b}\} = \langle P_1\{\tilde{x} := \tilde{b}\} \rangle$.
- ▷ $P \xrightarrow{\alpha} O$ is $\bar{a}(\tilde{b}).P_1 \xrightarrow{\bar{a}} (\nu)(\tilde{b})P_1$. By definition,

$$\langle P \rangle = (\nu p)(a[\langle \tilde{b} \rangle^{\hat{}}.a[\text{out}\langle a, p \rangle]] \mid \overline{\text{out}}(x, p).\langle P_1 \rangle) \xrightarrow{a \text{ put } \langle - \rangle} \langle P \rangle(\nu)(\tilde{b})P^*$$

for $p \notin (\text{fn}(P_1) \cup \{\tilde{b}\})$, and $P^* \equiv (\nu p)(a[a[\text{out}\langle a, p \rangle]] \mid \overline{\text{out}}(x, p).\langle P_1 \rangle)$. Since $x \notin \text{fv}(P_1)$, by Lemma 6.2, $P \gtrsim \langle P_1 \rangle$ as desired.

- ▷ $P \xrightarrow{\alpha} O$ is $P_1 \mid P_2 \xrightarrow{\tau} (\text{v}\tilde{c})(P'_1 \mid P'_2)$, derived from $P_1 \xrightarrow{\bar{a}} (\text{v}\tilde{c})\langle\tilde{b}\rangle P'_1$, and from $P_2 \xrightarrow{a(\tilde{b})} P'_2$, with $\text{fn}(P_2) \cap \{\tilde{c}\} = \emptyset$. By induction hypothesis, there exist P_1^* and P_2^* such that $\langle P_1 \rangle \xrightarrow{a \text{ put } -} (\text{v}\tilde{c})\langle\tilde{b}\rangle P_1^*$ and $\langle P_2 \rangle \xrightarrow{(\tilde{b})^a} P_2^*$, with $P_1^* \gtrsim \langle P'_1 \rangle$ and $P_2^* \gtrsim \langle P'_2 \rangle$. An inspection of the translation shows that $\text{fn}(P_2) \cap \{\tilde{c}\} = \emptyset$ implies $\text{fn}(\langle P_2 \rangle) \cap \{\tilde{c}\} = \emptyset$. Then $\langle P_1 \mid P_2 \rangle \xrightarrow{\tau} (\text{v}\tilde{c})(P_1^* \mid P_2^*)$. Since \gtrsim is closed by context, from $P_1^* \gtrsim \langle P'_1 \rangle$ and $P_2^* \gtrsim \langle P'_2 \rangle$ we have $(\text{v}\tilde{c})(P_1^* \mid P_2^*) \gtrsim (\text{v}\tilde{c})(\langle P'_1 \rangle \mid \langle P'_2 \rangle)$. We are done since $(\text{v}\tilde{c})(\langle P'_1 \rangle \mid \langle P'_2 \rangle) = \langle (\text{v}\tilde{c})(P'_1 \mid P'_2) \rangle$.
- ▷ The remaining cases, of the two structural transitions (RES) and (PAR) follow easily by the induction hypothesis and the fact \gtrsim is closed under restriction and parallel composition, respectively.

Part 2 is proved by induction on the structure of P . The case $P = \mathbf{0}$ is immediate.

- ▷ $P = a(\tilde{x}).P_1$. By definition, $\langle P \rangle = (\tilde{x})^a.\langle P_1 \rangle$, thus $\alpha = (\tilde{b})^a$ and $O = \langle P_1 \rangle \{\tilde{x} := \tilde{b}\}$. On the other hand, in π one has $P \xrightarrow{a(\tilde{b})} P_1 \{\tilde{x} := \tilde{b}\}$, and we are done since $\langle P_1 \{\tilde{x} := \tilde{b}\} \rangle = \langle P_1 \rangle \{\tilde{x} := \tilde{b}\}$.
- ▷ $P = \bar{a}\langle\tilde{b}\rangle.P_1$. By definition, $\langle P \rangle = (\text{v}p)(a[\langle\tilde{b}\rangle^\wedge.a[\text{out}\langle a, p \rangle]] \mid \overline{\text{out}}(x, p).\langle P_1 \rangle)$, with $p \notin (\text{fn}(P_1) \cup \{\tilde{b}\})$. Thus $O = (\text{v})\langle\tilde{b}\rangle P_1^*$, derived with $\alpha = a \text{ put } -$, and with $P_1^* \equiv (\text{v}p)(a[a[\text{out}\langle a, p \rangle]] \mid \overline{\text{out}}(x, p).\langle P_1 \rangle)$. On the other hand, in π , $P \xrightarrow{\bar{a}} (\text{v})\langle\tilde{b}\rangle P_1$. Now $P_1^* \gtrsim \langle P_1 \rangle$ follows by Lemma 6.2.
- ▷ $P = P_1 \mid P_2$. By definition $\langle P_1 \mid P_2 \rangle = \langle P_1 \rangle \mid \langle P_2 \rangle$. If $\langle P_1 \rangle \mid \langle P_2 \rangle \xrightarrow{\alpha} O$ derives by (PAR) the proof follows directly by the induction hypothesis. Otherwise, the transition must be of the form $\langle P_1 \rangle \mid \langle P_2 \rangle \xrightarrow{\tau} (\text{v}\tilde{c})(P_1^* \mid P_2^*)$, derived by (COMM) from $\langle P_1 \rangle \xrightarrow{(\tilde{b})^a} P_1^*$ and from $\langle P_2 \rangle \xrightarrow{a \text{ put } -} (\text{v}\tilde{c})\langle\tilde{b}\rangle P_2^*$, for $\text{fn}(\langle P_1 \rangle) \cap \{\tilde{c}\} = \emptyset$. The proof follows now routinely.
- ▷ $P = (\text{v}n)P_1$. This case follows by the induction hypothesis and the fact that \gtrsim is a congruence. \square

Lemma 6.3, extends readily to weak reductions. The proof of the following proposition derives directly from [2] (cf. *loc. cit.*, Proposition 3.6, pg 216).

Proposition 6.4. *Let $P \in \pi$:*

1. *if $P \Longrightarrow P'$ then $\langle P \rangle \Longrightarrow \gtrsim \langle P' \rangle$*
2. *if $\langle P \rangle \Longrightarrow Q$, then there exists P' such that $P \Longrightarrow P'$ and $Q \gtrsim \langle P' \rangle$*
3. *$P \Downarrow_n$ if and only if $\langle P \rangle \Downarrow_n$.*

Proof. Items 1 and 2 are both proved by induction on the number of reduction steps. Item 3 follows from 1 and 2.

1. The base case is trivial. For the inductive case, assume $P \Rightarrow^{n-1} P^* \rightarrow P'$. By induction hypothesis $\langle P \rangle \Rightarrow R \gtrsim \langle P^* \rangle$. From $P^* \rightarrow P'$, by Lemma 6.3(1) we know that $\langle P^* \rangle \rightarrow \gtrsim \langle P' \rangle$. From $R \gtrsim \langle P^* \rangle$ and $\langle P^* \rangle \rightarrow \gtrsim \langle P' \rangle$, we know that $R \Rightarrow \gtrsim \langle P' \rangle$. Thus $\langle P \rangle \Rightarrow R \Rightarrow \gtrsim \langle P' \rangle$ as desired.
2. The base case is again trivial. For the inductive step, assume $\langle P \rangle \Rightarrow Q' \rightarrow Q$. By induction hypothesis there exists $P' \in \pi$ such that $P \Rightarrow P'$ with $Q' \gtrsim \langle P' \rangle$. From this, and from $Q' \rightarrow Q$ we have two possible cases: either $Q \gtrsim \langle P' \rangle$, or $\langle P' \rangle \rightarrow P'' \gtrsim Q$. In the first case we are done. In the second, by Lemma 6.3(2) there is P^* such that $P' \rightarrow P^*$ with $P'' \gtrsim \langle P^* \rangle$. Thus, there is P^* such that $P \Rightarrow P' \rightarrow P^*$ with $Q \gtrsim P'' \gtrsim \langle P^* \rangle$, as desired.
3. From the definition of the encoding and Theorem 2.7, it is verified that $P \downarrow_n$ if and only if $\langle P \rangle \downarrow_n$.

Then, for the (only if) part of the claim, assume $P \Rightarrow P' \downarrow_n$. By (1) we have that $\langle P \rangle \Rightarrow R \gtrsim \langle P' \rangle$. Thus $R \downarrow_n$ and hence also $\langle P \rangle \downarrow_n$.

For the (if) part, assume $\langle P \rangle \Rightarrow Q \downarrow_n$. By (2) there exists P' such that $P \Rightarrow P'$ and $Q \gtrsim \langle P' \rangle$. Thus $\langle P' \rangle \downarrow_n$ which implies $P' \downarrow_n$ and then $P \downarrow_n$. \square

Exploiting this proposition together with the compositionality of $\langle \cdot \rangle$, we can show that the encoding is sound, in the sense below. Let \cong on π terms denote the reduction barbed congruence induced by the following definition of barb: $P \downarrow_n$ just in case $P \equiv (\nu \tilde{p})(\bar{n}\langle - \rangle.Q \mid R)$, for $n \notin \{\tilde{p}\}$.

Theorem 6.5 (Equational Soundness). *If $\langle P \rangle \cong \langle Q \rangle$ in NBA then $P \cong Q$ in π .*

Proof. Let $\mathcal{S} = \{(P, Q) \mid \langle P \rangle \cong \langle Q \rangle\}$: we show that \mathcal{S} is a reduction barbed congruence.

\mathcal{S} is easily shown to be a congruence. By the compositionality of the encoding, given any process P and context $C[\cdot]$, there exists a context D such that $\langle C[P] \rangle = D[\langle P \rangle]$. Let then $P \mathcal{S} Q$, and let $C[\cdot]$ be any context: we need to show that $C[P] \mathcal{S} C[Q]$, that is $\langle C[P] \rangle \cong \langle C[Q] \rangle$. By compositionality, we know that $\langle C[P] \rangle = D[\langle P \rangle]$ and $\langle C[Q] \rangle = D[\langle Q \rangle]$. Then the proof follows directly, because \cong (on NBA terms) is a congruence.

Next, we need to show that \mathcal{S} is barb preserving and reduction closed. Assume $P \mathcal{S} Q$.

- ▷ If $P \downarrow_n$, then by an inspection of the encoding we see that $\langle P \rangle \downarrow_n$, which in turn implies $\langle Q \rangle \downarrow_n$ and hence $Q \downarrow_n$, as desired, by Proposition 6.4(3).
- ▷ Now assume $P \rightarrow P'$. By Lemma 6.3(1) we know that $\langle P \rangle \rightarrow R \gtrsim \langle P' \rangle$. Since $\langle P \rangle \cong \langle Q \rangle$, we find S such that $\langle Q \rangle \Rightarrow S \cong R$. Then, by Proposition 6.4(2), there exists Q' such that $Q \Rightarrow Q'$ and $S \gtrsim \langle Q' \rangle$. Then we have $\langle P' \rangle \lesssim R \cong S \gtrsim \langle Q' \rangle$, thus $\langle P' \rangle \cong \langle Q' \rangle$, that is $P' \mathcal{S} Q'$ as desired.
- ▷ The proofs of the symmetric cases are exactly the same. \square

7 NBA versus BA

In order to relate BA and NBA formally and to characterise the differences between the respective semantics of communication, we present an encoding of BA into an extended version of NBA. Precisely, we enrich NBA with a limited, focused form of nondeterminism that we use in the encoding to circumscribe the communication interferences typical of BA (cf. page 3). This approach has the advantage of localising the gap between the two calculi in a single construct. Formally, we use below a sum operator with a semantics à la CCS, that is $P + Q \longrightarrow R$ if either $P \longrightarrow R$ or $Q \longrightarrow R$.

The encoding is defined parametrically over four names n, mv, pr, pw : n is the name of the ambient (if any) that encloses the process that we are encoding, while the remaining three names are used as passwords. To ease the notation, we use the following shorthands: $\overline{\text{cross}} = !\overline{\text{in}}(x, mv) \mid !\overline{\text{out}}(x, mv)$, $\text{in}\langle n \rangle = \text{in}\langle n, mv \rangle$, and $\text{out}\langle n \rangle = \text{out}\langle n, mv \rangle$. We define two mutually recursive translations, $\langle \cdot \rangle_n$ and $\llbracket \cdot \rrbracket_n$. The interesting cases are below.

$$\begin{aligned}
\langle P \rangle_n &= \overline{\text{cross}} \mid \llbracket P \rrbracket_n \\
\llbracket m[P] \rrbracket_n &= m[\langle P \rangle_n] \\
\llbracket (x)^a.P \rrbracket_n &= (x)^a.\llbracket P \rrbracket_n \\
\llbracket (x).P \rrbracket_n &= (x).\llbracket P \rrbracket_n + (x)^\wedge.\llbracket P \rrbracket_n + \overline{\text{out}}(y, pw).(x)^y.\llbracket P \rrbracket_n & y \notin \text{fn}(P) \\
\llbracket (x)^\dagger.P \rrbracket_n &= (\nu p)p[\text{out}\langle n, pr \rangle.(x)^\wedge.\text{in}\langle n, p \rangle.(x)^\wedge] \mid \overline{\text{in}}(y, p).(x)^y.\llbracket P \rrbracket_n & p, y \notin \text{fn}(P) \\
\llbracket \langle M \rangle^a.P \rrbracket_n &= \langle M \rangle^a.\llbracket P \rrbracket_n \\
\llbracket \langle M \rangle.P \rrbracket_n &= \langle M \rangle.\llbracket P \rrbracket_n + \langle M \rangle^\wedge.\llbracket P \rrbracket_n + \overline{\text{out}}(y, pr).\langle M \rangle^y.\llbracket P \rrbracket_n & y \notin \text{fn}(P) \\
\llbracket \langle M \rangle^\dagger.P \rrbracket_n &= (\nu p)p[\text{out}\langle n, pw \rangle.\langle M \rangle^\wedge.\text{in}\langle n, p \rangle.(x)^\wedge] \mid \overline{\text{in}}(y, p).(x)^y.\llbracket P \rrbracket_n & p, y \notin \text{fn}(P)
\end{aligned}$$

The remaining cases are defined compositionally. The translation $\langle \cdot \rangle_n$ provides unboundedly many co-capabilities, at all nesting levels, so that ambient mobility in BA is rendered faithfully. As for the translation of the communication primitives, the intuition is the following. The upward exchanges of a BA term are dealt with by the taxi ambients that exit the enclosing ambient n to deliver output (or collect input) and then return to n to unlock the continuation P . The use of restricted names as passwords is essential here for the continuation P to be able to identify its helper taxi ambient without risk of confusion. As for the translation of a local input/output, the three branches of the choice reflect the three possible synchronisations: local, from upward, from a nested ambient. Note in particular that the right-most branch of these choices may only match upward requests that encode upward requests from BA terms: this is guaranteed by the use of the two passwords pr and pw that regulate the moves of the read/write taxi ambients. The use of two different passwords ensure that they do not interfere with each other, nor they interfere with other BA ambients' moves (the latter use mv).

Using the algebraic laws in §4 we can show that the encoding is operationally correct (and equationally sound) for *single-threaded* terms. Here, the notion of single-threadedness, although morally identical to SA's, needs to be adapted to NBA to record that engaging

in inter-ambient communications is an activity across ambient boundaries that may create grave interferences. For instance, $a[\langle x \rangle^\wedge \mid \text{out}\langle n, k \rangle.P]$ cannot be considered single-threaded, as illustrated by, say, the context $\overline{\text{out}}(x, k).R \mid n[(x)^a.Q \mid -]$. To ease the presentation, we work with a direct syntactic characterisation of single-threadedness, rather than providing a type system as in [11]. We say that P is single threaded if it does not contain any subprocess of the form $S \mid S$, where S is built according to the following productions:

$$S ::= (\nu \tilde{p}) \pi_1 \dots \pi_k.M.S \mid (\nu \tilde{p}) \pi_1 \dots \pi_k.\langle M \rangle^\wedge.S \mid (\nu \tilde{p}) \pi_1 \dots \pi_k.(x)^\wedge.P \quad (k \geq 0)$$

Theorem 7.1. *If P and Q are single-threaded, then $\langle P \rangle_n \cong \langle Q \rangle_n$ implies $P \cong Q$.*

Proof. Follows the same pattern as the one given for the π calculus. with \cong on BA terms denoting the reduction barbed congruence arising in BA from the following definition of barb: $P \downarrow_n$ just in case $P \equiv (\nu \tilde{m})(n[(\cdot)^\uparrow.Q \mid R] \mid S)$, for $\{n\} \not\subseteq \{\tilde{m}\}$.

The single-threadedness hypothesis on the two source terms P and Q is needed to guarantee the atomicity of the protocol that implements an upward exchange (once the taxi ambient leaves n , we need to make sure that no process inside n causes n to move). \square

Typed Encoding. The encoding extends smoothly to the typed case. The definition is given inductively on the structure of terms, and relative a type environment that records the types of the free names and variables in such terms. The encoding of terms presupposes a corresponding encoding of types, which is indeed the most interesting aspect of the definition.

The structure of types in BA is similar to that of types in NBA, but somewhat more complex. Specifically, BA-ambient types are formed as $\text{amb}[E, F]$, where E is the type of local exchanges, and F the type of the upward exchanges. Capabilities types, in turn, have the form $\text{cap}[E]$, denoting capabilities exercised in ambients with upward exchanges of type E . Finally, process types have exactly the same structure (and interpretation) as the process types of NBA.

The different structure of ambient and capability types in the two calculi reflects the different semantics of communication, and in particular, the fact that in BA the upward exchanges of a migrating ambient may interfere with the local exchanges of the ambients traversed by the ambient on the move. The translation of types is given below:

$$\llbracket \text{amb}[E, F] \rrbracket = N[\llbracket E \rrbracket], \llbracket \text{cap}[E] \rrbracket = C[\text{shh}], \llbracket \text{shh} \rrbracket = \text{shh}, \llbracket [E, F] \rrbracket = [\llbracket E \rrbracket, \llbracket F \rrbracket].$$

Observe that the type traced in $\llbracket \text{amb}[E, F] \rrbracket$ is (the encoding of) the type of the local exchanges: this is because the upward exchanges of in BA are implemented by the helper taxi ambients, whose type will trace the (encoding of) the type F . The local exchanges (again of the source term) are used for typing the upward and local exchanges generated by the translation. The translation of the capability and process types follows the same intuitions, and are direct consequence of the fact that the upward exchanges in the source ambient types are disregarded in the translation (for the reasons we just explained).

The encoding of terms is given in Table 12. The main difference from the untyped case is in the use of a family of passwords pr_W and pw_W , indexed on types, with the implicit

$\langle\!\langle \Gamma \triangleright P \rangle\!\rangle_n$	$= \overline{\text{cross}} \mid \langle\!\langle \Gamma \triangleright P \rangle\!\rangle_n$
$\langle\!\langle \Gamma \triangleright \mathbf{0} \rangle\!\rangle_n$	$= \mathbf{0}$
$\langle\!\langle \Gamma \triangleright M.P \rangle\!\rangle_n$	$= M.\langle\!\langle \Gamma \triangleright P \rangle\!\rangle_n$
$\langle\!\langle \Gamma \triangleright (\nu a : W)P \rangle\!\rangle_n$	$= (\nu a : \langle\!\langle W \rangle\!\rangle) \langle\!\langle \Gamma, a : W \triangleright P \rangle\!\rangle_n$
$\langle\!\langle \Gamma \triangleright P \mid Q \rangle\!\rangle_n$	$= \langle\!\langle \Gamma \triangleright P \rangle\!\rangle_n \mid \langle\!\langle \Gamma \triangleright Q \rangle\!\rangle_n$
$\langle\!\langle \Gamma \triangleright !P \rangle\!\rangle_n$	$= !\langle\!\langle \Gamma \triangleright P \rangle\!\rangle_n$
$\langle\!\langle \Gamma \triangleright m[P] \rangle\!\rangle_n$	$= m[\langle\!\langle \Gamma \triangleright P \rangle\!\rangle_m]$
$\langle\!\langle \Gamma \triangleright (x:W)^\dagger.P \rangle\!\rangle_n$	$= (\nu p : \mathbf{N}[\langle\!\langle W \rangle\!\rangle]) \ p[\text{out}\langle n, \text{pr}_{\langle\!\langle W \rangle\!\rangle} \rangle. (x:\langle\!\langle W \rangle\!\rangle)^\wedge. \text{in}\langle n, p \rangle. \langle x \rangle^\wedge] \mid$ $\quad \overline{\text{in}}(y, p)(x:\langle\!\langle W \rangle\!\rangle)^p. \langle\!\langle \Gamma, x:W \triangleright P \rangle\!\rangle_n$ where $\Gamma(n) = \text{amb}[E, W]$ and $y \notin \text{fn}(P)$
$\langle\!\langle \Gamma \triangleright (x:W)^a.P \rangle\!\rangle_n$	$= (x:\langle\!\langle W \rangle\!\rangle)^a \langle\!\langle \Gamma, x:W \triangleright P \rangle\!\rangle_n \quad \text{where } \Gamma(a) = \text{amb}[W, E]$
$\langle\!\langle \Gamma \triangleright \langle M \rangle^\dagger.P \rangle\!\rangle_n$	$= (\nu p : \mathbf{N}[\langle\!\langle W \rangle\!\rangle]) \ p[\text{out}\langle n, \text{pw}_{\langle\!\langle W \rangle\!\rangle} \rangle. \langle M \rangle^\wedge. \text{in}\langle n, p \rangle. \langle M \rangle^\wedge] \mid$ $\quad \overline{\text{in}}(y, p)(x:\langle\!\langle W \rangle\!\rangle)^p. \langle\!\langle \Gamma, x:W \triangleright P \rangle\!\rangle_n$ where $x, y \notin \text{fn}(P)$ and $\Gamma(n) = \text{amb}[E, W]$
$\langle\!\langle \Gamma \triangleright \langle M \rangle^a.P \rangle\!\rangle_n$	$= \langle M \rangle^a \langle\!\langle \Gamma \triangleright P \rangle\!\rangle_n$
$\langle\!\langle \Gamma \triangleright (x:W)P \rangle\!\rangle_n$	$= (x:\langle\!\langle W \rangle\!\rangle) \langle\!\langle \Gamma, x:W \triangleright P \rangle\!\rangle_n + (x:\langle\!\langle W \rangle\!\rangle)^\wedge \langle\!\langle \Gamma, x:W \triangleright P \rangle\!\rangle_n +$ $\quad + \overline{\text{out}}(y, \text{pw}_{\langle\!\langle W \rangle\!\rangle})(x:\langle\!\langle W \rangle\!\rangle)^y \langle\!\langle \Gamma, x:W \triangleright P \rangle\!\rangle_n$ where $\Gamma(n) = \text{amb}[W, E]$ and $y \notin \text{fn}(P)$
$\langle\!\langle \Gamma \triangleright \langle M \rangle P \rangle\!\rangle_n$	$= \langle M \rangle \langle\!\langle \Gamma \triangleright P \rangle\!\rangle_n + \langle M \rangle^\wedge \langle\!\langle \Gamma \triangleright P \rangle\!\rangle_n + \overline{\text{out}}(y, \text{pr}_{\langle\!\langle W \rangle\!\rangle}) \langle M \rangle^y \langle\!\langle \Gamma \triangleright P \rangle\!\rangle_n$ where $\Gamma(n) = \text{amb}[W, E]$ and $y \notin \text{fn}(P)$

Table 12: Typed Encoding of BA into NBA with guarded choice

assumption that $\text{pr}_W, \text{pw}_W : \mathbf{N}[W]$ for all (NBA) types W . This indexing is required in the typed case, for each of these passwords enables exchanges of the corresponding type. The same would seem needed for the mv password. However, since the co-capabilities that the translation introduces to enable mobility à la BA do not have any continuation, we can safely keep with the sole mv, provided that we stipulate $\text{mv} : \mathbf{N}[\text{shh}]$.

Theorem 7.2 (Soundness of Typing). *If $\Gamma \vdash P : [E, F]$ is derivable in the simple type system for BA (cf. [3], pg.46) and $\Gamma(n) = \mathbf{N}[E, F]$, then $\langle\!\langle \Gamma \rangle\!\rangle \vdash \langle\!\langle \Gamma \triangleright P \rangle\!\rangle_n : \langle\!\langle [E, F] \rangle\!\rangle$ is derivable in NBA.*

Proof. By induction on the derivation of $\Gamma \vdash P : [E, F]$. □

8 Examples

We discuss two further examples that illustrate the power of the new constructs for communication and mobility of NBA in programming non-trivial protocols for distributed systems.

8.1 A point-to-point communication server

Our first example is a system that represents a server for point-to-point communication.

$$w(k) = k[\overline{\text{in}}(x, k). \overline{\text{in}}(y, k). (! (z)^x. \langle z \rangle^y \mid ! (z)^y. \langle z \rangle^x)]$$

Ambient $w(k)$ is a bidirectional forwarder for any pair of incoming ambients. An agent willing to participate in a point-to-point communication must know the password k and should be implemented as the process $A(a, k, P, Q) = a[\text{in}\langle k, k \rangle. P \mid \text{out}\langle k, k \rangle. Q]$, where P performs the expected (upward) exchanges. A complete implementation for the point-to-point server can be then defined as shown below.

$$\text{p2p}(k) = (\nu r) (r[\langle \rangle^{\wedge}] \mid ! ()^r. (w(k) \mid \overline{\text{out}}(-, k). \overline{\text{out}}(-, k). r[\langle \rangle^{\wedge}]))$$

The process $\text{p2p}(k)$ accepts a pair of ambients within the forwarder, provides them with the necessary support of the point-to-point exchange and then lets them out before preparing a new instance of $w(k)$ for a new protocol session. Given the configuration

$$\text{p2p}(k) \mid A(k, a_1, P_1, Q_1) \mid \cdots \mid A(k, a_n, P_n, Q_n),$$

we are guaranteed that at most one pair of agents can be active within k at any given time (k is locked until the two ambients are inside k). In particular, one has:

$$\begin{aligned} & (\nu k) (\text{p2p}(k) \mid A(k, a_1, \langle M \rangle^{\wedge}. P_1, Q_1) \mid A(k, a_2, (x)^{\wedge}. P_2\{x\}, Q_2) \mid \prod_{i \in I} A(K, a_i, R_i, S_i)) \\ & \implies \cong (\nu k) (\text{p2p}(k) \mid a_1[P_1 \mid Q_1] \mid a_2[P_1\{x := M\} \mid Q_2] \mid \prod_{i \in I} A(K, a_i, R_i, S_i)) \end{aligned}$$

This says that once (and if) the two agents have reached the forwarder, no other agent knowing the key k can interfere and prevent them from completing their exchange. The equivalence above follows by the mobility laws of Theorem 4.1 and the laws of Theorem 4.2. In particular, once the two ambients are back at top level, the currently active instance of the forwarder k has the form $k[!(z)^{a_1}. \langle z \rangle^{a_2} \mid ! (z)^{a_2}. \langle z \rangle^{a_1}] \cong \mathbf{0}$.

The use of the forwarder to implement a point-to-point communication protocol may at first appear artificial, for it would seem that two ambients wishing to communicate are likely to know their partner's name, and could then interact via a simpler medium. Indeed, in NBA the example can be simplified with this assumption. In BA, instead, the knowledge of names still leaves a number of problems to be solved, due to possible communication interferences. Consider implementing the protocol without using a forwarder, as shown below.

$$a[\text{in}\langle b \rangle. \text{in}\langle k \rangle. P] \mid b[k[!(x). \langle x \rangle^a \mid ! (x)^a. \langle x \rangle] \mid Q]$$

Process Q can read from/write to k to exchange values with P inside a , but it is not obvious what P should do. With k as given above, P should use local communication to talk with

k (hence with Q): but then, to avoid interference with its own local exchanges, P would need to redirect all the latter to a private ambient. There are similar problems with other possible implementations for k . A first solution is $k[!(x).\langle x \rangle]$: this, however, is problematic because a (or b) may end up re-reading their own messages. A second solution is $k[!(x).\langle x \rangle \mid (x)^\dagger.\langle x \rangle^a]$. Here, the problem is that the upward read by k may mistakenly synchronise with local output in b that was not intended to be for a . The local exchanges in b would again need to be protected from this kind of interference. Similarly for the local exchanges in a .

8.2 A print server

Our next example implements a print server to print jobs arriving off the network in the order of arrival. We give the implementation in steps. First consider the following process that assigns a progressive number to each incoming job. With abuse of notation we use here natural numbers as passwords.

$$\text{enqueue}(k) = (\nu c) (c[\langle 1 \rangle^\wedge] \mid !(n)^c.\overline{\text{in}}(x, k).\langle n \rangle^x.c[\langle n+1 \rangle^\wedge])$$

The (private) ambient c holds the current value of the counter. The process accepts a job and delivers it the current number. Then, it updates the counter and prepares for the next round. This can be turned into a print server mechanism:

$$\begin{aligned} \text{prtsrv}(k) &= k[\text{enqueue}(k) \mid \text{print}] \\ \text{print} &= (\nu c) (c[\langle 1 \rangle^\wedge] \mid !(n)^c.\overline{\text{out}}(x, n).(data)^x.(P\{data\} \mid c[\langle n+1 \rangle^\wedge])) \\ \text{job}(M, k) &= (\nu p)p[\text{in}\langle k, k \rangle.\langle n \rangle^\wedge.(\nu q)q[\text{out}\langle p, n \rangle.\langle M \rangle^\wedge]] \end{aligned}$$

The process $\text{job}(M, k)$ enters the server $\text{prtsrv}(k)$, it is assigned a number to be used as a password for carrying the job M to the printer process P . (Note that the use of passwords is critical here).

This situation appears hard to implement naturally with SA(P) or BA. In SA(P) because one would need to know the names of the incoming jobs to be able to assign them their numbers. In BA because dequeuing the jobs (according to the intended FIFO policy) requires a test of the number a job has been assigned, and an atomic implementation of such test is problematic, if possible at all.

9 A Characterization of Barbed Congruence

We conclude the analysis of NBA by studying an alternative labelled transition system whose associated notion of bisimilarity fully characterises barbed congruence.

We have not found a counter-example to the incompleteness of \approx_c . There is however some indication that this relation might be strictly contained in barbed congruence. The problem is the first-order transitions that enable ambient transitions. To exemplify, consider the case of the input prefix $(x)^\wedge$, and the associated transition $P \xrightarrow{(M)^\wedge} P'$. To show that \approx_c fully characterises \cong , one needs to find a distinguishing context for the label $(M)^\wedge$. This

context is typically defined as $C[\cdot] = m[[\cdot]] \mid \langle M \rangle^m . R$, with R exhibiting some fresh barb so as to probe the label. The problem is that this context tests the continuation P within the ambient n , whereas \approx_c tests P “at top level”. And $n[P] \cong n[Q]$ does not imply that $P \cong Q$, since $n[[\cdot]]$ blocks a number of actions for P and Q that could distinguish them.

A first attempt to solve the problem is to use transition of the form $P \xrightarrow{(M)^\wedge m} m[P']$. These are not quite right, however, because the resulting relation of bisimilarity is not a congruence. To make bisimilarity a congruence, we generalise this idea, and replace the transition $P \xrightarrow{(M)^\wedge} P'$ with the higher-order transition $P \xrightarrow{(M)^\wedge m[R]} m[P' \mid R]$. As we prove in this section, the labelled bisimilarity arising from transitions of this form is indeed closed by context. In addition, it also coincides with barbed congruence.

9.1 A refined labelled transition system

The set of (first-order) labels are defined as in Table 2. We introduce a new class of concretions of the form $\langle \bullet \rangle P$, with P a process, meant to tag our first order transitions. The usual conventions for composition and restrictions apply, namely:

$$\begin{aligned} \triangleright \langle \bullet \rangle P \mid P' &\triangleq \langle \bullet \rangle (P \mid P') \\ \triangleright (\nu \tilde{p}) \langle \bullet \rangle P &\triangleq \langle \bullet \rangle (\nu \tilde{p}) P \end{aligned}$$

Visible transitions. The transitions (OUTPUT), (PUT) and (EXIT) are as in Table 3, and so are the transitions (INPUT) and (CO-CAP) when $\eta \neq \hat{\alpha}$, and when $\pi(x) = \overline{\text{out}}(x, k)$, respectively. The remaining transitions are given below.

<p>(CAP)</p> $\frac{M \in \{\text{in}\langle n, k \rangle, \text{out}\langle n, k \rangle\}}{M.P \xrightarrow{M} \langle \bullet \rangle P}$	<p>(CO-CAP)</p> $\frac{\pi(x) \in \{\overline{\text{in}}(x, k), \overline{\text{out}}(x, k)\}}{\pi(x).P \xrightarrow{\pi(n)} \langle \bullet \rangle P\{x := n\}}$	<p>(PATH)</p> $\frac{M_1.(M_2.P) \xrightarrow{\alpha} O}{(M_1.M_2).P \xrightarrow{\alpha} O}$
<div style="display: flex; justify-content: space-around;"> <div style="text-align: center;"> <p>(INPUT$\hat{\alpha}$)</p> $\frac{}{(x)^\wedge.P \xrightarrow{(M)^\wedge} \langle \bullet \rangle P\{x := M\}}$ </div> <div style="text-align: center;"> <p>(GET)</p> $\frac{P \xrightarrow{(M)^\wedge} \langle \bullet \rangle P'}{m[P] \xrightarrow{m \text{ get } M} m[P']}$ </div> </div>		
<p>(ENTER)</p> $\frac{P \xrightarrow{\text{in}\langle n, k \rangle} \langle \bullet \rangle P'}{m[P] \xrightarrow{\text{enter}\langle n, k \rangle} (\nu) \langle m[P'] \rangle \mathbf{0}}$	<p>(CO-ENTER)</p> $\frac{P \xrightarrow{\overline{\text{in}}(n, k)} \langle \bullet \rangle P'}{m[P] \xrightarrow{m \overline{\text{enter}}(n, k)} (\nu) \langle P' \rangle \mathbf{0}}$	<p>(EXIT)</p> $\frac{P \xrightarrow{\text{out}\langle n, k \rangle} \langle \bullet \rangle P'}{m[P] \xrightarrow{\text{exit}\langle n, k \rangle} (\nu) \langle m[P'] \rangle \mathbf{0}}$

Structural and τ transitions. As in Table 5 and Table 4, respectively.

Higher-order transitions. Those in Table 6, plus the following one:

$$\begin{array}{c}
\text{(PREFIX HO)} \\
\frac{P \xrightarrow{\alpha} \langle \bullet \rangle P' \quad \alpha \in \{(M)^\wedge, \text{cap } \langle n, k \rangle, \overline{\text{in}}(n, k), \overline{\text{out}}(n, k)\}}{P \xrightarrow{\alpha m[R]} m[P' \mid R]}
\end{array}$$

Let now \approx_{fa} denote the labelled bisimulation associated with the new transition system: formally, \approx_{fa} is defined exactly as \approx_c in Definition 3.1. In particular, like \approx_c , also \approx_{fa} tests only transitions from processes to processes.

9.2 Full abstraction

The next two results establish the expected properties of \approx_{fa} , namely that it contains \approx_c , and is closed by contexts.

Theorem 9.1. $\approx_c \subseteq \approx_{fa}$.

Proof. Follows from Theorem 3.5 (on page 21), and from Theorem 9.6, proved later in this section. \square

Theorem 9.2. \approx_{fa} is a congruence.

Proof. Similar to Theorem 3.4. Given the new structure of the transitions for the input prefixes, the inductive proof must be conducted simultaneously on all operators, including input prefixes. We only give the cases that are new or different from those in the proof of Theorem 3.4. Let \mathcal{S} be the least equivalence that contains \approx_{fa} , is closed by substitution and preserved by all operators. We show that \mathcal{S} is a bisimulation (with respect to the new LTS).

▷ $\pi.P \mathcal{S} \pi.Q$ because $P \mathcal{S} Q$. Assume $\pi.P \xrightarrow{\lambda} P'$. When $\pi = M$, with M a capability, $\lambda = Mm[R]$ for suitable m and R , and the transition derives from $M.P \xrightarrow{M} \langle \bullet \rangle P$ with $P' \equiv m[P \mid R]$. But then, by the same reasoning one has $M.Q \xrightarrow{Mm[R]} m[Q \mid R]$ and that $m[P \mid R] \mathcal{S} m[Q \mid R]$ follows by the induction hypothesis (as $P \mathcal{S} Q$ and \mathcal{S} is a congruence).

When $\pi(x) \in \{(x)^\wedge, \overline{\text{in}}(x, k)\}$, $\lambda = \pi(n)m[R]$ and the transition derives from $\pi(x).P \xrightarrow{\pi(n)} \langle \bullet \rangle P\{x := n\}$, with $P' \equiv m[P\{x := n\} \mid R]$. The proof follows by the induction hypothesis, since \mathcal{S} is closed under substitution and preserved by parallel composition and ambient constructor.

▷ $P \mid R \mathcal{S} Q \mid R$ because $P \mathcal{S} Q$. The only new cases are those relative to the transitions (PREFIX HO), whose labels are of the form $\alpha m[R_1]$. We take the case when $\alpha = (M)^\wedge$ as representative. An inspection of the LTS shows that the transition in question must have the form $P \mid R \xrightarrow{(M)^\wedge m[R_1]} m[S \mid R_1]$, derived from $P \mid R \xrightarrow{(M)^\wedge} \langle \bullet \rangle S$. We have two possible sub-cases, depending on whether P or R moved.

The first case is when $S \equiv S_P \mid R$ and $P \xrightarrow{(M)^\wedge} \langle \bullet \rangle S_P$. From this transition, we derive $P \xrightarrow{(M)^\wedge m[R \mid R_1]} m[S_P \mid R \mid R_1]$ by (PREFIX HO). Then, by induction hypothesis, there exists a weak transition $Q \Rightarrow U \xrightarrow{(M)^\wedge m[R \mid R_1]} V \Rightarrow Q'$ with $m[S_P \mid R \mid R_1] \mathcal{S} Q'$. By examining the transition from U we know that there exists Z such that $V \equiv m[Z \mid R \mid R_1]$, and $U \xrightarrow{(M)^\wedge} \langle \bullet \rangle Z$. Then one derives $U \mid R \xrightarrow{(M)^\wedge} \langle \bullet \rangle (Z \mid R)$ by (PAR). Thus we have: $Q \mid R \Rightarrow U \mid R \xrightarrow{(M)^\wedge m[R_1]} m[Z \mid R \mid R_1] \Rightarrow Q'$, as desired.

The other case is when $S \equiv P \mid S_R$, and $R \xrightarrow{(M)^\wedge} \langle \bullet \rangle S_R$. From this transition we derive $Q \mid R \xrightarrow{(M)^\wedge} \langle \bullet \rangle (Q \mid S_R)$ by (PAR). Then by an application of (PREFIX HO), we have $Q \mid R \xrightarrow{(M)^\wedge m[R_1]} m[Q \mid S_R \mid R_1]$. Summarising, for $\lambda = (M)^\wedge m[R_1]$, we have $P \mid R \xrightarrow{\lambda} m[P \mid S_R \mid R_1]$, and we have found a weak transition $Q \mid R \xrightarrow{\lambda} m[Q \mid S_R \mid R_1]$. Then the proof follows from the induction hypothesis and the fact that \mathcal{S} is closed by context.

- ▷ The cases for the remaining constructs, namely ambient, restriction and parallel compositions are proved similarly. \square

Next we show that \approx_{fa} and barbed congruence coincide. We start by defining the following operator of internal choice, as in [12].

$$P \oplus Q = (\nu n)(n[\langle \rangle^\wedge] \mid ()^n.P \mid ()^n.Q) \quad (n \notin \text{fn}(P, Q))$$

Observe that the only possible activity in $P \oplus Q$ is a reduction to either P or Q . Until that choice is made, the process cannot engage in any interaction. We can then define two contexts that allow us to detect whether a generic process performs any action at all.

$$\text{SPY}_{\text{in}}\langle h_1, h_2, \cdot \rangle = (\nu r)(\overline{\text{in}}(x, h_1) \mid r[\langle \rangle^\wedge]) \oplus (\overline{\text{in}}(x, h_2) \mid r[\langle \rangle^\wedge]) \mid ()^r.[\cdot]$$

$$\text{SPY}_{\text{out}}\langle n, h_1, h_2, \cdot \rangle = (\nu r)(\text{out}(n, h_1) \mid r[\langle \rangle^\wedge]) \oplus (\text{out}(n, h_2) \mid r[\langle \rangle^\wedge]) \mid ()^r.[\cdot]$$

The ability to spy comes about when h_1 and h_2 are fresh. Then, a spy context exhibits both of barbs as long as the process plugged inside it has not moved. This is formalised by the following lemmas. With abuse of notation we write $P \downarrow_n$ if $P \xrightarrow{\alpha}$, where α is a (first order) label in Table 2, and $n \in \text{fn}(\alpha)$. Also, we say that a context is *static* if the hole does not appear under a prefix or a replication.

The first lemma characterises those transitions that only involve the spy contexts and do not touch the process that filled the hole.

Lemma 9.3. *Let $C[\cdot]$ be a static context, R a process, n a name, and h_1, h_2 fresh names.*

1. *If $C[\text{SPY}_{\text{in}}\langle h_1, h_2, R \rangle] \xrightarrow{\tau} P$ and $P \downarrow_{h_1, h_2}$, then there exists a static context $C'[\cdot]$ such that $P = C'[\text{SPY}_{\text{in}}\langle h_1, h_2, R \rangle]$, and $C[R] \xrightarrow{\tau} C'[R]$.*

2. If $C[m[\text{SPY}_{\text{out}}\langle n, h_1, h_2, R \rangle]] \xrightarrow{\tau} P$ and $P \Downarrow_{h_1, h_2}$, then there exists a static context $C'[\cdot]$ such that $P = C'[m[\text{SPY}_{\text{out}}\langle n, h_1, h_2, R \rangle]]$, and $C[m[R]] \xrightarrow{\tau} C'[m[R]]$.

Proof. By transition induction. \square

A further lemma allows the spy contexts to be removed.

Lemma 9.4. Let $C_1[\cdot]$ and $C_2[\cdot]$ be static contexts, R_1 and R_2 be (closed) processes, and h_1, h_2 be fresh names. Then

1. $C_1[\text{SPY}_{\text{in}}\langle h_1, h_2, R_1 \rangle] \cong C_2[\text{SPY}_{\text{in}}\langle h_1, h_2, R_2 \rangle]$ implies $C_1[R_1] \cong C_2[R_2]$
2. If $C_1[m[\text{SPY}_{\text{out}}\langle n, h_1, h_2, R_1 \rangle]] \cong C_2[m[\text{SPY}_{\text{out}}\langle n, h_1, h_2, R_2 \rangle]]$ then $C_1[m[R_1]] \cong C_2[m[R_2]]$

Proof. The proof is a generalisation of the corresponding lemma in [12]. For part 1, since \cong is closed under restriction,

$$(\nu h_1, h_2)(C_1[\text{SPY}_{\text{in}}\langle h_1, h_2, R_1 \rangle]) \cong (\nu h_1, h_2)(C_2[\text{SPY}_{\text{in}}\langle h_1, h_2, R_2 \rangle]).$$

Since h_1 and h_2 are fresh and the $C_i[\cdot]$ are static contexts,

$$(\nu h_1, h_2)(C_i[\text{SPY}_{\text{in}}\langle h_1, h_2, R_i \rangle]) \equiv C_i[(\nu h_1, h_2)\text{SPY}_{\text{in}}\langle h_1, h_2, R_i \rangle],$$

for $i \in \{1, 2\}$. Now, one shows by exhibiting the appropriate \approx_{fa} -bisimulation that

$$(\nu h_1, h_2)\text{SPY}_{\text{in}}\langle h_1, h_2, R \rangle \approx_{fa} R,$$

for all R . Since \approx_{fa} implies \cong , we have $C_1[R_1] \cong C_2[R_2]$ as desired. \square

We also need a last simple property.

Lemma 9.5. $P \mid R \cong Q \mid R$ and $\text{fn}(R) \cap \text{fn}(P, Q) = \emptyset$ implies $P \cong Q$.

Proof. Let $\tilde{r} = \text{fn}(R)$ and observe that $(\nu \tilde{r})R \cong \mathbf{0}$. Thus, $P \cong P \mid (\nu \tilde{r})R \equiv (\nu \tilde{r})(P \mid R) \cong (\nu \tilde{r})(Q \mid R) \equiv Q \mid (\nu \tilde{r})R \cong Q$. \square

Theorem 9.6. If $P \cong Q$ then $P \approx_{fa} Q$.

Proof. We show that \cong is a \approx_{fa} -bisimulation up to \equiv . Take $P \cong Q$, and assume $P \xrightarrow{\lambda} P^*$. We need to find a Q^* such that $Q \xrightarrow{\lambda} Q^*$ and $P^* \mathcal{S} Q^*$ (equivalently, $P^* \cong Q^*$). We reason by cases, depending on λ . We will often use the shorthand $h = f[\overline{\text{in}}(x, h)]$, where f will always be assumed fresh.

$\triangleright \lambda = \text{in}\langle n, k \rangle m[R]$. Then the transition in question is $P \xrightarrow{\lambda} \equiv m[P' \mid R]$. Define:

$$C[\cdot] = m[\cdot \mid \text{out}\langle n, h_0 \rangle \mid \text{SPY}_{\text{in}}\langle h_1, h_2, R \rangle] \mid n[\overline{\text{in}}(x, k)] \mid \overline{\text{out}}(-, h_0).(h_3 \oplus h_4)$$

with h_0 – h_4 fresh. We have $C[P] \xrightarrow{\tau} \xrightarrow{\tau} \xrightarrow{\tau} m[P' \mid \text{SPY}_{\text{in}}\langle h_1, h_2, R \rangle] \mid n[] \mid h_3$. Since, $P \cong Q$ we know that $C[Q] \Rightarrow Z \cong m[P' \mid \text{SPY}_{\text{in}}\langle h_1, h_2, R \rangle] \mid h_3$. Therefore, $Z \Downarrow_{h_1, h_2}$ and $Z \Downarrow_{h_4}$. This implies that the transitions from $C[Q]$ have consumed the two co-capabilities. In particular, we have:

$$\begin{aligned}
C[Q] &= m[Q \mid \text{out}\langle n, h_0 \rangle \mid \text{SPY}_{\text{in}}\langle h_1, h_2, R \rangle] \mid n[\overline{\text{in}}(x, k)] \mid \overline{\text{out}}(-, h_0).(h_3 \oplus h_4) \\
&\Rightarrow \xrightarrow{\tau} n[m[Q_1 \mid \text{out}\langle n, h_0 \rangle \mid \text{SPY}_{\text{in}}\langle h_1, h_2, R \rangle]] \mid \overline{\text{out}}(-, h_0).(h_3 \oplus h_4) \\
&\Rightarrow \xrightarrow{\tau} m[Q_2 \mid \text{SPY}_{\text{in}}\langle h_1, h_2, R \rangle] \mid n[] \mid (h_3 \oplus h_4) \\
&\Rightarrow \xrightarrow{\tau} m[Q_3 \mid \text{SPY}_{\text{in}}\langle h_1, h_2, R \rangle] \mid n[] \mid h_3 \\
&\Rightarrow m[Q_4 \mid \text{SPY}_{\text{in}}\langle h_1, h_2, R \rangle] \mid n[] \mid h_3 \\
&= Z \\
&\cong m[Q_4 \mid \text{SPY}_{\text{in}}\langle h_1, h_2, R \rangle] \mid h_3
\end{aligned}$$

Thus, we know that

$$m[P' \mid \text{SPY}_{\text{in}}\langle h_1, h_2, R \rangle] \mid h_3 \cong m[Q_4 \mid \text{SPY}_{\text{in}}\langle h_1, h_2, R \rangle] \mid h_3$$

Since h_3 is fresh by hypothesis, by Lemma 9.5

$$m[P' \mid \text{SPY}_{\text{in}}\langle h_1, h_2, R \rangle] \cong m[Q_4 \mid \text{SPY}_{\text{in}}\langle h_1, h_2, R \rangle]$$

Then, letting $C_1[\cdot] = m[P' \mid \cdot]$, and $C_2[\cdot] = m[Q_4 \mid \cdot]$, by Lemma 9.4,

$$m[P' \mid R] \cong m[Q_4 \mid R]$$

To conclude, we show that $Q \xrightarrow{\text{in}\langle n, k \rangle m[R]} m[Q_4 \mid R]$. To see that, note that the reduction steps in $C[Q] \Rightarrow Z$ above implies that $Q \Rightarrow \xrightarrow{\text{in}\langle n, k \rangle} \langle \bullet \rangle Q_1$ and $Q_1 \Rightarrow Q_4$. Thus, $Q \Rightarrow \xrightarrow{\text{in}\langle n, k \rangle m[R]} m[Q_1 \mid R] \Rightarrow m[Q_4 \mid R]$, as desired.

▷ The other cases of (PREFIX HO) are proved in a similar way, choosing appropriate contexts. In particular,

– when $\lambda = \text{out}\langle n, k \rangle m[R]$, choose

$$C[\cdot] = n[m[\cdot \mid \text{SPY}_{\text{in}}\langle h_1, h_2, R \rangle]] \mid \overline{\text{out}}(x, k).(h_3 \oplus h_4)$$

– when $\lambda = (M)^{\hat{m}}[R]$, choose

$$C[\cdot] = m[\cdot \mid \text{SPY}_{\text{in}}\langle h_1, h_2, R \rangle] \mid \langle M \rangle^m.(h_3 \oplus h_4)$$

– when $\lambda = \overline{\text{in}}(n, k)^{\hat{m}}[R]$ choose

$$C[\cdot] = m[\cdot \mid \text{SPY}_{\text{in}}\langle h_1, h_2, R \rangle] \mid n[\text{in}\langle m, k \rangle.\text{out}\langle n, h_0 \rangle] \mid \overline{\text{out}}(x, h_0).(h_3 \oplus h_4)$$

where the h_i 's are assumed fresh.

- ▷ $\lambda = \text{enter}\langle n, k \rangle R$. The transition in question is $P \xrightarrow{\lambda} (\nu \tilde{p})(n[m[P_1] \mid R\{x := m\}] \mid P_2)$. Let $C_1[\cdot] = (\nu \tilde{p})(n[m[P_1] \mid [\cdot]] \mid P_2)$, and define:

$$C[\cdot] = [\cdot] \mid n[\overline{\text{in}}(x, k).(\text{SPY}_{\text{in}}\langle h_1, h_2, R\{x := m\} \rangle \oplus r[\text{out}\langle n, h_3 \rangle])]$$

with r, h_1-h_3 fresh. We have $C[P] \xrightarrow{\tau} C_1[\text{SPY}_{\text{in}}\langle h_1, h_2, R\{x := m\} \rangle]$. Since $P \cong Q$, there exists a process Z such that $C[Q] \Longrightarrow Z \cong C_1[\text{SPY}_{\text{in}}\langle h_1, h_2, R\{x := m\} \rangle]$. Thus, in particular, $Z \Downarrow_{h_1, h_2}$ and $Z \Downarrow_{h_3}$, which implies that the co-capability $\overline{\text{in}}(x, k)$ must have been consumed in this derivation. Furthermore, by Lemma 9.3, the derivation must have the form:

$$\begin{aligned} C[Q] &= Q \mid n[\overline{\text{in}}(x, k).(\text{SPY}_{\text{in}}\langle h_1, h_2, R\{x := l\} \rangle \oplus r[\text{out}\langle n, h_3 \rangle])] \\ &\Longrightarrow^{\tau} C'[\text{SPY}_{\text{in}}\langle h_1, h_2, R\{x := l\} \rangle \oplus r[\text{out}\langle n, h_3 \rangle]] \\ &\Longrightarrow^{\tau} C''[\text{SPY}_{\text{in}}\langle h_1, h_2, R\{x := l\} \rangle] \\ &\Longrightarrow C_2[\text{SPY}_{\text{in}}\langle h_1, h_2, R\{x := l\} \rangle] = Z \end{aligned}$$

with $C'[\cdot], C''[\cdot]$ and $C_2[\cdot]$ static contexts. From $C_1[\text{SPY}_{\text{in}}\langle h_1, h_2, R\{x := m\} \rangle] \cong Z$, by Lemma 9.4, we know that $C_1[R\{x := m\}] \cong C_2[R\{x := l\}]$. To conclude, it remains to show that $Q \xrightarrow{\lambda} C_2[R\{x := l\}]$. Examining the above sequence of reductions from $C[Q]$ we see that $Q \Longrightarrow \xrightarrow{\text{enter}\langle n, k \rangle R} C'[R]$. Similarly, it is easily verified that $C'[\text{SPY}_{\text{in}}\langle h_1, h_2, R\{x := l\} \rangle] \Longrightarrow C_2[\text{SPY}_{\text{in}}\langle h_1, h_2, R\{x := l\} \rangle]$. Then, by Lemma 9.3, we know that $C'[R\{x := l\}] \Longrightarrow C_2[R\{x := l\}]$, as desired.

- ▷ The remaining cases are similar. Only they require an appropriate choice of the context $C[\cdot]$. In particular

- when $\lambda = m\overline{\text{enter}}\langle n, k \rangle R$, choose

$$C[\cdot] = [\cdot] \mid n[\text{in}\langle m, k \rangle.(\text{SPY}_{\text{out}}\langle n, h_1, h_2, R \rangle \oplus \text{out}\langle n, h_3 \rangle)]$$

- when $\lambda = \text{exit}\langle n, k \rangle RS$, choose

$$C[\cdot] = n[\cdot \mid \text{SPY}_{\text{in}}\langle h_1, h_2, R \rangle] \mid \overline{\text{out}}(x, k).(\text{SPY}_{\text{in}}\langle h_3, h_4, S \rangle \mid (h_5 \oplus h_6))$$

This case requires extending Lemmas 9.3 and 9.4 and to contexts with two holes. There is no difficulty in this extension, as the hypotheses of the lemma imply that the processes enclosed in the spy cages do not move, hence they do not interact.

- when $\lambda = \langle - \rangle^{\hat{n}}[R]S$, choose

$$C[\cdot] = n[\cdot \mid \text{SPY}_{\text{in}}\langle h_1, h_2, R \rangle] \mid (x)^n.(\text{SPY}_{\text{in}}\langle h_3, h_4, S \rangle \mid (h_5 \oplus h_6))$$

This case also requires the extension to Lemmas 9.3 and 9.4 discussed above.

- when $\lambda = \text{pop}\langle k \rangle R$, choose

$$C[\cdot] = [\cdot] \mid \overline{\text{out}}(x, k).(\text{SPY}_{\text{in}}\langle h_1, h_2, R \rangle \oplus r[\overline{\text{in}}(x, h_3)])$$

- when $\lambda = m \text{ put } \langle - \rangle R$, choose

$$C[\cdot] = [\cdot] \mid (x)^m.(\text{SPY}_{\text{in}}\langle h_1, h_2, R \rangle \oplus r[\overline{\text{in}}(x, h_3)])$$

▷ To conclude, there are only two first-order cases.

- when $\lambda = (M)$, choose $C[\cdot] = [\cdot] \mid \langle M \rangle.(h_1 \oplus h_2)$.
- when $\lambda = (M)^n$, choose $C[\cdot] = [\cdot] \mid n[\langle M \rangle^\wedge.r[\text{out}\langle n, h_1 \rangle]] \mid \overline{\text{out}}(x, h_1).(h_2 \oplus h_3)$.

□

Theorem 9.7. \approx_{fa} and \cong are the same relation

Proof. By Theorem 9.2, reasoning as in the proof of Theorem 3.5 we show that $\approx_{fa} \subseteq \cong$. The opposite inclusion follows by Theorem 9.6. □

10 Conclusions

We have developed new semantic foundations for the calculus of Boxed Ambients. In the original calculus [3] the model of communication bears similarities with MA's model of mobility. Much in the same way as a mobile ambient undergoes the move actions of its siblings and children, a boxed ambient is subject to the access to its local communication space by its parent and children. These similarities are also reflected in the complications that this one-sided form of interaction brings into the algebraic theory of the two calculi, in the form of grave interferences.

NBA removes grave interferences by resorting to co-capabilities and by providing each boxed ambient with two distinct channels. A local channel enables the interaction of processes local to the ambient. An upward channel allows communications with the enclosing context. The protocol for value exchange across boundaries is similar in spirit to that of mobility in Safe Ambients, and requires that explicit (mutual) actions be taken by the two parties involved in the interaction. In addition, NBA promotes movement co-capabilities to the role of binding constructs that inform ambients of the incoming ambient's name. Together with a system of password control which verifies the visitor's credentials, this yields an interesting way to learn names dynamically, and provides NBA with essentially the same expressive power as BA.

From the theoretical viewpoint, NBA enjoys a rich algebraic theory, and its barbed congruence admits a fully abstract coinductive characterisation built on a labelled transition semantics. Like companion characterisations in the literature on related calculi [12, 8, 7], our characterisation is rather complex, as it is achieved at the expense labelled transitions which effectively bring back quantification over contexts in terms of the process terms occurring in the higher-order labels.

The benefits of the new semantics of communication are also reflected in the simplicity of the typing system, whose generality again relies on passwords. While some of the typed analyses for Boxed Ambients have been carried out for original model of BA, those results can be re-established in NBA, with no difficulty. This is true not only of the type system developed in this paper, but also of type systems for BA developed by others, notably by Merro and Sassone in [13].

If we look at the expressive power of NBA, and contrast it with MA, the latter is certainly superior. The ability to dissolve boundaries conferred by open provides MA with powerful mechanisms for transferring control, for ambient renaming, and for representing systems with dynamic topology that are not available without open. However, even when disciplined by the control of co-capabilities, the expressive power of open appears to make programming with MA and analysing MA programs more difficult. These difficulties arise principally from open being very general, but also very basic as a programming construct. As we have argued, this makes the encoding of various protocols and systems, whose correctness depends on non-trivial forms of ‘atomicity’, rather complex, and sometimes hardly possible. With NBA this is rectified by resorting to a different, and higher-level set of core primitives that, while not as expressive as their MA counterparts, prove very effective as programming abstractions in the design and specification of such protocols and systems.

References

- [1] S. Arun-Kumar and M. Hennessy. An Efficiency Preorder for Processes. *Acta Informatica*, 29:737–760, 1992.
- [2] M. Boreale. On the Expressiveness of Internal Mobility in Name-Passing Calculi. *Theoretical Computer Science*, 195:205–226, 1998.
- [3] M. Bugliesi, G. Castagna, and S. Crafa. Boxed Ambients. In *TACS’01 Proc. of the 4th Int. Conference on Theoretical Aspects of Computer Science*, number 2215 in Lecture Notes in Computer Science, pages 38–63. Springer-Verlag, 2001.
- [4] M. Bugliesi, S. Crafa, M. Merro, and V. Sassone. Communication interference in mobile boxed ambients. In *FSTTCS’02, Int. Conf. on Foundations of Software Technology and Theoretical Computer Science*, number 2556 in Lecture Notes in Computer Science, pages 71–84. Springer-Verlag, 2002.
- [5] L. Cardelli and A. Gordon. Mobile Ambients. In *Proceedings of FOSSaCS’98*, number 1378 in Lecture Notes in Computer Science, pages 140–155. Springer, 1998.
- [6] L. Cardelli and A. Gordon. Equational Properties for Mobile Ambients. In *Proceedings FoSSaCS’99*, Lecture Notes in Computer Science. Springer-Verlag, 1999.
- [7] G. Castagna, J. Vitek, and F. Zappa Nardelli. The Seal Calculus. Available from <http://www.di.ens.fr/~castagna>, 2003.

- [8] G. Castagna and F. Zappa Nardelli. The Seal Calculus revisited: contextual equivalence and bisimilarity. In *FST&TCS '02, 22th Conference on the Foundations of Software Technology and Theoretical Computer Science*, number 2556 in Lecture Notes in Computer Science, pages 85–96. Springer-Verlag, 2002.
- [9] S. Crafa, M. Bugliesi, and G. Castagna. Information Flow Security for Boxed Ambients. In *F-WAN: Int. Workshop on Foundations of Wide Area Network Computing*, number 66.3 in ENTCS. Elsevier, 2002.
- [10] K. Honda and N. Yoshida. On Reduction-based Process Semantics. *Theoretical Computer Science*, 152(2):437–486, 1995.
- [11] F. Levi and D. Sangiorgi. Controlling interference in Ambients. In *Proceedings of POPL'00*, pages 352–364. ACM Press, 2000.
- [12] M. Merro and M. Hennessy. Bisimulation Congruences in Safe Ambients. In *POPL'02 Proc. 29th ACM Symposium on Principles of Programming Languages*, pages 71–80. ACM Press, 2002.
- [13] M. Merro and V. Sassone. Typing and Subtyping Mobility in Boxed Ambients. In *Proceedings of Concur'02*, number 2421 in Lecture Notes in Computer Science, pages 304–320. Springer, 2002.
- [14] R. Milner. *Communicating and Mobile Systems: the π -calculus*. Cambridge University Press, 1999.
- [15] R. Milner, J. Parrow, and D. Walker. A Calculus of Mobile Processes, Parts I and II. *Information and Computation*, 100:1–77, September 1992.
- [16] D. Sangiorgi. *Expressing Mobility in Process Algebras: First-Order and Higher-Order Paradigms*. PhD thesis CST–99–93, Department of Computer Science, University of Edinburgh, 1992.
- [17] D. Sangiorgi. Bisimulation for Higher-Order Process Calculi. *Information and Computation*, 131(2):141–178, 1996.
- [18] D. Sangiorgi and R. Milner. The problem of “Weak Bisimulation up to”. In *Proc. of CONCUR'92*, volume 630 of *Lecture Notes in Computer Science*, pages 32–46. Springer-Verlag, 1992.
- [19] D. Sangiorgi and R. Milner. The problem of “Weak Bisimulation up to”. In *Proc. CONCUR '92*, volume 630 of *Lecture Notes in Computer Science*, pages 32–46. Springer-Verlag, 1992.
- [20] D. Sangiorgi and D. Walker. *The π -calculus: a Theory of Mobile Processes*. Cambridge University Press, 2001.
- [21] J. Vitek and G. Castagna. Seal: A framework for Secure Mobile Computations. In *Internet Programming Languages*, 1999.