Available online at www.sciencedirect.com

SCIENCE DIRECT®

Information and Computation 202 (2005) 39–86

Information and Computation

www.elsevier.com/locate/ic

Communication and mobility control in boxed ambients ☆

Michele Bugliesi [a,*], Silvia Crafa [a], Massimo Merro [b], Vladimiro Sassone [c]

[a] *Dipartimento di Informatica, Università Ca' Foscari, Venezia, Italy*
[b] *Dipartimento di Informatica, Università di Verona, Italy*
[c] *Department of Informatics, University of Sussex, UK*

Received 28 April 2003; revised 16 November 2004
Available online 18 August 2005

Abstract

Boxed Ambients (BA) replace Mobile Ambients' *open* capability with communication primitives acting across ambient boundaries. The expressiveness of the new communication model is achieved at the price of communication interferences whose resolution requires synchronisation of activities at multiple, distributed locations. We study a variant of BA aimed at controlling communication as well as mobility interferences. 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 the 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.
© 2005 Elsevier Inc. All rights reserved.

1. 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

☆ This work was supported by EU FET-GC project 'MyThS' IST-2001-32617 and EU FET-GC project 'Mikado' IST-2001-32222.
* Corresponding author.
E-mail address: michele@dsi.unive.it (M. Bugliesi).

0890-5401/$ - see front matter © 2005 Elsevier Inc. All rights reserved.
doi:10.1016/j.ic.2005.06.002

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 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, $\mid$ is parallel composition, and square brackets delimit ambients' contents:

$$\begin{array}{r}n[\,\mathsf{in}\, m.P \mid Q\,] \mid m[\,R\,] \longrightarrow m[\,n[\,P \mid Q\,] \mid R\,]\\ m[\,n[\,\mathsf{out}\, m.P \mid Q\,] \mid R\,] \longrightarrow n[\,P \mid Q\,] \mid m[\,R\,].\end{array}$$

A third capability, open, can be used to dissolve ambient boundaries, as expressed by the reduction $\mathsf{open}\, n.P \mid n[\,Q\,] \longrightarrow P \mid Q$. 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.[1]

These ideas have given rise to an innovative calculus capturing several aspects of current distributed systems, and have posed some new interesting problems. Paper [13] unveiled a set of so-called grave interferences, i.e., situations where the inherent non-determinism of movement goes wild. For instance, in

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

ambient n may move out of the parent k, but also in the sibling m. 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 [13] 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

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

and similarly for out and open. Building on such infrastructure, a type-system enforced notion of *single-threadedness* ensures that ambients may never engage in more than one activity involving interactions across boundaries, hence ruling out grave interferences.

Recently, Merro and Hennessy [14] found it useful to work with a version of SA called SAP, where incoming ambients must be able to present a suitable password to cross ambient boundaries. Paper [14] 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.

[1] Unlike MA, here and throughout the paper, we rely on synchronous output. For a discussion on this choice, see [13,3].

Another source of potential problems is open in its own nature as ambient dissolver. A process exercising such a capability will inherit all the contents of the dissolved ambient, including its capabilities and migration strategies. There is nothing inherently wrong with that and it is from open that MA gain part of their expressiveness in modelling systems with dynamic topology. On the other hand, this very expressive power requires greatest care in the use of open in the design of safe and secure systems. The calculus of Boxed Ambients [3] (BA) was born out of the observation that greater control over system behaviour can be combined with comparable expressive power by resorting to communication across boundaries, as in the Seal calculus [29]. As shown below, in BA it is possible to draw an input from a sub-ambient local channel (viz. $(x)^n$) as well as from the parent's local channel (viz. $(x)^\uparrow$), and dually with the roles of input and output swapped.

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

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

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

These forms of interference are as grave as those that led to the definition of SA: indeed, they raise difficulties for a distributed implementation of BA, as there is a hidden, non-trivial distributed consensus problem to address at each communication.

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 (the *Non-interfering Boxed Ambients*), takes inspiration from [10], and is based on the idea that each ambient comes equipped with two mutually non-interfering channels, respectively, for local and upward communications.

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

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 in the child and travelling with it, and poses no particular implementation challenges.

From the theoretical viewpoint, immediate first consequences of the elimination of unwanted interferences are a set of good, expected algebraic laws for NBA, as illustrated in Section 5, and a simpler and cleaner type system. 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 Section 6.

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 [10]).

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. To regain expressive power we 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 $\overline{\mathsf{in}}(x)$ 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. The resulting synchronisation mechanism for mobility may thus be thought of as (an abstraction of) an access protocol, that requires newly arrived agents to register themselves to be granted access to local resources.

However, notice that a purely binding mechanism such as this would not in itself be able to control access, but only to register it. To provide ambients with a stronger control mechanisms, we add a second component to our (co-)capabilities and write rules as the one below.

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

This enhances our access protocol with a form of validation for 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 credentials that takes place when connecting to a (wireless) LAN using DHCP, or to an ISP using PPP.

Remarkably, our admission mechanism resembles quite closely the notion of password as developed in [14], which thus arises yet again as a natural notion to consider. As a consequence, we benefit from results similar to those in [14]. 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 central role in the type system, where their types trace the type of (the upward exchanges of) incoming ambients, so making it possible to provide static guarantees of type safety.

As the paper will show, besides having practical, implementation-oriented features and enjoying good theoretical properties, at the same time NBA remains expressive enough. In particular, by means of examples and encodings in Section 8 we show that the expressive power we loose with respect to BA is, as expected and planned, essentially that directly related to communication interferences.

1.1. Structure of the paper

Section 2 introduces the calculus, presents the reduction semantics and the associated notion of behavioural equivalence. Sections 3 and 4 develop an alternative semantics based on an LTS, and prove that the resulting labelled bisimilarity is sound with respect to the reduction barbed congruence of Section 2. In Section 5, we use labelled bisimilarity to prove a number of algebraic laws for the calculus. The type system of NBA is illustrated and discussed in Section 6, while Sections 7–9 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. Section 10 shows an alternative LTS, whose associated bisimilarity fully characterises barbed congruence at the price of introducing additional higher order labels. Finally, Section 11 concludes the presentation.

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

2. The untyped 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 $a, b, \ldots, m, n, \ldots, q, r$ 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. In addition, they are matched by co-actions $\overline{\mathsf{in}}(x, N)$ and $\overline{\mathsf{out}}(x, N)$ built using 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 $(\tilde{x}).P$ is a binder for the tuple of *variables* $\tilde{x}$, and so are the two co-actions $\overline{\mathsf{in}}(x, M).P$ and $\overline{\mathsf{out}}(x, M).P$ for the variable x; the restriction operator $(\nu n)P$ 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 $\mathrm{fn}(P)$ and $\mathrm{fv}(P)$, respectively, arise as expected, and so does the definition of *capture free* substitution $P\{\tilde{x} := \tilde{M}\}$. We sometimes use the notation $\mathrm{fn}(P, Q)$ as a shorthand for $\mathrm{fn}(P) \cup \mathrm{fn}(Q)$, and similarly $\mathrm{fv}(P, Q)$. A name (variable) is *fresh* in a term if it is different from any other free name (variable) in that term. A process (or an expression) is *closed* if has no free variables (though it may have free names). A *closing substitution* applied to a process P returns a process P' without free variables.

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 $\langle M_1, \ldots, M_k \rangle^{\eta}$ and $(x_1, \ldots, x_k)$, respectively, and similarly $(\nu \tilde{n})$ for $(\nu n_1) \ldots (\nu n_k)$. We omit trailing inactive processes, writing M for $M.\mathbf{0}$, $\langle \tilde{M} \rangle$ for $\langle \tilde{M} \rangle.\mathbf{0}$, and $n[\,]$ for $n[\mathbf{0}]$. We write $(\_).P$, $\overline{\mathsf{in}}(\_, M).P$, and $\overline{\mathsf{out}}(\_, M).P$ when the bound variable only appears in its binding occurrence.

2.1. Reduction semantics

The dynamics of the calculus is given, as usual, in terms of reduction and structural congruence. The definition of structural congruence, noted $\equiv$, is standard [5]: it is the least congruence that satisfies the structural laws in Table 2.

The reduction relation, noted $\longrightarrow$, is defined in Table 1. The *mobility rules* require, as in [14], 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. As in [14] (but unlike [13]) this co-capability is exercised by the target computation space rather than n; in this manner there is a clearer distinction between the role of an ambient in a reduction and the corresponding role of its environment. Finally, unlike [14], our co-out action in rule (Exit) does not mention the name of moving ambient, and so it provides for less control over ambient movement.

Table 1
Syntax and reduction rules

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

mobility

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

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

communication

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

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

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

structural rules

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

(CTX) $P \longrightarrow Q \;\Rightarrow\; C[P] \longrightarrow C[Q]$

The *communication rules* are explained and motivated in Section 1. As usual, in all communication rules we assume that tuples have the same arity, a condition that will be enforced by the type system.

The reduction relation is closed by structural congruence and context, as usual. We define contexts as processes with one hole

$$C[\cdot] ::= [\cdot] \;\mid P|C[\cdot] \mid (\nu n)C[\cdot] \mid !\pi.C[\cdot] \mid n[C[\cdot]] \mid \pi.C[\cdot].$$

Given a process P and a context $C[\cdot]$, $C[P]$ denotes the process that results from $C[\cdot]$ by substituting P for the hole (as usual this substitution may involve name and variable capture). A context in which the whole does not appear under a (replicated) prefix is called a *static*, or evaluation, context. In rule (CTX) of Table 1, the context C is static.

Table 2
Structural laws

$P \mid Q \equiv Q \mid P, \quad P \mid (Q \mid R) \equiv (P \mid Q) \mid R, \quad P \mid \mathbf{0} \ \equiv P$	
$!P \equiv !P \mid P$	
$(\boldsymbol{\nu} n)\mathbf{0} \ \equiv \mathbf{0}$	
$(\boldsymbol{\nu} n)(\boldsymbol{\nu} m)P \equiv (\boldsymbol{\nu} m)(\boldsymbol{\nu} n)P$	$(n \neq m)$
$(\boldsymbol{\nu} n)(P \mid Q) \equiv P \mid (\boldsymbol{\nu} n)Q$	$(n \notin \mathrm{fn}(P))$
$(\boldsymbol{\nu} n)m[P] \equiv m[(\boldsymbol{\nu} n)P]$	$(m \neq n)$
$(M.M').P \equiv M.(M'.P)$	

The reduction relation extends naturally to contexts. We write $\longrightarrow^*$ to denote the reflexive and transitive closure of $\longrightarrow$.

2.2. Behavioural equivalence

Our choice of behavioural semantics for NBA is based on a rather general notion of contextual equality defined in terms of *reduction barbed congruence*, a slight variant of Milner and Sangiorgi's *barbed congruence* [18] (also called *open barbed bisimilarity* [27]). Reduction barbed congruence was first studied by Honda and Yoshida for the π-calculus under the name of maximum sound theory [12]. It is defined in terms of reduction and observability, which appears appropriate to capture the dynamics 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. Let P be a closed process. We write

- $P \downarrow_n$ if $P \equiv (\nu \tilde{m})(n[\overline{\mathsf{in}}(x,k).Q \mid R] \mid S)$ for $\{n,k\} \cap \{\tilde{m}\} = \emptyset$.
- $P \Downarrow_n$ if $P \longrightarrow^* P'$ and $P' \downarrow_n$.

The definition of observations here follows that of [14], and differs from the one we used in [10], reflecting 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, our notion of equality has the extensional property we expect, namely it is independent of the particular choice of the barb (cf. Theorem 10).

Definition 2. A relation $\mathcal{R}$ over closed processes 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 $Q \Downarrow_n$.

Both previous definitions are given for closed processes (that they are well-defined follows as closed processes reduce to closed processes). Next, we introduce our notion of observational congruence, over general processes. Say that a relation $\mathcal{R}$ is contextual if $P\mathcal{R}Q$ implies $C[P]\mathcal{R}C[Q]$ for all contexts $C[\cdot]$.

Table 3
Labels, concretions, and outcomes

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

Definition 3 (*Reduction Barbed Congruence*). Reduction barbed congruence, written $\cong$, is the largest equivalence relation that is contextual and, when restricted to closed processes, is reduction closed and barb preserving.

3. Labelled transition semantics

In this section, we 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 [23,26,28].

The definition of labelled bisimilarity builds on the labelled transitions collected in Tables 4–6. To ease the notation, we present the transitions for the monadic version of the calculus; the generalisation to the polyadic variant of NBA is straightforward. Also, we omit the symmetric variant of the rules in Table 5 and of rule (Par) in Table 6.

The transitions are of the form $P \xrightarrow{\alpha} O$, where O is an "*outcome*." The label α, defined in Table 3, captures the context with which P may interact, as usual. The outcome O is either a process Q, when α is a prefix or the silent action, or a *concretion* of the forms $(\boldsymbol{\nu}\tilde{p})\langle P\rangle Q$ and $(\boldsymbol{\nu}\tilde{p})\langle M\rangle Q$, with P and Q processes, and M an expression. Intuitively, in $(\boldsymbol{\nu}\tilde{p})\langle P\rangle Q$ process P, the *prime*, represents the sub-component of the system that interacts with the environment, while in $(\boldsymbol{\nu}\tilde{p})\langle M\rangle Q$, the expression M represents a piece of information that is transmitted to the environment. In both cases Q represents the remaining components of the process that are not affected by the interaction with the environment, and $\tilde{p}$ is the (possibly empty) set of private names shared by P (or M) and Q. Whenever $\tilde{p}$ is the empty tuple, we write $(\boldsymbol{\nu})\langle P\rangle Q$.

Although our bisimilarity will only be defined in terms of 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 §4, via corresponding higher-order transitions). We use the following conventions.

- if O is the concretion $(\boldsymbol{\nu}\tilde{p})\langle P\rangle Q$, then:
  - ∘ $(\boldsymbol{\nu}r)O = (\boldsymbol{\nu}\tilde{p})\langle P\rangle(\boldsymbol{\nu}r)Q$, if $r \notin \mathrm{fn}(P)$, and $(\boldsymbol{\nu}r)O = (\boldsymbol{\nu}r,\tilde{p})\langle P\rangle Q$ otherwise;
  - ∘ $O \mid R = (\boldsymbol{\nu}\tilde{p})\langle P\rangle(Q \mid R)$,
  - ∘ $R \mid O = (\boldsymbol{\nu}\tilde{p})\langle P\rangle(R \mid Q)$,

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

Table 4
Commitments: visible transitions

(Cap)
$$\frac{M \in \{\mathsf{in}\langle n,k\rangle, \mathsf{out}\langle n,k\rangle\}}{M.P \xrightarrow{M} P}$$

(Co-cap)
$$\frac{\pi(x) \in \{\overline{\mathsf{in}}(x,k), \overline{\mathsf{out}}(x,k)\}}{\pi(x).P \xrightarrow{\pi(n)} P\{x := n\}}$$

(Path)
$$\frac{M_1.(M_2.P) \xrightarrow{\alpha} P'}{(M_1.M_2).P \xrightarrow{\alpha} P'}$$

(Input)
$$\frac{M \text{ closed}}{(x)^{\eta}.P \xrightarrow{(M)^{\eta}} P\{x := M\}}$$

(Output)
$$\frac{}{\langle M\rangle^{\eta}.P \xrightarrow{\langle -\rangle^{\eta}} (\boldsymbol{\nu})\langle M\rangle P}$$

(Get)
$$\frac{P \xrightarrow{(M)^{\hat{}}} P'}{m[P] \xrightarrow{m\ \mathsf{get}\ M} m[P']}$$

(Put)
$$\frac{P \xrightarrow{\langle -\rangle^{\hat{}}} (\boldsymbol{\nu}\tilde{p})\langle M\rangle P' \quad (m \notin \{\tilde{p}\})}{m[P] \xrightarrow{m\ \mathsf{put}\ \langle -\rangle} (\boldsymbol{\nu}\tilde{p})\langle M\rangle m[P']}$$

(Enter)
$$\frac{P \xrightarrow{\mathsf{in}\langle n,k\rangle} P'}{m[P] \xrightarrow{\mathsf{enter}\langle n,k\rangle} (\boldsymbol{\nu})\langle m[P']\rangle \mathbf{0}}$$

(Co-enter)
$$\frac{P \xrightarrow{\overline{\mathsf{in}}(n,k)} P'}{m[P] \xrightarrow{m\ \overline{\mathsf{enter}}(n,k)} (\boldsymbol{\nu})\langle P'\rangle \mathbf{0}}$$

(Exit)
$$\frac{P \xrightarrow{\mathsf{out}\langle n,k\rangle} P'}{m[P] \xrightarrow{\mathsf{exit}\langle n,k\rangle} (\boldsymbol{\nu})\langle m[P']\rangle \mathbf{0}}$$

(Pop)
$$\frac{P \xrightarrow{\mathsf{exit}\langle n,k\rangle} (\boldsymbol{\nu}\tilde{p})\langle m[P_1]\rangle P_2}{n[P] \xrightarrow{\mathsf{pop}\langle k\rangle} (\boldsymbol{\nu}\tilde{p})\langle m\rangle (m[P_1] \mid n[P_2])}$$

- if O is the concretion $(\boldsymbol{\nu}\tilde{p})\langle M\rangle P$, then:
  - $(\boldsymbol{\nu}r)O$ is $(\boldsymbol{\nu}\tilde{p})\langle M\rangle((\boldsymbol{\nu}r)P)$, if $r \notin \mathrm{fn}(M)$, and $(\boldsymbol{\nu}r, \tilde{p})\langle M\rangle P$ otherwise;
  - $O \mid R = (\boldsymbol{\nu}\tilde{p})\langle M\rangle(P \mid R)$,
  - $R \mid O = (\boldsymbol{\nu}\tilde{p})\langle M\rangle(R \mid P)$,

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

The labelled transition system builds on those in [13,14]. The main differences are in the transitions for hierarchical communications, distinctive of NBA, and in the transitions for mobility, as in the

Table 5
Commitments: τ transitions

(τ-ENTER)

$$\frac{P \xrightarrow{\mathsf{enter}\langle m,k\rangle} (\boldsymbol{\nu}\tilde{p})\langle n[P_1]\rangle P_2 \qquad (\mathrm{fn}(P_1) \cup \mathrm{fn}(P_2)) \cap \{\tilde{q}\} = \emptyset \qquad Q \xrightarrow{m\ \overline{\mathsf{enter}}(n,k)} (\boldsymbol{\nu}\tilde{q})\langle Q_1\rangle Q_2 \qquad \mathrm{fn}(Q) \cap \{\tilde{p}\} = \emptyset}{P \mid Q \xrightarrow{\tau} (\boldsymbol{\nu}\tilde{p},\tilde{q})(m[Q_1 \mid n[P_1]] \mid P_2 \mid Q_2)}$$

(τ-EXIT)

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

(τ-EXCHANGE)

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

(τ-PUT)

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

(τ-GET)

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

latter we 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 [14].

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. The only subtlety is that the output actions do not carry any message in the label. Instead, messages are included in the associated concretions. We refer the reader to [14] for a discussion about this choice: as in that case, the format of the output labels is motivated by the fact that the messages output by a process may not in general be observed by any context.

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): notice that the name of the moving ambient is not exposed

Table 6
Commitments: structural transitions

(Par) $$\frac{P \xrightarrow{\alpha} O}{P \mid Q \xrightarrow{\alpha} O \mid Q}$$	(Res) $$\frac{P \xrightarrow{\alpha} O \quad n \notin \mathrm{fn}(\alpha)}{(\nu n)P \xrightarrow{\alpha} (\nu n)O}$$
(τ-Amb) $$\frac{P \xrightarrow{\tau} P'}{n[P] \xrightarrow{\tau} n[P']}$$	(Repl) $$\frac{\pi.P \xrightarrow{\alpha} O}{!\pi.P \xrightarrow{\alpha} !\pi.P \mid O}$$

in the label, but rather recorded in the prime: this allows the (possibly restricted) name of the moving ambient to be extruded when communicated to the accepting context. 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 (Pop) rule completes the move by leaving the name m of the exiting ambient in a buffer. Finally, this name should 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:

- $(\nu\tilde{p})\langle P\rangle Q \equiv (\nu\tilde{p})\langle P'\rangle Q'$ if $P \equiv P'$ and $Q \equiv Q'$
- $(\nu\tilde{p})\langle M\rangle P \equiv (\nu\tilde{p})\langle M\rangle P'$ 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. The second relates labelled transitions and structural congruence. We only give the cases that involve 'in' moves, as the other cases are similar: in particular, it is easily shown that the results of the transitions in the premises of the (Pop) and (τ-Exit) rules have the formats expected by the rules.

Lemma 4.

(1) *If* $P \xrightarrow{\mathsf{in}\langle m,k\rangle} P'$ *then there exist names* $\tilde{p}$, *with* $\{m,k\} \cap \{\tilde{p}\} = \emptyset$, *and processes* P_1, P_2 *such that* $P \equiv (\nu\tilde{p})(\mathsf{in}\langle m,k\rangle.P_1 \mid P_2)$ *and* $P' \equiv (\nu\tilde{p})(P_1 \mid P_2)$.

(2) *If* $P \xrightarrow{\overline{\mathsf{in}}(m,k)} P'$ *then there exist names* $\tilde{p}$, *with* $\{m,k\} \cap \{\tilde{p}\} = \emptyset$, *and processes* P_1, P_2 *such that* $P \equiv (\nu\tilde{p})(\overline{\mathsf{in}}(x,k).P_1 \mid P_2)$ *and* $P' \equiv (\nu\tilde{p})(P_1\{x := m\} \mid P_2)$.

(3) *If* $P \xrightarrow{\mathsf{enter}\langle m,k\rangle} O$ *then there exist names* $\tilde{p}, n$, *with* $\{m,k\} \cap \{\tilde{p}\} = \emptyset$, *and processes* P_1, P_1', P_2 *such that* $P \equiv (\nu\tilde{p})(n[P_1] \mid P_2)$, $P_1 \xrightarrow{\mathsf{in}\langle m,k\rangle} P_1'$, *and* $O \equiv (\nu\tilde{p})\langle n[P_1']\rangle P_2$,

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

Proof. By transition induction. □

Lemma 5. *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 $O \equiv O'$.

The inductive cases are standard. There is 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 5. Note, to this regard, that all the τ-transitions have the side condition $\{\tilde{p}\} \cap \mathrm{fn}(Q) = \emptyset$ (or dually $\{\tilde{q}\} \cap \mathrm{fn}(P) = \emptyset$): this 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-case[2] when $P \equiv Q$ is $(\nu l)(P' \mid Q') \equiv (\nu l)P' \mid Q'$, for $l \notin \mathrm{fn}(Q')$. 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 $l \notin \mathrm{fn}(\alpha)$. Of the many possible cases to analyse, let us focus on the one where α is the silent action and the last transition is derived by (τ-Enter) from $P' \xrightarrow{\mathsf{enter}\langle m,k\rangle} (\nu\tilde{p})\langle n[P_1]\rangle P_2$ and $Q' \xrightarrow{m\ \overline{\mathsf{enter}}(n,k)} (\nu\tilde{q})\langle Q_1\rangle Q_2$, where $\{\tilde{q}\} \cap \mathrm{fn}(P_1, P_2) = \{\tilde{p}\} \cap \mathrm{fn}(Q') = \emptyset$ 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 $(\nu l)P' \mid Q' \xrightarrow{\tau}\equiv (\nu l)O$. To see that, we first observe that $l \neq m,k$, as $l \notin \mathrm{fn}(Q')$ by hypothesis, and $\{m,k\} \subseteq \mathrm{fn}(Q')$ as it can be shown by transition induction. Thus, from $P' \xrightarrow{\mathsf{enter}\langle m,k\rangle} (\nu\tilde{p})\langle n[P_1]\rangle P_2$, we derive $(\nu l)P' \xrightarrow{\mathsf{enter}\langle m,k\rangle} (\nu l)((\nu\tilde{p})\langle n[P_1]\rangle P_2)$ 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 $(\nu l)P' \xrightarrow{\mathsf{enter}\langle m,k\rangle} (\nu l,\tilde{p})\langle n[P_1]\rangle P_2$. 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 $l \notin \mathrm{fn}(Q')$. Note that the proof would *not* go through had we replaced

[2] 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 5, where P' contains the moving ambient whose name is transmitted with the move.

the side condition $\{\tilde{p}\} \cap \mathrm{fn}(Q') = \emptyset$ in rule (τ-ENTER) with $\{\tilde{p}\} \cap \mathrm{fn}(Q_1, Q_2) = \emptyset$ from [13,14]. In particular, the latter condition could be violated by l, as $l \notin \mathrm{fn}(Q')$ does not imply that $l \notin \mathrm{fn}(Q_1, Q_2)$, for l could be n, which may occur free in Q_1.

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

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

Theorem 6.

(1) *If* $P \xrightarrow{\tau} P'$ *then* $P \longrightarrow P'$.
(2) *If* $P \longrightarrow P'$ *then there exists* Q *such that* $P \xrightarrow{\tau} Q$ *and* $Q \equiv P'$.

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 4 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

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

with $\mathrm{fn}(P_1, P_2) \cap \{\tilde{q}\} = \mathrm{fn}(Q) \cap \{\tilde{p}\} = \emptyset$. By (repeated applications of) Lemma 4 there exist $\tilde{r}, \tilde{s}, R_1, R_2, S_1, S_2$ such that

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

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

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

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 5, with O a process, to handle the case when $P \longrightarrow P'$ by (STRUCT). □

As a further illustration between the reduction and the labelled transition semantics, we note that our definition of barb coincides with the choice of one particular action. We write $P \xrightarrow{\alpha}$ to say that $P \xrightarrow{\alpha} P'$ for some P'.

Lemma 7. *$P \downarrow_n$ if and only if $P \xrightarrow{n\ \overline{\mathsf{enter}}(m,k)}$ 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 our definition of equality, based on reduction 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 $\cong$. Below, we use the notation $\Longrightarrow$ to denote $\xrightarrow{\tau}{}^{*}$.

Definition 8. For $\alpha \in \mathit{Labels}$ we write $P \downarrow_\alpha$ if $P \xrightarrow{\alpha}$, and $P \Downarrow_\alpha$ if $P \Longrightarrow\xrightarrow{\alpha}$. Let then $\alpha \in \mathit{Labels} \setminus \{\tau\}$, and define $\cong_\alpha$ 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 $Q \Downarrow_\alpha$.

Proposition 9. *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, $Q' \Downarrow_\alpha$ and we conclude $Q \Longrightarrow Q' \Downarrow_\alpha$. $\square$

Theorem 10. *For all $\alpha \in \mathit{Labels} \setminus \{\tau\}, P \cong Q$ if and only if $P \cong_\alpha Q$.*

Proof. Since the definitions of $\cong$ and $\cong_\alpha$ differ only in the notion of barb, it is enough to show that the two barbs imply each other. As in [14], the use of passwords is fundamental to prove this result.

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

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

  Given any R with ℓ fresh in R, it is easy to show that $R \Downarrow_{n\ \mathsf{put}\ \langle - \rangle}$ if and only if $C[R] \Downarrow_\ell$. This is enough to complete the proof, for $P \downarrow_{n\ \mathsf{put}\ \langle - \rangle}$ implies $C[P] \Downarrow_{n\ \mathsf{put}\ \langle - \rangle}$, and since $P \cong Q$, one has $C[Q] \Downarrow_{n\ \mathsf{put}\ \langle - \rangle}$ which implies $Q \Downarrow_{n\ \mathsf{put}\ \langle - \rangle}$.
  For the reverse implication, let $P \cong_{n\ \mathsf{put}\ \langle - \rangle} Q$, and $P \downarrow_n$. Consider the context defined as follows:

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

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

∘ if $R \Downarrow_n$ then there exists k such that $C^k[R] \Downarrow_{\ell \ \mathsf{put}\ \langle - \rangle}$;
∘ $C^k[R] \Downarrow_{\ell \ \mathsf{put}\ \langle - \rangle}$ implies $R \Downarrow_n$.

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

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

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

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

$$C^k[\cdot] \triangleq [\cdot] \mid \ell[\mathsf{in}\langle n, k \rangle.\mathsf{out}\langle n, h \rangle]$$

with h fresh. For each R with $h \notin \mathrm{fn}(R)$, we have *(i)* $R \Downarrow_n$ implies that $C^k[R] \Downarrow_{\mathsf{pop}\langle h \rangle}$ for a suitable k, and *(ii)* $C^k[R] \Downarrow_{\mathsf{pop}\langle h \rangle}$ implies $R \Downarrow_n$. From this, we conclude as in the previous case.

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

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

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

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

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

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

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

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

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

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

- The other cases are handled similarly. □

4. 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 transitions 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 Tables 7 and 8. 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 $\hat{}$ 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.

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 3 as well as the higher-order labels determined by the transitions in Tables 7 and 8. We denote with λ any label in the set Λ. As usual, we focus on weak transitions, and use the following notation:

Table 7
Commitments I: higher-order transitions

(Output HO)

$$\frac{P \xrightarrow{\langle - \rangle^{\eta}} (\boldsymbol{\nu}\tilde{p})\langle M\rangle P' \quad \mathrm{fv}(Q) \subseteq \{x\},\ \{\tilde{p}\} \cap \mathrm{fn}(Q) = \emptyset,\ \eta \neq \hat{}}{P \xrightarrow{\langle - \rangle^{\eta} Q} (\boldsymbol{\nu}\tilde{p})(P' \mid Q\{x := M\})}$$

(Output $\hat{}$ HO)

$$\frac{P \xrightarrow{\langle - \rangle^{\hat{}}} (\boldsymbol{\nu}\tilde{p})\langle M\rangle P' \quad \begin{array}{l}\mathrm{fv}(R) \subseteq \{x\},\ \mathrm{fv}(Q) = \emptyset \\ \{\tilde{p}\} \cap \mathrm{fn}(n[Q], R) = \emptyset\end{array}}{P \xrightarrow{\langle - \rangle^{\hat{}} n[Q] R} (\boldsymbol{\nu}\tilde{p})(n[P' \mid Q] \mid R\{x := M\})}$$

(Put HO)

$$\frac{P \xrightarrow{m\ \mathsf{put}\ \langle - \rangle} (\boldsymbol{\nu}\tilde{p})\langle M\rangle P' \quad \begin{array}{l}\mathrm{fv}(Q) \subseteq \{x\}, \\ \{\tilde{p}\} \cap \mathrm{fn}(Q) = \emptyset\end{array}}{P \xrightarrow{m\ \mathsf{put}\ \langle - \rangle Q} (\boldsymbol{\nu}\tilde{p})(P' \mid Q\{x := M\})}$$

Table 8
Commitments II: higher-order transitions

(ENTER HO)

$$\frac{P \xrightarrow{\mathsf{enter}\langle n,k\rangle} (\nu\tilde{p})\langle m[P_1]\rangle P_2 \quad \begin{array}{l}\mathrm{fv}(Q) \subseteq \{x\},\\ \{\tilde{p}\} \cap \mathrm{fn}(Q) = \emptyset\end{array}}{P \xrightarrow{\mathsf{enter}\langle n,k\rangle Q} (\nu\tilde{p})(n[m[P_1] \mid Q\{x := m\}] \mid P_2)}$$

(CO-ENTER HO)

$$\frac{P \xrightarrow{m\ \overline{\mathsf{enter}}(n,k)} (\nu\tilde{p})\langle P_1\rangle P_2 \quad \begin{array}{l}\mathrm{fv}(Q) = \emptyset,\\ \{\tilde{p}\} \cap \mathrm{fn}(Q) = \emptyset\end{array}}{P \xrightarrow{m\ \overline{\mathsf{enter}}(n,k)Q} (\nu\tilde{p})(m[n[Q] \mid P_1] \mid P_2)}$$

(EXIT HO)

$$\frac{P \xrightarrow{\mathsf{exit}\langle n,k\rangle} (\nu\tilde{p})\langle m[P_1]\rangle P_2 \quad \begin{array}{l}\mathrm{fv}(R) \subseteq \{x\}, \mathrm{fv}(Q) = \emptyset\\ \{\tilde{p}\} \cap \mathrm{fn}(Q,R) = \emptyset\end{array}}{P \xrightarrow{\mathsf{exit}\langle n,k\rangle QR} (\nu\tilde{p})(m[P_1] \mid n[P_2 \mid Q] \mid R\{x := m\})}$$

(POP HO)

$$\frac{P \xrightarrow{\mathsf{pop}\langle k\rangle} (\nu\tilde{p})\langle m\rangle P' \quad \begin{array}{l}\mathrm{fv}(Q) \subseteq \{x\},\\ \{\tilde{p}\} \cap \mathrm{fn}(Q) = \emptyset\end{array}}{P \xrightarrow{\mathsf{pop}\langle k\rangle Q} (\nu\tilde{p})(P' \mid Q\{x := m\})}$$

- $\overset{\lambda}{\Longrightarrow}$ denotes $\Longrightarrow\overset{\lambda}{\longrightarrow}\Longrightarrow$
- $\overset{\hat{\lambda}}{\Longrightarrow}$ denotes $\Longrightarrow$ if $\lambda = \tau$ and $\overset{\lambda}{\Longrightarrow}$ otherwise.

Definition 11 (*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 $Q \overset{\hat{\lambda}}{\Longrightarrow} Q'$ and $P'\mathcal{R}Q'$. Two processes P and Q are bisimilar, written $P \approx Q$, if $P\mathcal{R}Q$ for some bisimulation $\mathcal{R}$.

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 6 only apply when $\lambda \in$ *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 apply.)

As given, bisimilarity is only defined over closed processes. That the definition is well founded follows by observing that whenever P is a closed process, and $P \xrightarrow{\lambda} P'$, then λ is so that P' is also a closed process. We generalise the definition to arbitrary processes as follows:

Definition 12 (*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.

We end this section showing that full bisimilarity is a congruence. We first need the following lemma.

Lemma 13.

(1) *If* $P \xrightarrow{\mathsf{exit}\langle n,k\rangle \mathbf{0} R} P'$ *then* $n[P] \xrightarrow{\mathsf{pop}\langle k\rangle R} P'$.
(2) *If* $P \xrightarrow{\langle -\rangle \hat{\ } n[\mathbf{0}]R} P'$ *then* $n[P] \xrightarrow{n\ \mathsf{put}\ \langle -\rangle R} P'$.

Proof. By transition induction. □

Theorem 14. $\approx_{\mathrm{c}}$ *is a congruence.*

Proof. The proof that $\approx_{\mathrm{c}}$ is an equivalence relation is standard. It is also easy to show that $\approx_{\mathrm{c}}$ is preserved by input prefixes (these include proper input prefixes and co-capability prefixes). For instance, assuming $P \approx_{\mathrm{c}} Q$, 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 $P \approx_{\mathrm{c}} Q$.

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

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

We show that $\mathscr{S}$ is a bisimulation up to $\equiv$.[3] The theorem follows directly from this fact ($\mathscr{S}$ is itself a bisimulation, hence $\mathscr{S} \subseteq \approx$, which implies $\mathscr{S} = \approx$). The proof is by induction on the formation of $\mathscr{S}$.

- $\underline{P\mathscr{S}Q \text{ because } P \approx Q}$. This case follows by definition.
- $\underline{\pi.P\mathscr{S}\pi.Q \text{ because } P\mathscr{S}Q}$. 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 $P\mathscr{S}Q$ 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.

[3] Briefly, $\mathscr{S}$ is a bisimulation up to $\equiv$ if $P\mathscr{S}Q$ and $P \xrightarrow{\alpha} P'$ implies $Q \overset{\widehat{\alpha}}{\Longrightarrow} Q'$ with $P' \equiv \mathscr{S} \equiv Q'$. The soundness of the up-to $\equiv$ technique is standard (cf. [26]).

- ∘ $\pi.P \xrightarrow{\lambda} P'$ because $\pi = \langle M\rangle^{\eta}$ with $\eta \neq \hat{}$, $\lambda = \langle -\rangle^{\eta}R$ 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 $P\mathscr{S}Q$ by hypothesis, and since $\mathscr{S}$ is closed by parallel composition, we conclude $P \mid R\{x := M\}\mathscr{S}Q \mid R\{x := M\}$, as desired.
- ∘ The case when $\pi = \langle M\rangle^{\hat{}}$ is similar: it also requires the closure of $\mathscr{S}$ by the ambient constructor.

- $\underline{P \mid R\mathscr{S}Q \mid R}$ because $P\mathscr{S}Q$. We proceed by a case analysis of the reduction $P \mid R \xrightarrow{\lambda} O$, with O a process (not a concretion). There are 13 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 $P\mathscr{S}Q$, by induction hypothesis we find a weak transition $Q \overset{\lambda}{\Longrightarrow} Q'$ with $P'\mathscr{S}Q'$. Thus, we also have a weak transition $Q \mid R \overset{\lambda}{\Longrightarrow} Q' \mid R$, and since $\mathscr{S}$ is closed by parallel composition, $P' \mid R\mathscr{S}Q' \mid R$ as desired.

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

  - ∘ $P \mid R \xrightarrow{\tau} O$ derives from

$$P \xrightarrow{\mathsf{enter}\langle m,k\rangle} (\nu\tilde{p})\langle n[P_1]\rangle P_2, \text{and } R \xrightarrow{m\ \overline{\mathsf{enter}}(n,k)} (\nu\tilde{r})\langle R_1\rangle R_2$$

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 $Q \mid R \Longrightarrow O'$ with $O\mathscr{S}O'$. One has $P \xrightarrow{\mathsf{enter}\langle m,k\rangle R_x} P' \equiv (\nu\tilde{p})(m[n[P_1] \mid R_1] \mid P_2)$ by rule (Enter HO). Since $P\mathscr{S}Q$, by induction hypothesis there exists Q' such that $P'\mathscr{S}Q'$, for which $Q \overset{\mathsf{enter}\langle m,k\rangle R_x}{\Longrightarrow} Q'$. Thus, $Q \Longrightarrow V \xrightarrow{\mathsf{enter}\langle m,k\rangle R_x} Z \Longrightarrow Q'$ for appropriate V and Z. An inspection of the transition rules shows that $Z \equiv (\nu\tilde{q})(m[l[Q_1] \mid R_x\{x := l\}] \mid Q_2)$ for suitable names $l, \tilde{q}$ and processes Q_1 and Q_2. Furthermore, the transition $V \xrightarrow{\mathsf{enter}\langle m,k\rangle R_x} Z$ must derive from $V \xrightarrow{\mathsf{enter}\langle m,k\rangle} (\nu\tilde{q})\langle l[Q_1]\rangle Q_2$. From $R \xrightarrow{m\ \overline{\mathsf{enter}}(n,k)} (\nu\tilde{r})\langle R_1\rangle R_2$, we have $R \xrightarrow{m\ \overline{\mathsf{enter}}(l,k)} (\nu\tilde{r})\langle R_x\{x := l\}\rangle R_2$. Hence by an application of the rule (τ Enter), we have $Q \mid R \Longrightarrow V \mid R \xrightarrow{\tau} (\nu\tilde{r})(Z \mid R_2) \Longrightarrow (\nu\tilde{r})(Q' \mid R_2)$. From $P'\mathscr{S}Q'$, since $\mathscr{S}$ is closed by restriction and parallel composition, it follows that $O \equiv (\nu\tilde{r})(P' \mid R_2)\mathscr{S}(\nu\tilde{r})(Q' \mid R_2) \equiv O'$, as desired.
  - ∘ $P \mid R \xrightarrow{\tau} O$ derives from

$$P \xrightarrow{m\ \overline{\mathsf{enter}}(n,k)} (\nu\tilde{p})\langle P_1\rangle P_2, \text{ and } R \xrightarrow{\mathsf{enter}\langle m,k\rangle} (\nu\tilde{r})\langle n[R_1]\rangle R_2$$

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 $Q \mid R \Longrightarrow O'$ with $O\mathscr{S}O'$. By an application of rule (Co-Enter HO) one has $P \xrightarrow{m\ \overline{\mathsf{enter}}(n,k)R_1} P' \equiv \underline{(\nu\tilde{p})}(m[n[R_1] \mid P_1] \mid P_2)$, with $\{\tilde{p}\} \cap \mathrm{fn}(R_1) = \emptyset$. Since $P\mathscr{S}Q$, there exists Q' such that $Q \Longrightarrow V \xrightarrow{m\ \overline{\mathsf{enter}}(n,k)R_1} Z \Longrightarrow Q'$ with $P'\mathscr{S}Q'$. 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 $V \xrightarrow{m\ \mathsf{enter}\langle n,k\rangle R_1} Z$ must have been derived from $V \xrightarrow{m\ \overline{\mathsf{enter}}(n,k)} (\nu\tilde{q})\langle Q_1\rangle Q_2$. Thus, by an application of (τ Enter)

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

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

- $P \mid R \xrightarrow{\tau} O$ because $P \xrightarrow{\mathsf{pop}\langle k\rangle} (\nu\tilde{p})\langle m\rangle P'$ and $R \xrightarrow{\overline{\mathsf{out}}(m,k)} R'$, where O is 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 $P \xrightarrow{\mathsf{pop}\langle k\rangle R_x} O$. Since $P\mathscr{S}Q$, by the induction hypothesis we find a transition $Q \Longrightarrow V \xrightarrow{\mathsf{pop}\langle k\rangle R_x} Z \Longrightarrow O'$ with $O\mathscr{S}O'$. The transition $V \xrightarrow{\mathsf{pop}\langle k\rangle R_x} Z$ must have been derived from $V \xrightarrow{\mathsf{pop}\langle k\rangle} (\nu\tilde{r})\langle l\rangle V'$ for suitable V' and l, with Z of the form $(\nu\tilde{r})(V' \mid R_x\{x := l\})$. Also, from $R \xrightarrow{\overline{\mathsf{out}}(m,k)} R'$, it follows that $R \xrightarrow{\overline{\mathsf{out}}(l,k)} R_x\{x := l\}$, hence $V \mid R \xrightarrow{\tau} Z$. We are done, as $Q \mid R \Longrightarrow V \mid R \xrightarrow{\tau} Z \Longrightarrow O'$.
- $P \mid R \xrightarrow{\tau} O$ because $P \xrightarrow{\overline{\mathsf{out}}(m,k)} P'$ and $R \xrightarrow{\mathsf{pop}\langle k\rangle} (\nu\tilde{r})\langle m\rangle R'$ with O structurally equivalent to $(\nu\tilde{r})(R' \mid P')$. Since $P\mathscr{S}Q$, by induction hypothesis, we know that $Q \Longrightarrow U \xrightarrow{\overline{\mathsf{out}}(m,k)} Z \Longrightarrow Q'$. Thus

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

  Now, $O\mathscr{S}O'$ derives from $P'\mathscr{S}Q'$ because $\mathscr{S}$ 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 $P\mathscr{S}Q$, we have $Q \Longrightarrow U \xrightarrow{\langle -\rangle R_x} Z \Longrightarrow O'$ with $O\mathscr{S}O'$. 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 (Output $\hat{}$ HO), which is the most complex.

- $P \mid R \xrightarrow{\langle - \rangle\hat{\ } n[R_1]R_2} O$, because $P \mid R \xrightarrow{\langle - \rangle\hat{\ }} K_S \equiv (\nu\tilde{s})\langle M\rangle S$, and O is structurally equivalent to $(\nu\tilde{s})(n[S \mid R_1] \mid R_2\{x := M\})$. 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 $K_R \equiv (\nu\tilde{s})\langle M\rangle R'$ and $S \equiv R' \mid P$. Thus $O \equiv C[P]$ where $C[P] \equiv (\nu\tilde{s})(n[R' \mid P \mid R_1] \mid R_2\{x := M\})$. 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 $K_P \equiv (\nu\tilde{s})\langle M\rangle P'$ and $S \equiv P' \mid R$. Thus $O \equiv (\nu\tilde{s})(n[P' \mid R \mid R_1] \mid R_2\{x := M\})$. Now from $P \xrightarrow{\langle - \rangle\hat{\ }} K_P$, we derive $P \xrightarrow{\langle - \rangle\hat{\ } n[R \mid R_1]R_2} O$ by (Output $\hat{\ }$ HO). Since $P\mathcal{S}Q$, by the induction hypothesis there exist O' such that $Q \Longrightarrow U \xrightarrow{\langle - \rangle\hat{\ } n[R \mid R_1]R_2} Z \Longrightarrow O'$ with $O\mathcal{S}O'$. Here $Z \equiv (\nu\tilde{m})(n[Q' \mid R \mid R_1] \mid R_2\{x := N\})$, as it is verified by inspecting the transition rules. Furthermore, the transition from U must derive from $U \xrightarrow{\langle - \rangle} (\nu\tilde{m})\langle N\rangle Q'$. Then $U \mid R \xrightarrow{\langle - \rangle} (\nu\tilde{m})\langle N\rangle Q' \mid R$, by rule (Par), and then $U \mid R \xrightarrow{\langle - \rangle\hat{\ } n[R_1]R_2} Z$. We are done, since $Q \mid R \Longrightarrow U \mid R$ and $Z \Longrightarrow O'$.

- $\underline{n[P]\mathcal{S}n[Q] \text{ because } P\mathcal{S}Q}$. 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 under 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 $Q \overset{(M)\hat{\ }}{\Longrightarrow} Q'$ with $P'\mathcal{S}Q'$. From this, we have $n[Q] \overset{n \text{ get } M}{\Longrightarrow} n[Q'] \equiv O'$, and $O\mathcal{S}O'$ because $\mathcal{S}$ is closed by the ambient constructor.
- $n[P] \xrightarrow{\text{exit}\langle m,k\rangle RS} O$ because $n[P] \xrightarrow{\text{exit}\langle m,k\rangle} (\nu)\langle n[P']\rangle\mathbf{0}$, where O is structurally equivalent to $n[P'] \mid m[R] \mid S\{x := n\}$. The latter transition must have been derived from $P \xrightarrow{\text{out}\langle m,k\rangle} P'$. Since $P\mathcal{S}Q$, by the induction hypothesis there exists Q' such that it follows by induction that $Q \Longrightarrow \xrightarrow{\text{out}\langle m,k\rangle} \Longrightarrow Q'$ and $P'\mathcal{S}Q'$. Then

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

  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 $(\nu\tilde{p})(m[P_1] \mid n[P_2] \mid R\{x := m\})$. 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. $Q \Longrightarrow \xrightarrow{\mathsf{exit}\langle n,k\rangle\mathbf{0}\ R} Z \Longrightarrow O'$ where $Z \equiv (\nu\tilde{q})(l[Q_1] \mid n[Q_2] \mid R\{x := l\})$ and $O\mathscr{S}O'$. By Lemma 13(1), we then have the desired transition $n[Q] \xRightarrow{\mathsf{pop}\langle k\rangle R} (\nu\tilde{q})(l[Q_1] \mid n[Q_2] \mid R\{x := l\}) \Longrightarrow O'$.

- $n[P] \xrightarrow{n\ \mathsf{put}\ \langle -\rangle R} O$ because $n[P] \xrightarrow{n\ \mathsf{put}\ \langle -\rangle} (\nu\tilde{p})\langle M\rangle n[P']$, where O is structurally equivalent to $(\nu\tilde{p})(n[P'] \mid R\{x := M\})$. The last transition must derive from $P \xrightarrow{\langle -\rangle^{\hat{}}} (\nu\tilde{p})\langle M\rangle P'$, from which one derives

$$P \xrightarrow{\langle -\rangle^{\hat{}} n[\mathbf{0}]R} (\nu\tilde{p})(n[P'] \mid R\{x := M\})$$

an application of (Output $\hat{}$ HO). Since $P\mathscr{S}Q$, by induction hypothesis it follows that there exists O' such that $Q \xRightarrow{\langle -\rangle^{\hat{}} n[\mathbf{0}]R} O'$ and $O\mathscr{S}O'$. 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 13(2), we then have $n[Q] \xRightarrow{n\ \mathsf{put}\ \langle -\rangle R} O'$ as desired.
- The remaining cases, namely (Enter HO) and (Co-enter HO) are similar to and simpler than the previous ones.

• $\underline{(\nu n)P\mathscr{S}(\nu n)Q \text{ because } P\mathscr{S}Q}$. Assume $(\nu n)P\mathscr{S}(\nu n)Q$, and consider the transition $(\nu n)P \xrightarrow{\lambda} P'$. 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 $\mathscr{S}$ 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 $(\nu n)P\mathscr{S}(\nu n)Q$, and $(\nu n)P \xrightarrow{\mathsf{pop}\langle k\rangle R} O$, and let this transition be derived by (Pop HO) from $(\nu n)P \xrightarrow{\mathsf{pop}\langle k\rangle} (\nu n,\tilde{p})\langle m\rangle P'$, with $O \equiv (\nu n,\tilde{p})(P' \mid R\{x := m\})$ and $\{n,\tilde{p}\} \cap \mathrm{fn}(R) = \emptyset$. We need to find a weak transition $(\nu n)Q \xRightarrow{\mathsf{pop}\langle k\rangle R} O'$ with $O\mathscr{S}O'$. The transition from $(\nu n)P$ must derive by (Res) from $P \xrightarrow{\mathsf{pop}\langle k\rangle} (\nu\tilde{p})\langle m\rangle P'$. From this transition, we have $P \xrightarrow{\mathsf{pop}\langle k\rangle R} P^*$ with $P^* \equiv (\nu\tilde{p})(P' \mid R\{x := m\})$ (and thus $O \equiv (\nu n)P^*$). Since $P\mathscr{S}Q$ we find a weak transition of the form $Q \Longrightarrow V \xrightarrow{\mathsf{pop}\langle k\rangle R} Z \Longrightarrow Q^*$ with $P^*\mathscr{S}Q^*$. By examining the transition $V \xrightarrow{\mathsf{pop}\langle k\rangle R} Z$, we see that it must derive from $V \xrightarrow{\mathsf{pop}\langle k\rangle} (\nu\tilde{q})\langle l\rangle V'$, for $Z \equiv (\nu\tilde{q})(V' \mid R\{x := l\})$ and a suitable l. Now $(\nu n)V \xrightarrow{\mathsf{pop}\langle k\rangle} (\nu n,\tilde{q})\langle l\rangle V'$ derives by (Res), and then by (Pop HO), $(\nu n)V \xrightarrow{\mathsf{pop}\langle k\rangle R} \equiv (\nu n)Z$. Since $(\nu n)Q \Longrightarrow (\nu n)V$ and $(\nu n)Z \Longrightarrow (\nu n)Q^*$, we have found a weak transition $(\nu n)Q \xRightarrow{\mathsf{pop}\langle k\rangle R} (\nu n)Q^*$. We are done since $(\nu n)Q^*\mathscr{S}(\nu n)P^*$ follows by $P^*\mathscr{S}Q^*$ and the assumption that $\mathscr{S}$ is closed by restriction.

• $\underline{!P\mathscr{S}!Q \text{ because } P\mathscr{S}Q}$. Assume $!P\mathscr{S}!Q$, 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 $P\mathscr{S}Q$, we may use induction to find a move $Q \xRightarrow{\lambda} Q'$ with $P'\mathscr{S}Q'$. Thus, $!Q \xRightarrow{\lambda} !Q \mid Q'$ by an application of (Repl). Then we have $!P\mathscr{S}!Q$ and $P'\mathscr{S}Q'$. Since $\mathscr{S}$ is closed by parallel composition, this implies $!P \mid P'\mathscr{S}!Q \mid Q'$, 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 that $\approx_c$ is contained in our relation of barbed congruence. An alternative notion of labelled bisimilarity that completely captures barbed congruence will be discussed in §10.

Theorem 15 (Soundness of full bisimilarity)**.** *If $P \approx_c Q$ then $P \cong Q$.*

Proof. By Theorem 14, we know that $\approx_c$ is preserved by all contexts. Then, we need to show that, when restricted to closed processes, $\approx_c$ is barb preserving and reduction closed. Since on closed processes $\approx_c$ coincides with $\approx$, it enough to show that $\approx$ reduction closed and barb preserving. Assume, then, $P \approx Q$. If $P \downarrow_n$, by Lemma 7, $P \xrightarrow{n\ \overline{\mathsf{enter}}(m,k)}$ for some m, k. From $P \approx Q$, we know that $Q \xRightarrow{n\ \overline{\mathsf{enter}}(m,k)}$, from which $Q \Downarrow_n$ again by Lemma 7. Now assume that $P \longrightarrow P'$. By Theorem 6, $P \xrightarrow{\tau} \equiv P'$. Since $P \approx Q$, there exists Q' such that $Q \Longrightarrow Q'$ and $P' \equiv \approx \equiv Q'$, as desired. □

5. Algebraic laws

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. A richer algebraic theory may be obtained by adopting a typed version of the calculus which supports an accurate type system to control both agent mobility and communication as for instance that in [15]. We concentrate on the untyped calculus instead, to further emphasize the import of the new model of communication and mobility.

5.1. Mobility

The first set of laws are related to mobility and inherited from Safe Ambients (without passwords). These laws 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 16.

(1) $(\boldsymbol{\nu} p)(m[\mathsf{in}\langle n, p\rangle.P] \mid n[\overline{\mathsf{in}}(x, p).Q]) \cong (\boldsymbol{\nu} p)(n[Q\{x := m\} \mid m[P]])$

(2) $l[m[\mathsf{in}\langle n, p\rangle.P] \mid n[\overline{\mathsf{in}}(x, p).Q]] \cong l[n[Q\{x := m\} \mid m[P]]]$

(3) $(\boldsymbol{\nu} p)(n[m[\mathsf{out}\langle n, p\rangle.P]] \mid \overline{\mathsf{out}}(x, p).Q) \cong (\boldsymbol{\nu} p)(m[P] \mid Q\{x := m\})$

(4) $l[n[m[\mathsf{out}\langle n, p\rangle.P]] \mid \overline{\mathsf{out}}(x, p).Q] \cong l[m[P] \mid Q\{x := m\}]$

Proof. For all cases, let *LHS* and *RHS* denote, respectively, the left-hand side and the right-hand side of the equation, and $\mathscr{I}$ is the identity. The proof follows by showing that the relation

$$\mathscr{S}_\sigma \triangleq \{(LHS\sigma, RHS\sigma),\ (RHS\sigma, LHS\sigma)\} \cup \mathscr{I}$$

is a bisimulation for all closing substitutions σ. □

5.2. Garbage collection

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

Theorem 17. *For any I, J, H finite:*

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

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

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 (without 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.

5.3. Buffer laws

A further set of laws shows how outputs distribute over ambients. A first observation is that $l[\ \Pi_{j\in J}\langle\tilde{M}_j\rangle.P_j\ ] \cong \Pi_{j\in J}\ l[\langle\tilde{M}_j\rangle.P_j\ ]$. This follows directly by Theorem (17)(2), as both sides are equivalent to the null process. The equation is a consequence of the semantics of communication of NBA, which makes local communication not observable. This 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 $C[\cdot] = [\cdot] \mid n[\mathsf{in}\langle l\rangle.(x)^\uparrow.(x)^\uparrow.\mathsf{out}\langle l\rangle.\langle\rangle^\uparrow]$ distinguishes them, as $C[P] \Downarrow_n$ while $C[Q] \not\Downarrow_n$, according to the semantics of BA (cf. Section 1).

Theorem 18. *For any finite J, $l[\ \Pi_{j\in J}\langle\tilde{M}_j\rangle^{\hat{}}\ ] \cong \Pi_{j\in J}\ l[\langle\tilde{M}_j\rangle^{\hat{}}\ ]$*

Proof. We reason by induction on the size of J. For the base case, when $J = \emptyset$, the equation follows by Theorem 17(1). For the inductive case, we first show that

$$(l[\ \Pi_{j\in J}\langle\tilde{M}_j\rangle^{\hat{}}\ ])\sigma \ \approx\ (l[\langle\tilde{M}_k\rangle^{\hat{}}\ ] \mid l[\ \Pi_{j\in J\setminus\{k\}}\langle\tilde{M}_j\rangle^{\hat{}}\ ])\sigma \qquad (1)$$

for all closing substitutions σ. Let *LHS* and *RHS* denote the left-hand side and right-hand side of (1), respectively. We give a direct proof, showing that the derivatives of the two terms are bisimilar.

Assume $LHS \xrightarrow{\lambda} P'$. An inspection of the transition rules shows that $\lambda = l\sigma$ put $\langle - \rangle S$, and that $P' \equiv (l[\ \Pi_{j \in J\setminus\{k\}} \langle \tilde{M}_j \rangle^{\hat{}}\ ] \mid S\{\tilde{x} := \tilde{M}_k\})\sigma$, for some process S, and $k \in J$. For the right-hand side, first observe that $(l[\langle \tilde{M}_k \rangle^{\hat{}}])\sigma \xrightarrow{l\sigma\ \mathsf{put}\ \langle - \rangle} (\boldsymbol{\nu})\langle \tilde{M}_k\sigma \rangle l\sigma[\,]$. Then, an application of the (Par) rule allows us to derive

$$RHS \xrightarrow{l\sigma\ \mathsf{put}\ \langle - \rangle} (\boldsymbol{\nu})\langle \tilde{M}_k\sigma \rangle (l[\,] \mid \Pi_{j \in J\setminus\{k\}}\, l[\langle \tilde{M}_j \rangle^{\hat{}}])\sigma.$$

From this, by (Output HO) we have $RHS \xrightarrow{\lambda} l\sigma[\,] \mid P'$ which is what we need, because $l\sigma[] \approx \mathbf{0}$. The reasoning for the symmetric case is essentially the same.

From (1), we have $l[\ \Pi_{j \in J} \langle \tilde{M}_j \rangle^{\hat{}}\ ] \cong l[\langle \tilde{M}_k \rangle^{\hat{}}\ ] \mid l[\ \Pi_{j \in J\setminus\{k\}} \langle \tilde{M}_j \rangle^{\hat{}}\ ]$ by Theorem 14. Then we may use the induction hypothesis and conclude

$$\begin{aligned} l[\ \Pi_{j \in J} \langle \tilde{M}_j \rangle^{\hat{}}\ ] &\cong l[\langle \tilde{M}_k \rangle^{\hat{}}\ ] \mid l[\ \Pi_{j \in J\setminus\{k\}} \langle \tilde{M}_j \rangle^{\hat{}}\ ] \\ &\cong l[\langle \tilde{M}_k \rangle^{\hat{}}\ ] \mid \Pi_{j \in J\setminus\{k\}}\, l[\langle \tilde{M}_j \rangle^{\hat{}}\ ] \\ &\equiv \Pi_{j \in J}\, l[\langle \tilde{M}_j \rangle^{\hat{}}\ ] \end{aligned}$$

as desired. □

The equation holds in BA as well. In neither case it generalises to output prefixes with non-null continuation, as in general $n[P_1 \mid P_2] \not\cong n[P_1] \mid n[P_2]$. As a simple example, take $P_1 = (\,).\overline{\mathsf{in}}(x,n).\mathbf{0}$ and $P_2 = \langle\,\rangle$. Then, $n[P_1] \mid n[P_2] \cong \mathbf{0}$, by Theorem 17, while $n[P_1 \mid P_2] \Longrightarrow n[\overline{\mathsf{in}}(x,n)]$ which is active and observable.

5.4. Communication

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

Theorem 19. *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) $(\boldsymbol{\nu} l)(\,(\tilde{x})^l.P \mid l[\langle \tilde{M} \rangle^{\hat{}}.Q]\,) \cong (\boldsymbol{\nu} l)(\,P\{\tilde{x} := \tilde{M}\} \mid l[Q]\,)$
(3) $m[(\tilde{x})^l.P \mid l[\langle \tilde{M} \rangle^{\hat{}}.Q]] \cong m[P\{\tilde{x} := \tilde{M}\} \mid l[Q]]$

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

Proof. Again, by showing that the relation $\mathcal{S}_\sigma$, defined below, is a bisimulation for all possible σ. In all cases the bisimulation has the same form

$$\mathcal{S}_\sigma \triangleq \{(LHS\sigma, RHS\sigma),\ (RHS\sigma, LHS\sigma)\} \cup \mathcal{I}$$

where LHS and RHS denote, respectively, the left-hand side and the right-hand side of the equation, and $\mathcal{I}$ is the identity. □

The first equation, 19(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.

6. The type system

We proceed our analysis of NBA by studying a typed version of the calculus. As in its companion calculi, types may be employed for a variety of purposes in the calculus: our present goal is to provide static guarantees of a basic safety property, that is the type-correctness of hierarchical exchanges between ambients. Interestingly, the absence of communication interference conveyed by the combination of the new semantics of exchanges and the new protocols for ambient mobility yields a simple and rather elegant type system, which on the other hand allows a degree of flexibility comparable with that of the moded types of [3]. The use of types is illustrated in Section 8, where a typed encoding of BA into NBA is presented. Similarly, the encodings of the π-calculus illustrated in Section 7 could be easily typed. More interestingly, drawing on the type system of this section, one may derive more powerful systems to analyze and enforce diverse behavioral invariants such as those studied for BA, namely: access control [3], absence of information flow [10], or mobility control [15].

The typed syntax is derived directly from the untyped version of the calculus by associating types with names and variables introduced by restrictions and input prefixes. Accordingly, we henceforth denote restricted processes by $(\nu n{:}W)P$ and input processes by $(\tilde{x}{:}\,\tilde{W})P$, where W is a message type, defined next, and $\tilde{x}{:}\,\tilde{W}$ is short for $x_1 : W_1, \ldots, x_k : W_k$. The relations of structural congruence and reduction extend to the typed syntax as expected. The structure of types is defined below.

$$\begin{array}{llll}
\textit{MessageTypes} & W & ::= \mathsf{N}[E] & \text{ambient/password} \\
 & & \mid\ \mathsf{C}[E] & \text{capability} \\
\textit{ExchangeTypes} & E,F & ::= \mathsf{shh} & \text{no exchange} \\
 & & \mid\ W_1 \times \ldots \times W_k & \text{tuples } (k \geqslant 0) \\
\textit{ProcessTypes} & T & ::= [E,F] & \text{composite exchange}
\end{array}$$

$\mathsf{N}[E]$ is the type of ambients enclosing processes with upward exchanges of type E. Differently from the type systems of [3], here ambient types are defined as just one-place constructors. This simplification is a direct consequence of the semantics of communication, which guarantees the absence of interferences between the local exchanges of an ambient and the upward exchanges of its nested sub-ambients.

In addition, in the present system the types of the form $\mathsf{N}[E]$ also serve as the types of passwords: hence, $\mathsf{N}[E]$ is indeed the class of *name* types. When used as a password type, $\mathsf{N}[E]$ informs on the type E of the upward exchanges of any ambient whose movement is probed by a $\mathsf{N}[E]$ 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 types: 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, $\mathsf{C}[E]$ is the type of capabilities exercised within ambients with upward exchanges of type E. Tracing the type E is necessary to provide static guarantees of type safety: this is required by 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 have the same structure as in previous type systems for BA. More precisely, an exchange type E can be either a tuple of messagee types $W_1 \times \cdots \times W_k$, or shh to signal no exchange. In addition, type shh 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 $\mathsf{N}[\mathsf{shh}]$ will not be used by the receiving environment. Thus, unless the target ambient knows the name n, the use of a $\mathsf{N}[\mathsf{shh}]$ password guarantees that ambients can safely move irrespective of their upward exchanges. Finally, $[E, F]$ is the type of processes with local exchanges of type E and upward exchanges of type F. We often write $\tilde{W}$ as a shorthand for the exchange type $W_1 \times \cdots \times W_k$.

Table 9 gives the typing rules for messages. The rules for valid type environments are completely standard. The notation $F \leqslant G$, with F and G exchange types, holds true if and only if $F \in \{\mathsf{shh}, G\}$; operator $\sqcup$ is the (partial) *lub* operator associated with $\leqslant$. 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 11, they interpret the types of passwords as *interfaces* for mobility. In particular, if the type associated with the password N is $\mathsf{N}[\tilde{W}]$, then N requires any ambient relying upon N for mobility to have upward exchanges of type $\tilde{W}$ [cf. rules (Prefix) and (Amb) in Table 11]. If, instead, $F = \mathsf{shh}$, then the moving ambient can have upward exchanges of any type G: this is sound, because, as we said above, a move based on an $\mathsf{N}[\mathsf{shh}]$ password does not reveal the name of the incoming ambient to the target context [cf. rules (Co-In/Out-silent) in Table 11 and their explanation in the following]. Rule (Path) follows the same intuition: it is applicable only when $E_1 \sqcup E_2$ is defined.

Tables 10 and 11 define the typing of processes. The rules in Table 10 are standard. The ambient and prefix rules in Table 11 complement those in Table 9 in governing mobility. Rule (Amb) is standard, and construes the type $\mathsf{N}[E]$ as the interface of the ambient M for any process that

Table 9
Good messages: $\Gamma \vdash M : W$

(Env Empty) $\dfrac{}{\varnothing \vdash \diamond}$	(Env name) $\dfrac{\Gamma \vdash \diamond \quad a \notin \mathsf{Dom}(\Gamma)}{\Gamma, a : W \vdash \diamond}$
(Projection) $\dfrac{\Gamma, a : W, \Gamma' \vdash \diamond}{\Gamma, a : W, \Gamma' \vdash a : W}$	(Path) $\dfrac{\Gamma \vdash M_1 : \mathsf{C}[E_1] \quad \Gamma \vdash M_2 : \mathsf{C}[E_2]}{\Gamma \vdash M_1.M_2 : \mathsf{C}[E_1 \sqcup E_2]}$
(In) $\dfrac{\Gamma \vdash M : \mathsf{N}[E] \quad \Gamma \vdash N : \mathsf{N}[F] \quad (F \leqslant G)}{\Gamma \vdash \mathsf{in}\langle M, N\rangle : \mathsf{C}[G]}$	(Out) $\dfrac{\Gamma \vdash M : \mathsf{N}[E] \quad \Gamma \vdash N : \mathsf{N}[F] \quad (F \leqslant G)}{\Gamma \vdash \mathsf{out}\langle M, N\rangle : \mathsf{C}[G]}$

Table 10
Good processes I: $\Gamma \vdash P : [E,F]$

(Par) $\dfrac{\Gamma \vdash P : [E,F] \quad \Gamma \vdash Q : [E,F]}{\Gamma \vdash P \mid Q : [E,F]}$	(Repl) $\dfrac{\Gamma \vdash P : [E,F]}{\Gamma \vdash !P : [E,F]}$
(Dead) $\dfrac{\Gamma \vdash \diamond}{\Gamma \vdash \mathbf{0} \ : [E,F]}$	(New) $\dfrac{\Gamma, n : \mathsf{N}[G] \vdash P : [E,F]}{\Gamma \vdash (\boldsymbol{\nu} n : \mathsf{N}[G])P : [E,F]}$

Table 11
Good processes II

(Amb) $\dfrac{\begin{array}{c}\Gamma \vdash M : \mathsf{N}[E] \\ \Gamma \vdash P : [F,E]\end{array}}{\Gamma \vdash M[P] : [G,H]}$	(Prefix) $\dfrac{\begin{array}{c}\Gamma \vdash M : \mathsf{C}[F] \\ \Gamma \vdash P : [E,G] \quad (F \leqslant G)\end{array}}{\Gamma \vdash M.P : [E,G]}$
(Co-in) $\dfrac{\begin{array}{c}\Gamma \vdash M : \mathsf{N}[\tilde{W}] \\ \Gamma, x : \mathsf{N}[\tilde{W}] \vdash P : [E,F]\end{array}}{\Gamma \vdash \overline{\mathsf{in}}(x,M).P : [E,F]}$	(Co-out) $\dfrac{\begin{array}{c}\Gamma \vdash M : \mathsf{N}[\tilde{W}] \\ \Gamma, x : \mathsf{N}[\tilde{W}] \vdash P : [E,F]\end{array}}{\Gamma \vdash \overline{\mathsf{out}}(x,M).P : [E,F]}$
(Co-in-silent) $\dfrac{\begin{array}{c}\Gamma \vdash M : \mathsf{N}[\mathsf{shh}] \\ \Gamma \vdash P : [E,F] \quad (x \notin \mathrm{fv}(P))\end{array}}{\Gamma \vdash \overline{\mathsf{in}}(x,M).P : [E,F]}$	(Co-out-silent) $\dfrac{\begin{array}{c}\Gamma \vdash M : \mathsf{N}[\mathsf{shh}] \\ \Gamma \vdash P : [E,F] \quad (x \notin \mathrm{fv}(P))\end{array}}{\Gamma \vdash \overline{\mathsf{out}}(x,M).P : [E,F]}$
(Input) $\dfrac{\Gamma, \tilde{x} : \tilde{W} \vdash P : [\tilde{W},E]}{\Gamma \vdash (\tilde{x} : \tilde{W}).P : [\tilde{W},E]}$	(Output) $\dfrac{\Gamma \vdash \tilde{M} : \tilde{W} \quad \Gamma \vdash P : [\tilde{W},E]}{\Gamma \vdash \langle \tilde{M} \rangle.P : [\tilde{W},E]}$
(Input $\hat{}$) $\dfrac{\Gamma, \tilde{x} : \tilde{W} \vdash P : [E,\tilde{W}]}{\Gamma \vdash (\tilde{x} : \tilde{W})^{\hat{}}.P : [E,\tilde{W}]}$	(Output $\hat{}$) $\dfrac{\Gamma \vdash \tilde{M} : \tilde{W} \quad \Gamma \vdash P : [E,\tilde{W}]}{\Gamma \vdash \langle \tilde{M} \rangle^{\hat{}}.P : [E,\tilde{W}]}$
(Input M) $\dfrac{\begin{array}{c}\Gamma \vdash M : \mathsf{N}[\tilde{W}] \\ \Gamma, \tilde{x} : \tilde{W} \vdash P : [G,H]\end{array}}{\Gamma \vdash (\tilde{x} : \tilde{W})^{M}.P : [G,H]}$	(Output N) $\dfrac{\begin{array}{c}\Gamma \vdash N : \mathsf{N}[\tilde{W}] \\ \Gamma \vdash \tilde{M} : \tilde{W} \quad \Gamma \vdash P : [G,H]\end{array}}{\Gamma \vdash \langle \tilde{M} \rangle^{N}.P : [G,H]}$

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 $\mathsf{N}[\tilde{W}]$ as in rules (CO-IN) and (CO-OUT), we are guaranteed that $\tilde{W}$ is indeed the type of the exchanges of the incoming ambient. If instead the password type is $\mathsf{N}[\mathsf{shh}]$, no such guarantee can be made, as easily verified by inspecting (PREFIX) and the communication rules in Table 11. 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 generalize 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 11 are those for input output and contain no surprise. In rules for output the judgment $\Gamma \vdash \tilde{M} : \tilde{W}$ stands for the judgments $\Gamma \vdash M_i : W_i$ for $i = 1, \ldots, n$ when $\tilde{W} = W_1 \times \cdots \times W_n$. The first four rules require that the types of the values exchanged locally, respectively, upwardly, comply with the local, respectively, upward, exchange type recorded into the process type. On the other hand, rules (INPUT M) and (OUTPUT M) require that the types of values exchanged with a sub-ambient M comply with the type of upward communications of M, that is with the exchange type $\tilde{W}$ recorded into the interface of M.

To illustrate the typing rules, consider the process

$$n[\mathsf{in}\langle m,k\rangle.\langle M\rangle^{\hat{}}.P] \mid m[\overline{\mathsf{in}}(x,k).Q],$$

where we assume that $M : W_M$. In order for ambient n to be well typed, it must be the case that $n : \mathsf{N}[W_M]$ and $\mathsf{in}\langle m,k\rangle : \mathsf{C}[W_M]$. In addition, the password k must have either type $\mathsf{N}[W_M]$ or $\mathsf{N}[\mathsf{Shh}]$. In the first case the process Q may contain free occurrences of x, and hence may safely communicate with n (via x) by means of downward exchanges of the form $(y : W_M)^x.Q'$. On the other hand, if the ambient n relies on a silent password, i.e., if $k : \mathsf{N}[\mathsf{Shh}]$, the typing rules ensure that process Q has no free occurrences of x. Correspondingly, Q can perform no downward exchanges with the incoming ambient, unless it already knows the name n (with its type $\mathsf{N}[W_M]$).

The soundness of the type system is a direct consequence of the following, standard result.

Theorem 20 (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 $n[\mathsf{in}\langle m,k\rangle.P_1 \mid P_2]$ (similarly $n[\mathsf{out}\langle m,k\rangle.P_1 \mid P_2]$) is well typed for $n : \mathsf{N}[E]$, then $k : \mathsf{N}[F]$ for $F \leqslant E$, 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 $F = \mathsf{shh}$, in which case the context probing n with k must know the name n, hence its type $\mathsf{N}[E]$, to have exchanges with $n[P_1 \mid P_2]$. □

7. Encoding π-calculus channels

As a first test of expressive power for NBA, we give an encoding of the following, choice-free fragment of the synchronous π-calculus [17].

$$P \in \pi ::= \overline{a}\langle\tilde{b}\rangle.P \mid a(\tilde{x}).P \mid P|P \mid (\nu a)P$$

There are several choices for the encoding: we discuss two of them as representatives. In our first solution the exchanges of values in π-calculus is realized by means of the binding mechanism underlying NBA's co-actions. We only show the encoding of channels; the remaining clauses are defined compositionally.

$$\begin{aligned} \langle\!\langle\, \overline{a}\langle b\rangle.P \,\rangle\!\rangle &\triangleq (\boldsymbol{\nu} p)(p[b[\mathsf{out}\langle p,a\rangle.p[\mathsf{out}\langle b,p\rangle]]] \mid \overline{\mathsf{out}}(x,p).\,\langle\!\langle\, P \,\rangle\!\rangle\,) \quad (x \notin \mathrm{fv}(P)) \\ \langle\!\langle\, a(x).P \,\rangle\!\rangle &\triangleq \overline{\mathsf{out}}(x,a).\,\langle\!\langle\, P \,\rangle\!\rangle\,. \end{aligned}$$

For the monadic π-calculus the soundness proof of this encoding follows a standard pattern (see below). On the other hand, the extension to the case of polyadic communication is somewhat involved: in fact, communicating a tuple of k values requires a protocol involving k ambients, each one transmitting one element of the tuple, and additional machinery to ensure the atomicity of the protocol.

A simpler encoding is derived from the solution proposed in [3] now tailored to the new semantics of communication. This solution also illustrates the power of the synchronization mechanisms associated with NBA's co-actions.

$$\begin{aligned} \langle\!\langle\, \overline{a}\langle\tilde{b}\rangle.P \,\rangle\!\rangle &\triangleq (\nu p)\,(\, a[\langle\tilde{b}\rangle^{\hat{}}.a[\mathsf{out}\langle a,p\rangle]] \mid \overline{\mathsf{out}}(\_,p).\,\langle\!\langle\, P \,\rangle\!\rangle\,) \quad (p \notin \mathrm{fn}(P)) \\ \langle\!\langle\, a(\tilde{x}).P \,\rangle\!\rangle &\triangleq (\tilde{x})^{a}.\,\langle\!\langle\, P \,\rangle\!\rangle\,. \end{aligned}$$

Given the direct nature of this latter translation, 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 12. The definition is adapted from [19]: it uses concretions of the form $(\boldsymbol{\nu}\tilde{p})\langle\tilde{q}\rangle P$ with $\{\tilde{p}\} \subseteq \{\tilde{q}\}$, and relies on the same conventions for the notation $(\nu n)O$ and $O \mid Q$ defined in Section 3.

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 $\longrightarrow^{+}$ indicates *one* or more reduction steps.

Definition 21 (*Barbed expansion [27]*)**.** A reflexive and transitive relation $\mathscr{R}$ is an *expansion* if it is closed by context and whenever $P\mathscr{R}Q$, with P and Q closed processes, one has:

- for each name n, $P\downarrow_n$ implies $Q\Downarrow_n$, and $Q\downarrow_n$ implies $P\downarrow_n$.
- $P \longrightarrow P'$ implies $Q \longrightarrow^{+} Q'$ with $P'\mathscr{R}Q'$
- $Q \longrightarrow Q'$ implies $P\mathscr{R}Q'$ or $P \longrightarrow P'$ with $P'\mathscr{R}Q'$

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

Table 12
Commitments for the π-calculus

(Input)	(Output)
$\dfrac{}{a(\tilde{x}).P \xrightarrow{a(\tilde{b})} P\{\tilde{x} := \tilde{b}\}}$	$\dfrac{}{\overline{a}\langle\tilde{b}\rangle.P \xrightarrow{\overline{a}} (\boldsymbol{\nu})\langle\tilde{b}\rangle P}$
(comm)	
$\dfrac{P \xrightarrow{\overline{a}} (\boldsymbol{\nu}\tilde{c})\langle\tilde{b}\rangle P' \quad Q \xrightarrow{a(\tilde{b})} Q' \quad \mathrm{fn}(Q) \cap \{\tilde{c}\} = \emptyset}{P \mid Q \xrightarrow{\tau} (\boldsymbol{\nu}\tilde{c})(P' \mid Q')}$	
(Res)	(Par)
$\dfrac{P \xrightarrow{\alpha} O \quad a \notin \mathrm{fn}(\alpha)}{(\boldsymbol{\nu}a)P \xrightarrow{\alpha} (\boldsymbol{\nu}a)O}$	$\dfrac{P \xrightarrow{\alpha} O}{P \mid Q \xrightarrow{\alpha} O \mid Q}$

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

Proof. Let *LHS* and *RHS* denote the left-hand and right-hand sides, respectively, and let $\mathscr{R}$ be the following relation, where $C[\cdot]$ denotes any arbitrary context:

$$\mathscr{R} = \{\ (C[RHS], C[LHS])\ \} \cup \{\ (C[P], C[P \mid n[\mathbf{0}]]) \ \mid\ P \textit{ is a process}\ \}$$

Clearly, $\mathscr{R}$ is closed by context. That $\mathscr{R}$ is an expansion follows by observing that if $C[LHS]$ moves, as in $C[LHS] \longrightarrow R$, then either $R \equiv C'[LHS]$ with $C[\cdot] \longrightarrow C'[\cdot]$, or $R \equiv C[RHS \mid n[\mathbf{0}]]$. $\square$

As an immediate corollary of Lemma 22, we have:

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

We will use this latter relation in the proof of the following result.

Lemma 23 (Operational correspondence). *Let $P \in \pi$.*

(1) *Assume $P \xrightarrow{\alpha} O$. Then the following cases arise*:
 (a) $\alpha = a(\tilde{b})$, O is a process and $\langle\!\langle P \rangle\!\rangle \xrightarrow{(\tilde{b})^a} \ \gtrsim \langle\!\langle O \rangle\!\rangle$.
 (b) $\alpha = \overline{a}$, $O \equiv (\boldsymbol{\nu}\tilde{c})\langle\tilde{b}\rangle P'$ and one has $\langle\!\langle P \rangle\!\rangle \xrightarrow{a\ \mathsf{put}\ \langle - \rangle} (\boldsymbol{\nu}\tilde{c})\langle\tilde{b}\rangle P^*$ with $P^* \gtrsim \langle\!\langle P' \rangle\!\rangle$.
 (c) $\alpha = \tau$, O is a process and $\langle\!\langle P \rangle\!\rangle \xrightarrow{\tau} \ \gtrsim \langle\!\langle O \rangle\!\rangle$.

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

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, $\langle\!\langle P \rangle\!\rangle = (x)^a.\langle\!\langle P_1 \rangle\!\rangle$, and then $\langle\!\langle P \rangle\!\rangle \xrightarrow{(\tilde{b})^a} \langle\!\langle P_1 \rangle\!\rangle\{\tilde{x} := \tilde{b}\}$. We are done since $\langle\!\langle P_1 \rangle\!\rangle\{\tilde{x} := \tilde{b}\} = \langle\!\langle P_1\{\tilde{x} := \tilde{b}\} \rangle\!\rangle$.
- $P \xrightarrow{\alpha} O$ is $\bar{a}\langle\tilde{b}\rangle.P_1 \xrightarrow{\bar{a}} (\boldsymbol{\nu})\langle\tilde{b}\rangle P_1$. By definition,
  $\langle\!\langle P \rangle\!\rangle \equiv (\boldsymbol{\nu} p)(a[\langle\tilde{b}\rangle^{\hat{}}.a[\mathsf{out}\langle a,p\rangle]] \mid \overline{\mathsf{out}}(x,p).\langle\!\langle P_1 \rangle\!\rangle) \xrightarrow{a\ \mathsf{put}\ \langle - \rangle} \langle\!\langle P \rangle\!\rangle(\boldsymbol{\nu})\langle\tilde{b}\rangle P^*$

  for $p \notin (\mathrm{fn}(P_1) \cup \{\tilde{b}\})$, and $P^* \equiv (\boldsymbol{\nu} p)(a[a[\mathsf{out}\langle a,p\rangle]] \mid \overline{\mathsf{out}}(x,p).\langle\!\langle P_1 \rangle\!\rangle)$. Since $x \notin \mathrm{fv}(P_1)$, by Lemma 22, $P \gtrsim \langle\!\langle P_1 \rangle\!\rangle$ as desired.
- $P \xrightarrow{\alpha} O$ is $P_1 \mid P_2 \xrightarrow{\tau} (\boldsymbol{\nu}\tilde{c})(P_1' \mid P_2')$, derived from $P_1 \xrightarrow{\bar{a}} (\boldsymbol{\nu}\tilde{c})\langle\tilde{b}\rangle P_1'$, and from $P_2 \xrightarrow{a(\tilde{b})} P_2'$, with $\mathrm{fn}(P_2) \cap \{\tilde{c}\} = \emptyset$. By induction hypothesis, there exist P_1^* and P_2^* such that $\langle\!\langle P_1 \rangle\!\rangle \xrightarrow{a\ \mathsf{put}\ -} (\boldsymbol{\nu}\tilde{c})\langle\tilde{b}\rangle P_1^*$ and $\langle\!\langle P_2 \rangle\!\rangle \xrightarrow{(\tilde{b})^a} P_2^*$, with $P_1^* \gtrsim \langle\!\langle P_1' \rangle\!\rangle$ and $P_2^* \gtrsim \langle\!\langle P_2' \rangle\!\rangle$. An inspection of the translation shows that $\mathrm{fn}(P_2) \cap \{\tilde{c}\} = \emptyset$ implies $\mathrm{fn}(\langle\!\langle P_2 \rangle\!\rangle) \cap \{\tilde{c}\} = \emptyset$.
  Then $\langle\!\langle P_1 \mid P_2 \rangle\!\rangle \xrightarrow{\tau} (\boldsymbol{\nu}\tilde{c})(P_1^* \mid P_2^*)$. Since $\gtrsim$ is closed by context, from $P_1^* \gtrsim \langle\!\langle P_1' \rangle\!\rangle$ and $P_2^* \gtrsim \langle\!\langle P_2' \rangle\!\rangle$ we have $(\boldsymbol{\nu}\tilde{c})(P_1^* \mid P_2^*) \gtrsim (\boldsymbol{\nu}\tilde{c})(\langle\!\langle P_1' \rangle\!\rangle \mid \langle\!\langle P_2' \rangle\!\rangle)$. We are done since $(\boldsymbol{\nu}\tilde{c})(\langle\!\langle P_1' \rangle\!\rangle \mid \langle\!\langle P_2' \rangle\!\rangle) = \langle\!\langle (\boldsymbol{\nu}\tilde{c})(P_1' \mid P_2') \rangle\!\rangle$.
- 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, $\langle\!\langle P \rangle\!\rangle = (\tilde{x})^a.\langle\!\langle P_1 \rangle\!\rangle$, thus $\alpha = (\tilde{b})^a$ and $O = \langle\!\langle P_1 \rangle\!\rangle\{\tilde{x} := \tilde{b}\}$. On the other hand, in π one has $P \xrightarrow{a(\tilde{b})} P_1\{\tilde{x} := \tilde{b}\}$, and we are done since $\langle\!\langle P_1\{\tilde{x} := \tilde{b}\} \rangle\!\rangle = \langle\!\langle P_1 \rangle\!\rangle \{\tilde{x} := \tilde{b}\}$.
- $P = \bar{a}\langle\tilde{b}\rangle.P_1$. By definition, $\langle\!\langle P \rangle\!\rangle \equiv (\boldsymbol{\nu} p)(a[\langle\tilde{b}\rangle^{\hat{}}.a[\mathsf{out}\langle a,p\rangle]] \mid \overline{\mathsf{out}}(x,p).\langle\!\langle P_1 \rangle\!\rangle)$, with $p \notin (\mathrm{fn}(P_1) \cup \{\tilde{b}\})$. Thus, $O = (\boldsymbol{\nu})\langle\tilde{b}\rangle P_1^*$, derived with $\alpha = a\ \mathsf{put}\ -$, and with $P_1^* \equiv (\boldsymbol{\nu} p)(a[a[\mathsf{out}\langle a,p\rangle]] \mid \overline{\mathsf{out}}(x,p).\langle\!\langle P_1 \rangle\!\rangle)$, On the other hand, in π, $P \xrightarrow{\bar{a}} (\boldsymbol{\nu})\langle\tilde{b}\rangle P_1$. Now $P_1^* \gtrsim \langle\!\langle P_1 \rangle\!\rangle$ follows by Lemma 22.
- $P = P_1 \mid P_2$. By definition $\langle\!\langle P_1 \mid P_2 \rangle\!\rangle = \langle\!\langle P_1 \rangle\!\rangle \mid \langle\!\langle P_2 \rangle\!\rangle$. We have two sub-cases. If $\langle\!\langle P_1 \rangle\!\rangle \mid \langle\!\langle P_2 \rangle\!\rangle \xrightarrow{\alpha} O$ is obtained by (Par) the proof follows directly by the induction hypothesis. Otherwise, the transition must be of the form $\langle\!\langle P_1 \rangle\!\rangle \mid \langle\!\langle P_2 \rangle\!\rangle \xrightarrow{\tau} (\boldsymbol{\nu}\tilde{c})(P_1^* \mid P_2^*)$, derived by (Comm) from $\langle\!\langle P_1 \rangle\!\rangle \xrightarrow{(\tilde{b})^a} P_1^*$ and from $\langle\!\langle P_2 \rangle\!\rangle \xrightarrow{a\ \mathsf{put}\ -} (\boldsymbol{\nu}\tilde{c})\langle\tilde{b}\rangle P_2^*$, for $\mathrm{fn}(\langle\!\langle P_1 \rangle\!\rangle) \cap \{\tilde{c}\} = \emptyset$. The proof follows now routinely.
- $P = (\boldsymbol{\nu} n)P_1$. This case follows by the induction hypothesis and the fact that $\gtrsim$ is a congruence.

□

Lemma 23, extends readily to weak reductions. The proof of the following proposition derives exactly as in [2] (cf. [2], Proposition 3.6, p. 216).

Proposition 24. *Let $P \in \pi$:*

(1) *if $P \longrightarrow^* P'$ then $\langle\!\langle P \rangle\!\rangle \longrightarrow^* \gtrsim \langle\!\langle P' \rangle\!\rangle$.*
(2) *if $\langle\!\langle P \rangle\!\rangle \longrightarrow^* Q$, then there exists P' such that $P \longrightarrow^* P'$ and $Q \gtrsim \langle\!\langle P' \rangle\!\rangle$.*
(3) *$P \Downarrow_n$ if and only if $\langle\!\langle P \rangle\!\rangle \Downarrow_n$.*

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

Theorem 25 (Soundness). *If $\langle\!\langle P \rangle\!\rangle \cong \langle\!\langle Q \rangle\!\rangle$ in NBA then $P \cong Q$ in π.*

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

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

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

- If $P \downarrow_n$, then by an inspection of the encoding we see that $\langle\!\langle P \rangle\!\rangle \downarrow_n$, which in turn implies $\langle\!\langle Q \rangle\!\rangle \Downarrow_n$ and hence $Q \Downarrow_n$, as desired, by Proposition 24(3).
- Now assume $P \longrightarrow P'$. By Lemma 23(1) we know that $\langle\!\langle P \rangle\!\rangle \longrightarrow R \gtrsim \langle\!\langle P' \rangle\!\rangle$. Since $\langle\!\langle P \rangle\!\rangle \cong \langle\!\langle Q \rangle\!\rangle$, we find S such that $\langle\!\langle Q \rangle\!\rangle \longrightarrow^* S \cong R$. Then, by Proposition 24(2), there exists Q' such that $Q \longrightarrow^* Q'$ and $S \gtrsim \langle\!\langle Q' \rangle\!\rangle$. Then we have $\langle\!\langle P' \rangle\!\rangle \lesssim R \cong S \gtrsim \langle\!\langle Q' \rangle\!\rangle$, thus $\langle\!\langle P' \rangle\!\rangle \cong \langle\!\langle Q' \rangle\!\rangle$, that is $P' \mathscr{S} Q'$ as desired.
- The proofs of the symmetric cases are exactly the same. □

8. NBA versus BA

To formally relate BA and NBA 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 guarded-choice operator with a semantics à la CCS, namely: $\pi_1.P + \pi_2.Q \mid S \longrightarrow R$ if $\pi_1.P \mid S \longrightarrow R$ or $\pi_2.Q \mid S \longrightarrow R$.

As we illustrate below, this extension allows us to localize the gap between the two calculi in a single construct that we employ in the encoding to represent the interferences that arise in BA from a choice in the 'direction' of communication (cf. page 41). While it is certainly possible to build on existing π-calculus encodings (cf. [22,20]) to investigate the extent to which our use of choice is programmable in NBA, such investigation is beyond our present concerns. Indeed, the encoding is not meant to translate between the two calculi, but rather to support our claim that the expressivity gap between NBA and BA is only directly related to communication interferences.

Table 13
Encoding of BA into NBA with guarded choice

$\langle\!\langle P \rangle\!\rangle_n$	$= \overline{\mathsf{cross}} \mid \{\!\!\{ P \}\!\!\}_n$	
$\{\!\!\{ m[P] \}\!\!\}_n$	$= m[\, \langle\!\langle P \rangle\!\rangle_m]$	
$\{\!\!\{ (x)^a.P \}\!\!\}_n$	$= (x)^a.\{\!\!\{ P \}\!\!\}_n$	
$\{\!\!\{ (x).P \}\!\!\}_n$	$= (x).\{\!\!\{ P \}\!\!\}_n + (x)^{\hat{}}.\{\!\!\{ P \}\!\!\}_n + \overline{\mathsf{out}}(y, \mathsf{pw}).(x)^y.\{\!\!\{ P \}\!\!\}_n$	$y \notin \mathrm{fn}(P)$
$\{\!\!\{ (x)^\uparrow.P \}\!\!\}_n$	$= (\boldsymbol{\nu} p)p[\mathsf{out}\langle n, \mathsf{pr}\rangle.(x)^{\hat{}}.\mathsf{in}\langle n, p\rangle.\langle x\rangle^{\hat{}}\,] \mid \overline{\mathsf{in}}(y, p).(x)^y.\{\!\!\{ P \}\!\!\}_n$	$p, y \notin \mathrm{fn}(P)$
$\{\!\!\{ \langle M\rangle^a.P \}\!\!\}_n$	$= \langle M\rangle^a.\{\!\!\{ P \}\!\!\}_n$	
$\{\!\!\{ \langle M\rangle.P \}\!\!\}_n$	$= \langle M\rangle.\{\!\!\{ P \}\!\!\}_n + \langle M\rangle^{\hat{}}.\{\!\!\{ P \}\!\!\}_n + \overline{\mathsf{out}}(y, \mathsf{pr}).\langle M\rangle^y.\{\!\!\{ P \}\!\!\}_n$	$y \notin \mathrm{fn}(P)$
$\{\!\!\{ \langle M\rangle^\uparrow.P \}\!\!\}_n$	$= (\boldsymbol{\nu} p)p[\mathsf{out}\langle n, \mathsf{pw}\rangle.\langle M\rangle^{\hat{}}.\mathsf{in}\langle n, p\rangle.\langle\rangle^{\hat{}}\,] \mid \overline{\mathsf{in}}(y, p).()^y.\{\!\!\{ P \}\!\!\}_n$	$p, y \notin \mathrm{fn}(P)$

The encoding is defined parametrically over the name n of the ambient (if any) that encloses the process that we are encoding; in addition, it uses three distinguished, well-known names[4] $\mathsf{mv}, \mathsf{pr}, \mathsf{pw}$ as special-purpose passwords. To ease the notation, we use the following shorthands: $\overline{\mathsf{cross}} = !\overline{\mathsf{in}}(x, \mathsf{mv}) \mid !\overline{\mathsf{out}}(x, \mathsf{mv})$, $\mathsf{in}\langle n\rangle = \mathsf{in}\langle n, \mathsf{mv}\rangle$, and $\mathsf{out}\langle n\rangle = \mathsf{out}\langle n, \mathsf{mv}\rangle$.

We define two mutually recursive translations, $\langle\!\langle \cdot \rangle\!\rangle_n$ and $\{\!\!\{ \cdot \}\!\!\}_n$. The interesting cases are given in Table 13; the remaining cases are defined compositionally. The translation $\langle\!\langle \cdot \rangle\!\rangle_n$ 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 §5 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 BA to record that engaging in inter-ambient communications is an activity across ambient boundaries that may create grave interferences. For instance, $a[\, \langle x\rangle^\uparrow \mid \mathsf{out}\langle n, k\rangle.P\,]$ cannot be considered single-threaded, as illustrated by, say, the context $R \mid n[\, (x)^a.Q \mid [\cdot]\,]$. To ease the presentation, we work with a direct syntactic characterisation of single-threadedness, rather than providing a type system as in [13]. 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:

$$S ::= (\nu\tilde{p})\ \pi_1 \ldots \pi_k.M.S \mid (\nu\tilde{p})\ \pi_1 \ldots \pi_k.\langle M\rangle^\uparrow.S \mid (\nu\tilde{p})\ \pi_1 \ldots \pi_k.(x)^\uparrow.P\ (k \geqslant 0)$$

4 These names are used uniformly in the encoding, and do not depend on the parameter n.

Theorem 26. *If P and Q are single-threaded, then $\langle\!\langle P \rangle\!\rangle_n \cong \langle\!\langle Q \rangle\!\rangle_n$ implies $P \cong Q$.*

Proof sketch. The structure of the proof is the same as the one of Theorem 25, 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 $P \equiv (\nu \tilde{m})(n[\langle M \rangle^{\uparrow}.Q \mid R] \mid S)$, for $n \notin \{\tilde{m}\}$. 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). $\square$

As it is the case for most encodings in the literature of process calculi, the converse of Theorem 26, i.e., completeness, does not hold. This failure of completeness is a direct consequence of the absence of co-capabilities in BA, which makes observers in BA insensitive to the moves of any silent ambient. As a consequence, phenomena like *stuttering* [24] may not be observed in BA, whereas they are observable in NBA.

To see the problem, consider the following BA processes (we use a sum operator à la CCS to simplify the presentation: the example can be phrased equivalently using replication as suggested in [30]):

$$P \triangleq \mathsf{in}\, a.\mathsf{out}\, a.\mathsf{in}\, a.\langle\rangle^{\uparrow}, \quad Q \triangleq P + \mathsf{in}\, a.\langle\rangle^{\uparrow}$$

One can show that these two processes may not be distinguished in BA. On the other hand, the following context may tell the encoding of the two terms apart

$$C[\cdot] = b[\cdot] \mid a[\overline{\mathsf{in}}(\_, \mathsf{mv}).()^{b}.\langle\rangle^{\hat{}}] \mid ()^{a}.w[\overline{\mathsf{in}}(\_, p)]$$

In particular, one has $C[\langle\!\langle Q \rangle\!\rangle_b] \Downarrow_w$, while on the other hand $C[\langle\!\langle P \rangle\!\rangle_b] \not\Downarrow_w$ as the context does not offer co-capabilities to enable the out and the subsequent in move to be exercised to exhibit the barb w.

Notice that the discriminating power of the context relies critically on the use of the special-purpose password mv to detect ambient moves. It would seem, then, that a form of completeness for the encoding could be achieved with respect to NBA contexts that do not use the password mv. On the other hand, restricting the power of contexts in this manner makes the encoding unsound. To see the problem, consider the *perfect firewall* equation $(\nu n)n[P] \cong \mathbf{0}$ of [6]. This law is not valid in BA, nor is it valid in NBA, SA, or SAP. On the other hand, no observing context may tell $\langle\!\langle (\nu n)n[P] \rangle\!\rangle$ from $\langle\!\langle \mathbf{0} \rangle\!\rangle$, unless it uses the password mv to observe the moves of the former process.

8.1. A typed encoding

The encoding extends smoothly to the typed case. The definition is given inductively on the structure of terms, and in relation with 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. Indeed, the encoding of types is the most interesting aspect of the definition, as it provides further insight into the different nature of the communication primitives in the two calculi.

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 $\mathsf{amb}[E, F]$, where E is the type of local exchanges, and

F the type of the upward exchanges. Capabilities types, in turn, have the form $\mathsf{cap}[E]$, denoting capabilities exercised in ambients with upward exchanges of type E. Finally, process types have exactly the same structure (and interpretation) as in 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:

$$\begin{aligned}
&\{\!|\, \mathsf{amb}[E,F] \,|\!\} = \mathsf{N}[\{\!|\, E \,|\!\}],\\
&\{\!|\, \mathsf{cap}[E] \,|\!\} = \mathsf{C}[\mathsf{shh}],\\
&\{\!|\, \mathsf{shh} \,|\!\} = \mathsf{shh},\\
&\{\!|\, [E,F] \,|\!\} = [\{\!|\, E \,|\!\}, \{\!|\, E \,|\!\}].
\end{aligned}$$

Observe that the type traced in $\{\!|\, \mathsf{amb}[E,F] \,|\!\}$ 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 14. 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 assumption that $\mathsf{pr}_W, \mathsf{pw}_W : \mathsf{N}[W]$ 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 $\mathsf{mv} : \mathsf{N}[\mathsf{shh}]$.

Theorem 27. *If $\Gamma \vdash P : [E,F]$ is derivable in the simple type system for BA (cf. [3], pg.46) and $\Gamma(n) = \mathsf{N}[E,F]$, then $\langle\!\langle \Gamma \rangle\!\rangle \vdash \langle\!\langle \Gamma \triangleright P \rangle\!\rangle_n : \{\!|\, [E,F] \,|\!\}$ is derivable in NBA.*

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

9. 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.

9.1. A point-to-point communication server

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

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

Table 14
Typed encoding of BA into NBA with guarded choice

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

Ambient $\mathsf{w}(k)$ 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 $A(k, a, P) = a[\mathsf{in}\langle k, k\rangle.P \mid \mathsf{out}\langle k, k\rangle]$, where P performs the expected (upward) exchange (clearly, any realistic implementation will lock the out move until P has completed the intended exchanges). A complete implementation for the point-to-point server can be then defined as shown below.

$$\mathsf{p2p}(k) = (\boldsymbol{\nu} r)\ (\ r[\langle\,\rangle^{\hat{}}\,] \mid !\,(\,)^{r}.(\mathsf{w}(k) \mid \overline{\mathsf{out}}(\_, k).\overline{\mathsf{out}}(\_, k).r[\langle\,\rangle^{\hat{}}\,])\ )$$

The process $\mathsf{p2p}(k)$ 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 $\mathsf{w}(k)$ for a new protocol session. Given the configuration

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

with k different from all the a_i's and fresh in all the P_i's, we are guaranteed that at most one pair of agents can be active inside k at any given time.
In particular, take the following two agents:

$$B = b[(\nu l)(\mathsf{in}\langle k,k\rangle.\langle M\rangle^{\hat{}}.(P \mid l[\langle\,\rangle^{\hat{}}]) \mid ()^{l}.\mathsf{out}\langle k,k\rangle)]$$
$$C = c[(\nu l)(\mathsf{in}\langle k,k\rangle.(x)^{\hat{}}.(Q \mid l[\langle\,\rangle^{\hat{}}]) \mid ()^{l}.\mathsf{out}\langle k,k\rangle)]$$

Then one has:

$$\begin{aligned}
&(\nu k)(\ \mathsf{p2p}(k) \mid B \mid C \mid \Pi_{i\in I}A(k,a_i,P_i)\ )\\
&\longrightarrow^{*}\ (\nu k)(\ (\nu r)(\overline{\mathsf{out}}(\_,k).\overline{\mathsf{out}}(\_,k).r[\langle\,\rangle^{\hat{}}\,] \mid !\,()^{r}.(\mathsf{w}(k) \mid \overline{\mathsf{out}}(\_,k).\overline{\mathsf{out}}(\_,k).r[\langle\,\rangle^{\hat{}}\,])\\
&\qquad \mid k\,[\,b[(\nu l)(\langle M\rangle^{\hat{}}.(P \mid l[\langle\,\rangle^{\hat{}}]) \mid ()^{l}.\mathsf{out}\langle k,k\rangle)] \mid\\
&\qquad\quad c[(\nu l)((x)^{\hat{}}.(Q \mid l[\langle\,\rangle^{\hat{}}]) \mid ()^{l}.\mathsf{out}\langle k,k\rangle)]\\
&\qquad\quad (z)^{b}.\langle z\rangle^{c} \mid !(z)^{b}.\langle z\rangle^{c} \mid !(z)^{c}.\langle z\rangle^{b}\,]\\
&\qquad \mid \Pi_{i\in I}A(k,a_i,P_i)\\
&\cong\ (\nu k)(\ \mathsf{p2p}(k) \mid b[P] \mid c[Q\{x := M\}] \mid \Pi_{i\in I}A(k,a_i,P_i)\ )
\end{aligned}$$

This says that once (and if) the two agents B and C have reached the forwarder, no other agent knowing the key k can interfere in their exchange. The equivalence above follows by the mobility laws of Theorem 16 and the garbage collection laws of Theorem 17. In particular, once the two ambients are back at top level, the currently active instance of the forwarder k has the form $k[!(z)^{b}.\langle z\rangle^{c} \mid !(z)^{c}.\langle z\rangle^{b}]\cong\mathbf{0}$.

9.2. 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.

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

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:

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

The process $\mathsf{job}(M,k)$ enters the server $\mathsf{prtsrv}(k)$, 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.

10. 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 completeness of full bisimulation $\approx_c$. 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 $(x)^{\hat{}}$, and the associated transition $P \xrightarrow{(M)^{\hat{}}} P'$. To show that $\approx_c$ fully characterises $\cong$ (i.e., $\cong\ \subseteq \approx_c$), one needs to find a distinguishing context for the label $(M)^{\hat{}}$. This context is typically defined as $C[\cdot] = n[[\cdot]] \mid \langle M\rangle^n.R$, with R exhibiting some fresh barb so as to probe the label. The problem is that $\approx_c$ tests the continuation P' "at top level", whereas the context $C[\cdot]$ tests it within the ambient n 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 transitions of the form $P \xrightarrow{(M)^{\hat{}}n} n[P']$. These are not quite right, however, because the resulting bisimilarity relation is not a congruence. To make bisimilarity a congruence, we generalise this idea, and replace the transition $P \xrightarrow{(M)^{\hat{}}} P'$ with the higher-order transition $P \xrightarrow{(M)^{\hat{}}n[R]} n[P' \mid R]$. As we prove in this section, the labelled bisimilarity arising from transitions of this form is indeed closed under context. In addition, it also coincides with barbed congruence.

10.1. A refined labelled transition system

The set of (first-order) labels are defined as in Table 3. 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:

$$\langle\bullet\rangle P \mid P' \triangleq \langle\bullet\rangle(P \mid P') \quad \text{and} \quad (\nu\tilde{p})\langle\bullet\rangle P \triangleq \langle\bullet\rangle(\nu\tilde{p})P$$

Visible transitions. The transitions (Output), (Put) and (Exit) are as in Table 4, and so are the transitions (Input) and (co-cap) when $\eta \neq \hat{}$, and when $\pi(x) = \overline{\mathsf{out}}(x,k)$, respectively. The remaining transitions are given below.

(Cap)

$$\frac{M \in \{\mathsf{in}\langle n,k\rangle, \mathsf{out}\langle n,k\rangle\}}{M.P \xrightarrow{M} \langle\bullet\rangle P}$$

(Co-Cap)

$$\frac{\pi(x) \in \{\overline{\mathsf{in}}(x,k), \overline{\mathsf{out}}(x,k)\}}{\pi(x).P \xrightarrow{\pi(n)} \langle\bullet\rangle P\{x := n\}}$$

(Path)

$$\frac{M_1.(M_2.P) \xrightarrow{\alpha} O}{(M_1.M_2).P \xrightarrow{\alpha} O}$$

(Input$\hat{\ }$)

$$\frac{}{(x)^{\hat{}}.P \xrightarrow{(M)^{\hat{}}} \langle\bullet\rangle P\{x := M\}}$$

(Get)

$$\frac{P \xrightarrow{(M)^{\hat{}}} \langle\bullet\rangle P'}{m[P] \xrightarrow{m\ \mathsf{get}\ M} m[P']}$$

(Enter)

$$\frac{P \xrightarrow{\mathsf{in}\langle n,k\rangle} \langle\bullet\rangle P'}{m[P] \xrightarrow{\mathsf{enter}\langle n,k\rangle} (\nu)\langle m[P']\rangle\mathbf{0}}$$

(Co-enter)

$$\frac{P \xrightarrow{\overline{\mathsf{in}}(n,k)} \langle\bullet\rangle P'}{m[P] \xrightarrow{m\ \overline{\mathsf{enter}}(n,k)} (\nu)\langle P'\rangle\mathbf{0}}$$

(Exit)

$$\frac{P \xrightarrow{\mathsf{out}\langle n,k\rangle} \langle\bullet\rangle P'}{m[P] \xrightarrow{\mathsf{exit}\langle n,k\rangle} (\nu)\langle m[P']\rangle\mathbf{0}}$$

Structural and τ transitions. As in Tables 6 and 5, respectively.

Higher-order transitions. Those in Tables 7 and 8, plus the following one:

(Prefix HO)

$$\frac{P \xrightarrow{\alpha} \langle\bullet\rangle P' \quad \alpha \in \{(M)^{\hat{}}, \mathsf{cap}\ \langle n,k\rangle, \overline{\mathsf{in}}(n,k), \overline{\mathsf{out}}(n,k)\} \quad \mathrm{fv}(R) = \emptyset}{P \xrightarrow{\alpha\, m[R]} m[P' \mid R]}$$

Let now $\approx_{fa}$ denote the labelled bisimulation associated with the new transition system: formally, $\approx_{fa}$ is defined exactly as $\approx_{\mathrm{c}}$ in Definition 11. In particular, like $\approx_{\mathrm{c}}$, also $\approx_{fa}$ tests only transitions from processes to processes.

10.2. Full abstraction

We first show that $\approx_{fa}$ is closed by contexts.

Theorem 28. *$\approx_{fa}$ is a congruence.*

Proof. Similar to Theorem 14. 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 14. Let $\mathscr{S}$ be the least equivalence relation containing $\approx_{fa}$, closed by substitution and preserved by all operators. We show that $\mathscr{S}$ is a bisimulation (with respect to the new LTS).

- $\underline{\pi.P \mathscr{S} \pi.Q \text{ because } P\mathscr{S}Q}$. 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 $m[P \mid R]\mathscr{S}m[Q \mid R]$ follows by the induction hypothesis (as $P\mathscr{S}Q$ and $\mathscr{S}$ is a congruence).
  When $\pi(x) \in \{(x)^{\hat{}}, \overline{\mathsf{in}}(x,k)\}$, $\lambda = \pi(n)m[R]$ 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 $\mathscr{S}$ is closed under substitution and preserved by parallel composition and ambient constructor.
- $\underline{P \mid R\mathscr{S}Q \mid R \text{ because } P\mathscr{S}Q}$. 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 $\alpha = (M)^{\hat{}}$ as representative. By inspection of the LTS, the transition in question must have the form $P \mid R \xrightarrow{(M)^{\hat{}}m[R_1]} m[S \mid R_1]$, derived from $P \mid R \xrightarrow{(M)^{\hat{}}} \langle\bullet\rangle S$. 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 $P \xrightarrow{(M)^{\hat{}}} \langle\bullet\rangle S_P$. From this transition, we derive $P \xrightarrow{(M)^{\hat{}}m[R \mid R_1]} m[S_P \mid R \mid R_1]$ by (Prefix HO). Then, by induction hypothesis, we have $Q \Longrightarrow U \xrightarrow{(M)^{\hat{}}m[R \mid R_1]} V \Longrightarrow Q'$ with $m[S_P \mid R \mid R_1]\mathscr{S}Q'$. By examining the transition from U we know that there exists Z such that $V \equiv m[Z \mid R \mid R_1]$, and $U \xrightarrow{(M)^{\hat{}}} \langle\bullet\rangle Z$. Then one derives $U \mid R \xrightarrow{(M)^{\hat{}}} \langle\bullet\rangle(Z \mid R)$ by (Par). Thus, we have: $Q \mid R \Longrightarrow U \mid R \xrightarrow{(M)^{\hat{}}m[R_1]} m[Z \mid R \mid R_1] \Longrightarrow Q'$, as desired.
  The other case is when $S \equiv P \mid S_R$, and $R \xrightarrow{(M)^{\hat{}}} \langle\bullet\rangle S_R$. From this transition we derive $Q \mid R \xrightarrow{(M)^{\hat{}}} \langle\bullet\rangle(Q \mid S_R)$ by (Par). Then by an application of (Prefix HO), we have $Q \mid R \xrightarrow{(M)^{\hat{}}m[R_1]} m[Q \mid S_R \mid R_1]$. Summarising, for $\lambda = (M)^{\hat{}}m[R_1]$, we have $P \mid R \xrightarrow{\lambda} m[P \mid S_R \mid R_1]$, and we have found a weak transition $Q \mid R \overset{\lambda}{\Longrightarrow} m[Q \mid S_R \mid R_1]$. Then the proof follows from the induction hypothesis and the fact that $\mathscr{S}$ is closed by context.
- The cases for the remaining constructs, namely ambient, restriction, and parallel composition are proved similarly. □

Next we show that $\approx_{fa}$ and reduction barbed congruence coincide. We start by defining the following operator of internal choice, as in [14].

$$P \oplus Q = (\nu n)(n[\langle\rangle^{\hat{}}] \mid ()^n.P \mid ()^n.Q) \qquad (n \notin \mathrm{fn}(P,Q))$$

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. Following [14], we can then define two contexts that allow us to detect whether a generic process performs any action at all.

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

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

The ability to spy comes about when h_1 and h_2 are fresh. Then, a spy context exhibits both 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 3, and $n \in \mathrm{fn}(\alpha)$. Recall 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 fills the hole.

Lemma 29. *Let $C[\cdot]$ be a static context, R a process, n a name, and h_1, h_2 fresh names.*

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

Proof. By transition induction. □

A further lemma allows the spy contexts to be removed.

Lemma 30. *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*

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

Proof. The proof is a generalisation of the corresponding lemma in [14]. For part 1, since $\cong$ is closed under restriction,

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

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

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

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

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

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

We also need a last simple property, whose proof is immediate.

Lemma 31. *$P \mid R \cong Q \mid R$ and $\mathrm{fn}(R) \cap \mathrm{fn}(P,Q) = \emptyset$ implies $P \cong Q$.*

Theorem 32 (Full-abstraction)**.** *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 $Q \overset{\lambda}{\Longrightarrow} Q^*$ and $P^*{\cong}Q^*$. We reason by cases, depending on λ. We will often use the shorthand $h = f[\overline{\mathsf{in}}(x,h)]$, where f will always be assumed fresh.

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

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

with h_0–h_4 fresh. We have $C[P] \xrightarrow{\tau}\xrightarrow{\tau}\xrightarrow{\tau} m[P' \mid \mathrm{SPY_{in}}\langle h_1,h_2,R\rangle] \mid n[\,] \mid h_3$. Since $P{\cong}Q$, we know that $C[Q] \Longrightarrow Z{\cong}m[P' \mid \mathrm{SPY_{in}}\langle h_1,h_2,R\rangle] \mid h_3$. Therefore, $Z \Downarrow_{h_1,h_2}$ and $Z \not\Downarrow_{h_4}$. This implies that the transitions from $C[Q]$ have consumed the two co-capabilities. In particular, we have:

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

Thus, we know that

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

Since h_3 is fresh by hypothesis, by Lemma 31

$$m[P' \mid \mathrm{SPY_{in}}\langle h_1,h_2,R\rangle]{\cong}m[Q_4 \mid \mathrm{SPY_{in}}\langle h_1,h_2,R\rangle]$$

Then, letting $C_1[\cdot] = m[P' \mid \cdot]$, and $C_2[\cdot] = m[Q_4 \mid \cdot]$, by Lemma 30,

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

To conclude, we show that $Q \overset{\mathsf{in}\langle n,k\rangle m[R]}{\Longrightarrow} m[Q_4 \mid R]$. To see that, note that the reduction steps in $C[Q] \Longrightarrow Z$ above imply that $Q \Longrightarrow \xrightarrow{\mathsf{in}\langle n,k\rangle} \langle\bullet\rangle Q_1$ and $Q_1 \Longrightarrow Q_4$. Thus, $Q \Longrightarrow \xrightarrow{\mathsf{in}\langle n,k\rangle m[R]} m[Q_1 \mid R] \Longrightarrow m[Q_4 \mid R]$, as desired.
- The other cases of (Prefix HO) are proved in a similar way, choosing appropriate contexts. In particular,
  - when $\lambda = \mathsf{out}\langle n,k\rangle m[R]$, choose

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

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

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

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

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

where the h_i's are assumed fresh.

- $\lambda = \mathsf{enter}\langle n,k\rangle R$. 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:

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

with r, h_1–h_3 fresh. We have $C[P] \xrightarrow{\tau}\xrightarrow{\tau} C_1[\mathrm{SPY}_{\mathrm{in}}\langle h_1, h_2, R\{x := m\}\rangle]$. Since $P \cong Q$, $C[Q] \Longrightarrow Z \cong C_1[\mathrm{SPY}_{\mathrm{in}}\langle h_1, h_2, R\{x := m\}\rangle]$, for a suitable process Z. Thus, in particular, $Z \Downarrow_{h_1,h_2}$ and $Z \not\Downarrow_{h_3}$, which implies that the co-capability $\overline{\mathsf{in}}(x,k)$ must have been consumed in this derivation. Furthermore, by Lemma 29, the derivation must have the form:

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

with $C'[\cdot], C''[\cdot]$ and $C_2[\cdot]$ static contexts. From $C_1[\mathrm{SPY}_{\mathrm{in}}\langle h_1, h_2, R\{x := m\}\rangle] \cong Z$, by Lemma 30, we know that $C_1[R\{x := m\}] \cong C_2[R\{x := l\}]$. To conclude, it remains to show that $Q \overset{\lambda}{\Longrightarrow} C_2[R\{x := l\}]$. Examining the above sequence of reductions from $C[Q]$ we see that $Q \Longrightarrow \xrightarrow{\mathsf{enter}\langle n,k\rangle R} C'[R]$. Similarly, it is easily checked that

$$C'[\mathrm{SPY}_{\mathrm{in}}\langle h_1, h_2, R\{x := l\}\rangle] \Longrightarrow C_2[\mathrm{SPY}_{\mathrm{in}}\langle h_1, h_2, R\{x := l\}\rangle].$$

Then, by Lemma 29, 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 $\lambda = m\,\overline{\mathsf{enter}}(n,k)R$, choose

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

- when $\lambda = \mathsf{exit}\langle n,k\rangle R\, S$, choose

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

  This case requires extending Lemmas 29 and 30 to contexts with two holes. There is no difficulty in this extension, as the hypotheses of the lemmas 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

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

  This case also requires the extension to Lemmas 29 and 30 discussed above.
- when $\lambda = \mathsf{pop}\langle k\rangle R$, choose

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

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

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

- 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 $C[\cdot] = [\cdot] \mid n[\langle M\rangle\hat{}.r[\mathsf{out}\langle n,h_1\rangle]] \mid \overline{\mathsf{out}}(x,h_1).(h_2 \oplus h_3)$. □

Theorem 33. *Relations $\approx_{fa}$ and $\cong$ coincide.*

Proof. By Theorem 28, reasoning as in the proof of Theorem 15 we show that $\approx_{fa} \subseteq \cong$. The opposite inclusion follows by Theorem 32. □

11. Conclusion

We have developed new foundations for the calculus of Boxed Ambients, based on new semantics of communication and mobility. The ambients of NBA are provided with two distinct channels: a local channel to enable the interaction among local processes and an upward channel to allow communications with the enclosing context. The protocol for value exchange across boundaries requires explicit (mutual) actions to 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, the new mobility primitives provide NBA with essentially the same expressive power as BA.

The benefits of the new semantics of mobility and communication are immediately reflected in the expressiveness of the algebraic theory of the calculus, as well as in the simplicity and generality of its typing system. In some respects, we may view NBA as standing to BA in the same way as

SA stands to MA. However, this correspondence is only superficial. The main novelty of SA, with respect to MA, is that the presence of co-capabilities in the former calculus makes it possible to rule out (grave) mobility interferences with a type system. Similarly, NBA removes communication interferences from BA by introducing a two-sided mechanism for value exchange. In this case, however, the benefits of the new semantics are available directly in the untyped setting (cf. Section 5).

There is arguably room for improving NBA's algebraic theory in at least two respects. On one side, as in companion calculi [14,8,9,16], our coinductive characterisation of barbed congruence is rather complex, as it is achieved at the expenses of labelled transitions which effectively bring back quantification over contexts in terms of the process terms occurring in the higher-order labels. Work on simplifying that characterisation would be worthwhile. Also, a richer set of equational laws would be desirable to allow for non-trivial, or at least non-local optimizations of NBA programs. While this may be difficult to achieve in an untyped setting, we expect such equations to become available with the aid of type systems supporting an accurate control of mobility and communication. This is indeed a promising direction as most of the results relative to the typed analysis for BA, notably those developed by two of the authors in [15], can be re-established for NBA with no difficulty.

From a more practical point of view, we argue that NBA provides a promising basis for the design of tractable programming languages based on mobile ambients. The original calculus of MA has long been acknowledged to be a very elegant model for mobility and controlled access to resources. However, the abstraction level at which the mobility and resource access protocols may be expressed in MA is often the source of many difficulties in reasoning about, and implementing ambient programs. With NBA this is rectified by resorting to a different, and higher-level set of core primitives specifically designed with programming in mind. This is distinctively true of the binding mechanism associated with NBA's co-capabilities. On the one hand, this mechanism provides a useful abstraction—the ability to dynamically learn new agent names—which is not available in MA although certainly required in applications. On the other hand, by making this mechanism primitive (hence atomic), NBA provides solid grounds for reasoning on higher-level protocols built on top of it, such as resource assignment control and accounting, or agent authentication.

A final note on implementation. The implementation of ambient calculi has received some attention in the literature [11,25]. These papers have provided solutions to the problems arising in the synchronizations required for mobility. Clearly, an implementation of boxed ambients would benefit from this body of work (cf. [21] for a recent proposal in that direction). On the other side, the semantics of communication poses similar problems. NBA greatly enhances the distributed nature of BA in the precise sense of removing non-determinism between local and non-local communication. A further simplification would result from adopting an asynchronous semantics for the hierarchical exchange of values, in the style suggested by the reductions below:

$$\begin{aligned}\langle\tilde{M}\rangle^{n}.P \mid n[Q] &\longrightarrow P \mid n[\langle\tilde{M}\rangle^{\bar{\hat{\hat{}}}} \mid Q],\\ n[\langle\tilde{M}\rangle^{\hat{\hat{}}}.Q \mid R] &\longrightarrow \langle\tilde{M}\rangle^{\overline{n}} \mid n[Q \mid R]\end{aligned}$$

where $\langle M\rangle^{\bar{\hat{\hat{}}}}$ and $\langle M\rangle^{\overline{n}}$ are new process forms. The exchange of values would then be a purely local operation governed by the following rule for communication

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

where we convene that $\star = \overline{\star}$. Taken together, these reductions provide for a two-fold model of communication that combines synchronous local communication with asynchronous transmission of values across boundaries, a solution that has first been advocated in [7] and has since then gained increasing popularity. Future work include plans on investigating this and other alternatives in the development of an implementation of NBA.

Acknowledgments

The stimulating cricitism and valuable suggestions by the anonymous referees of Information and Computation were of great help in improving the presentation. We also acknowledge the comments from the program committee of FSTTCS'02.

References

[1] S. Arun-Kumar, M. Hennessy, An efficiency preorder for processes, Acta Informatica 29 (1992) 737–760.

[2] M. Boreale, On the expressiveness of internal mobility in name-passing calculi, Theoretical Computer Science 195 (1998) 205–226.

[3] M. Bugliesi, G. Castagna, S. Crafa, Access Control for Mobile Agents: the Calculus of Boxed Ambients, TOPLAS, ACM Transactions on Programming Languages and Systems 26 (1) (2004) 57–124. An extended abstract appeared in TACS'01, LNCS 2215, 2001.

[4] M. Bugliesi, S. Crafa, M. Merro, V. Sassone, Communication interference in mobile boxed ambients, in: FSTTCS'02, 22th Conference on the Foundations of Software Technology and Theoretical Computer Science, Lecture Notes in Computer Science, vol. 2556, Springer, Berlin, 2002, pp. 71–84.

[5] L. Cardelli, A. Gordon Mobile Ambients, Theoretical Computer Science 240 (1) (2000) 177–213, An extended abstract appeared in FoSSaCS'98, LNCS 1378, 1998.

[6] L. Cardelli, A. Gordon, Equational properties for mobile ambients, Mathematical Structures in Computer Science 13 (3) (2003) 371–408, An extended abstract appeared in FoSSaCS'99, LNCS 1578, 1999..

[7] L. Cardelli, Global computing, slides (2000).

[8] G. Castagna, 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, Lecture Notes in Computer Science, vol. 2556, Springer, Berlin, 2002, pp. 85–96.

[9] G. Castagna, J. Vitek, F. Zappa Nardelli, The Seal Calculus, Information and Computation x (x) (2005) xx–xx (to appear). Available from: <`http://www.di.ens.fr/~castagna`>.

[10] S. Crafa, M. Bugliesi, G. Castagna, Information flow security for boxed ambients, in: F-WAN, International Workshop on Foundations of Wide Area Network Computing, ENTCS, vol. 66.3 Elsevier, 2002.

[11] C. Fournet, J.-J. Levy, A. Schmitt, An asynchronous, distributed implementation of mobile ambients, in: IFIP TCS 2000, International Conference on Theoretical Computer Science, Lecture Notes in Computer Science, vol. 1872, Springer, Berlin, 2000, pp. 348–364.

[12] K. Honda, N. Yoshida, On reduction-based process semantics, Theoretical Computer Science 152 (2) (1995) 437–486.

[13] F. Levi, D. Sangiorgi, Mobile safe ambients, TOPLAS, ACM Transactions on Programming Languages and Systems 25 (1) (2003) 1–69, An extended abstract appeared in POPL'00, 27th ACM Symposium on Principles of Programming Languages, 2000.

[14] M. Merro, M. Hennessy, A bisimulation-based semantic theory of Safe Ambients, TOPLAS, ACM Transactions on Programming Languages and Systems x (x) (2005) xx–xx (to appear). An extended abstract appeared in POPL'02, 29th ACM Symposium on Principles of Programming Languages, 2002.

[15] M. Merro, V. Sassone, Typing and subtyping mobility in boxed ambients, in: CONCUR'02, 14th International Conference on Concurrency Theory, Lecture Notes in Computer Science, vol. 2421, Springer, Berlin, 2002, pp. 304–320.

[16] M. Merro, F. Zappa-Nardelli, Bisimulation proof methods for mobile ambients, in: ICALP'03, 30th International Colloquium on Automata, Languages and Programming, Lecture Notes in Computer Science, vol. 2719, Springer, Berlin, 2003, pp. 584–598.

[17] R. Milner, J. Parrow, D. Walker, A calculus of mobile processes, Parts I and II, Information and Computation 100 (1992) 1–77.

[18] R. Milner, D. Sangiorgi, Barbed bisimulation, in: ICALP'92, 19th International Colloquium on Automata, Languages and Programming, Lecture Notes in Computer Science, vol. 623, Springer, Berlin, 1992, pp. 685–695.

[19] R. Milner, Communicating and Mobile Systems: The π-calculus, Cambridge University Press, Cambridge, 1999.

[20] U. Nestmann, What is a 'good' encoding of guarded choice?, Information and Computation 156 (2000) 287–319.

[21] A. Phillips, N. Yoshida, S. Eisenbach, A distributed abstract machine for boxed ambient calculi, in: ESOP'04, 7th European Symposium on Programming, LNCS, vol. 2986, Springer, Berlin, 2004, pp. 155–170.

[22] B. Pierce, U. Nestmann, Decoding choice encodings, Information and Computation 163 (1) (2000) 1–59.

[23] 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.

[24] D. Sangiorgi, Extensionality and intensionality of the ambient logic, in: POPL'01, 28th ACM Symposium on Principles of Programming Languages, ACM Press, 2001, pp. 4–13.

[25] D. Sangiorgi, A. Valente, A distributed abstract machine for Safe Ambients, in: ICALP'01, 28th International Colloquium on Automata, Languages and Programming, Lecture Notes in Computer Science, vol. 2076, Springer, Berlin, 2001, pp. 408–420.

[26] D. Sangiorgi, R. Milner, The problem of "Weak Bisimulation up to", in: CONCUR'92, 3rd International Conference on Concurrency Theory, Lecture Notes in Computer Science, vol. 630, Springer, Berlin, 1992, pp. 32–46.

[27] D. Sangiorgi, D. Walker, The pi-Calculus: A Theory of Mobile Processes, Cambridge University Press, Cambridge, 2001.

[28] D. Sangiorgi, Bisimulation for higher-order process calculi, Information and Computation 131 (2) (1996) 141–178.

[29] J. Vitek, G. Castagna, Seal: a framework for secure mobile computations, in: ICCL Workshop: Internet Programming Languages, Lecture Notes in Computer Science, vol. 1686, Springer, Berlin, 1999, pp. 47–77.

[30] F. Zappa-Nardelli, De la sémantique des processus d'ordre supérieur, PhD thesis, Université Paris 7, 2003.