Bisimulation Proof Methods for Mobile Ambients

Massimo Merro[1] and Francesco Zappa Nardelli[2]

[1] Università di Verona, Italy
[2] LIENS, Paris, France

Abstract. We study the behavioural theory of Cardelli and Gordon's Mobile Ambients. We give an *LTS* based operational semantics, and a labelled *bisimulation* based equivalence that coincides with reduction barbed congruence. We also provide *up-to proof techniques* and prove a set of *algebraic laws*, including the perfect firewall equation.

Introduction

The calculus of *Mobile Ambients* [4], abbreviated MA, has been introduced as a process calculus for describing *mobile agents*. In MA, the term $n[P]$ represents an agent, or *ambient*, named n, executing the code P. The ambient n is a bounded, protected, and (potentially) mobile space where the computation P takes place. In turn P may contain other ambients, may perform (local) *communications*, or may exercise *capabilities*, which allow entry to or exit from named ambients. *Ambient names*, such as n, are used to control access to the ambient's computation space and may be dynamically created as in the π-calculus, [17], using the construct $(\boldsymbol{\nu} n)P$. A *system* in MA consists of a collection of ambients running in parallel where the knowledge of certain ambient names may be restricted.

Behavioural equality is a central idea in process calculi. In this paper we focus on a generalisation of *reduction barbed congruence* of [12]. Reduction barbed congruence is the largest equivalence relation that (i) is a congruence, (ii) preserves, in some sense, the reduction semantics of the language; (iii) preserves *barbs*, some simple observational property of terms.

However, context-based behavioural equalities, such as reduction barbed congruence, suffer from the universal quantification on contexts. Simpler proof techniques are based on *labelled bisimilarities*, [19], whose definitions do not use context quantification. These bisimilarities should imply, or (better) coincide with, reduction barbed congruence [20, 1, 8]. The behaviour of processes is characterised using co-inductive relations defined over a *labelled transition system*, or *LTS*, a collection of relations of the form

$$P \xrightarrow{\alpha} Q.$$

Intuitively, the action α in the judgment $P \xrightarrow{\alpha} Q$ represents some small context P can interact with; if the labelled bisimilarity coincides with reduction barbed congruence then this collection of small contexts, codified as actions, is sufficient to capture all possible interactions that processes can have with arbitrary contexts.

Although the idea of bisimulation is very general and does not rely on the specific syntax of the calculus, the definition of an appropriate notion of bisimilarity for Mobile Ambients revealed harder than expected. The reasons of that can be resumed as follows:

– It is difficult for an ambient n to control interferences that may originate either from other ambients in its environment or from the computation running at n itself, [13].
– Ambient mobility is asynchronous — no permission is required to migrate into an ambient. As noticed in [23], this may cause a *stuttering* phenomenon originated by ambients that may repeatedly enter and exit another ambient. As stuttering cannot be observed, any successful characterisation of reduction barbed congruence should not observe stuttering [23].

– One of the main algebraic laws of MA is the *perfect firewall equation*, [4]:

$$(\boldsymbol{\nu} n)n[P] = \mathbf{0} \qquad \text{for } n \text{ not in } P.$$

If you suppose $P = \texttt{in\_}k.\mathbf{0}$, it is evident that a bisimilarity that wants to capture this law must not observe the movements of *secret ambients*, that is those ambients, like n, whose names are not known by the rest of the system.

In [14], it is introduced a labelled bisimilarity for an "easier" variant of MA, called SAP, equipped with (i) *synchronous mobility*, as in Levi and Sangiorgi's *Safe Ambients* [13], and (ii) *passwords* to exercise control over, and differentiate between, different ambients that may wish to exercise a capability. The main result in [14] is the characterisation of reduction barbed congruence in terms of the labelled bisimilarity. The result holds only in SAP and crucially relies on the two features (i) and (ii) mentioned above.

The current paper is the natural continuation of [14] where, now, we tackle the original problem: *to provide bisimulation proof methods for Mobile Ambients.*

Contribution First of all, as in the Distributed π-calculus [11], we rewrite the syntax of MA in two levels: *processes* and *systems*. This is because we are interested in studying systems rather than processes. So, our behavioural equalities are defined over systems. This little expedient allows us to (i) focus on higher-order actions, where movement of code is involved, and (ii) to model stuttering in terms of standard τ-actions.

We introduce a new labelled transition system for MA that is used to define a labelled bisimilarity over systems. The resulting bisimilarity, denoted by $\approx$, is in *late* style. The definition of $\approx$ reminds us that one of the asynchronous bisimilarity found in [1]. Indeed, as for inputs in asynchronous π, our bisimilarity does not observe the movements of secret ambients.

We prove that the relation $\approx$ completely characterises reduction barbed congruence over systems. Then, we enhance our proof methods by defining two *up-to proof techniques*, along the lines of [18, 22, 25]. More precisely, we develop both *up-to expansion* and *up-to context* proof techniques and prove their soundness. We are not aware of other forms of up-to proof techniques for higher-order calculi. Finally, we apply our bisimulation proof methods to prove a collection of *algebraic laws* (including the perfect firewall equation); we also prove the correctness of the protocol, introduced in [4], for controlling access through a firewall.

The treatment of *communication* is only outlined; however, in [15], the above mentioned results are smoothly extended to the calculus with communication.

The paper ends with a comparison with related work. For lack of space proofs are sketched or omitted; full proofs can be found in [15].

1 Mobile Ambients in Two Levels

In Table 1 we report the syntax of MA, where $\mathbf{N}$ denotes an infinite set of names. Unlike other definitions of MA in the literature, our syntax is defined in a two-level structure, a lower one for *processes*, and an upper one for *systems*.

The syntax for processes is standard, [4], except for replication that is replaced by replicated prefixing, $!C.P$. As in the π-calculus, replicated prefixing allows us to derive a simpler LTS; theory and results in this paper could be easily adapted to a calculus with full replication. A system is a collection of ambients running in parallel, where the knowledge of certain ambient names may be restricted among two or more ambients. We use a number of notational conventions. Parallel composition has the lowest precedence among the operators. The process $C.C'.P$ is read

Table 1 The Mobile Ambients in Two Levels: Syntax and Reduction Rules

Names: $n, h, \ldots \in \mathbf{N}$

Systems:

$$\begin{array}{rcll} M, N & ::= & \mathbf{0} & \text{nil system} \\ & | & M_1 \mid M_2 & \text{parallel} \\ & | & (\boldsymbol{\nu} n)M & \text{restriction} \\ & | & n[M] & \text{ambient} \end{array}$$

Processes:

$$\begin{array}{rcll} P, Q, R & ::= & \mathbf{0} & \text{nil process} \\ & | & P_1 \mid P_2 & \text{parallel} \\ & | & (\boldsymbol{\nu} n)P & \text{restriction} \\ & | & n[P] & \text{ambient} \\ & | & C.P & \text{prefixing} \\ & | & !C.P & \text{replication} \end{array}$$

Capabilities:

$$\begin{array}{rcll} C & ::= & \mathtt{in\_}n & \text{may enter into } n \\ & | & \mathtt{out\_}n & \text{may exit out of } n \\ & | & \mathtt{open\_}n & \text{may open } n \end{array}$$

Reduction Rules:

$$n[\mathtt{in\_}m.P \mid Q] \mid m[R] \rightarrow m[n[P \mid Q] \mid R] \qquad m[n[\mathtt{out\_}m.P \mid Q] \mid R] \rightarrow n[P \mid Q] \mid m[R]$$

$$\mathtt{open\_}n.P \mid n[Q] \rightarrow P \mid Q \qquad P \equiv Q \quad Q \rightarrow R \quad R \equiv S \text{ implies } P \rightarrow S$$

as $C.(C'.P)$. We omit trailing dead processes, writing C for $C.\mathbf{0}$, and $n[]$ for $n[\mathbf{0}]$. Restriction $(\boldsymbol{\nu} n)P$ acts as binder for name n, and the set of *free names* of P, $\mathrm{fn}(P)$, is defined accordingly.

A *static context* is a process context where the hole does not appear under a prefix or a replication. The dynamics of the calculus is specified by a *reduction relation*, $\rightarrow$, which is the least relation over processes closed under static contexts and satisfying the rules in Table 1. As systems are processes with a special structure, the rules of Table 1 also describe the evolution of systems. The *reduction semantics* relies on an auxiliary relation called *structural congruence* that brings the participants of a potential interaction into contiguous positions. We refer to [4] for the definition of structural congruence, $\equiv$. It is easy to check that systems always reduce to systems.

We introduce a basic equivalence by considering natural, desirable, properties. We choose a generalisation of the *reduction barbed congruence*, [12], a *contextual*, *reduction closed*, and *barb preserving* equivalence relation. We now explain what these properties mean.

A *system context* is a context generated by the following grammar:

$$C[-] \quad ::= \quad - \quad \mid \quad C[-] \mid M \quad \mid \quad (\boldsymbol{\nu} n)C[-] \quad \mid \quad n[C[-] \mid P]$$

where M is an arbitrary system, and P is an arbitrary process. A relation $\mathcal{R}$ over systems is *contextual* if $M \; \mathcal{R} \; N$ implies $C[M] \; \mathcal{R} \; C[N]$ for all system contexts $C[-]$.

A relation $\mathcal{R}$ over systems is *reduction closed* if whenever $M \; \mathcal{R} \; N$ and $M \rightarrow M'$ there is some N' such that $N \rightarrow^* N'$ and $M' \; \mathcal{R} \; N'$, where $\rightarrow^*$ denotes the reflexive and transitive closure of $\rightarrow$.

In Mobile Ambients the observation predicate $M \downarrow_n$ denotes the possibility of the system M interacting with the environment via the ambient n. We write $M \downarrow_n$ if $M \equiv (\boldsymbol{\nu} \tilde{m})(n[P] \mid M')$ with $n \notin \{\tilde{m}\}$. We write $M \Downarrow_n$ if there exists M' such that $M \rightarrow^* M'$ and $M' \downarrow_n$. A relation $\mathcal{R}$ over systems is *barb preserving* if $M \; \mathcal{R} \; N$ and $M \downarrow_n$ implies $N \Downarrow_n$.

Definition 1 (Reduction barbed congruence). *Reduction barbed congruence, written $\cong$, is the largest symmetric relation over systems which is reduction closed, contextual, and barb preserving.*

2 A Labelled Transition Semantics

In our language, the prefixes C give rise to transitions of the form $P \xrightarrow{C} Q$; for example we have

$$\mathtt{in\_}n.P_1 \mid P_2 \xrightarrow{\mathtt{in\_}n} P_1 \mid P_2.$$

However, similarly to [14], capabilities induce different and more complicated actions. The LTS is defined over processes, although in the labelled bisimilarity we only consider actions going from systems to systems. We make a distinction between *pre-actions* and *env-actions*: the former denote the possibility to exercise certain capabilities whereas the latter model the interaction of a system with its environment. As usual, we also have τ-*actions* to model internal computations. Only env-actions and τ-actions model the evolution of a system at run-time.

The pre-actions, defined in Table 3, are of the form $P \xrightarrow{\boldsymbol{\pi}} O$ where the ranges of π and of O, the *outcomes*, are reported in Table 2. An outcome may be a simple process Q, if for example π is a prefix of the language, or a *concretion*, of the form $(\boldsymbol{\nu}\tilde{m})\langle P\rangle Q$, when an ambient boundary is somehow involved. In this case, intuitively, P represents the part of the system affected by the action while Q is not, and $\tilde{m}$ is the set of private names shared by P and Q. We adopt the convention that if K is the concretion $(\boldsymbol{\nu}\tilde{m})\langle P\rangle Q$, then $(\boldsymbol{\nu}r)K$ is a shorthand for $(\boldsymbol{\nu}\tilde{m})\langle P\rangle(\boldsymbol{\nu}r)Q$, if $r \notin \mathrm{fn}(P)$, and the concretion $(\boldsymbol{\nu}r\tilde{m})\langle P\rangle Q$ otherwise. We have a similar convention for the rule (π Par): $K \mid R$ is defined to be the concretion $(\boldsymbol{\nu}\tilde{m})\langle P\rangle(Q \mid R)$, where $\tilde{m}$ are chosen, using α-conversion if necessary, so that $\mathrm{fn}(R) \cap \{\tilde{m}\} = \emptyset$. Moreover, $(\boldsymbol{\nu}\tilde{m})\langle P\rangle(\mathbf{0} \mid R)$ is abbreviated by $(\boldsymbol{\nu}\tilde{m})\langle P\rangle R$.

The τ-actions, defined in Table 4, model the internal evolution of processes. The env-actions, defined in Table 5, are of the form $M \xrightarrow{\boldsymbol{\sigma}} M'$, where the range of σ is given in Table 2. In practice, env-actions turn concretions into running systems by explicitly introducing the environment's ambient interacting with the process being considered. The content of this ambient is instantiated later, in the bisimilarity, with a process. For convenience, we extend the syntax of processes with the special process $\circ$ to pinpoint those ambients whose content will be instantiated later. The process $\circ$ does not reduce: it is simply a placeholder. Note that, unlike pre-actions and τ-actions, env-actions do not have structural rules; this is because env-actions are supposed to be performed by systems that can directly interact with the environment.

We call *actions* the set of env-actions extended with τ. As our bisimilarity will be defined over systems, we will only consider actions (and not pre-actions) in its definition.

Proposition 1. *If T is a system (resp. a process), and $T \xrightarrow{\boldsymbol{\alpha}} T'$, then T' is a system (resp. a process), possibly containing the special process $\circ$.*

We explain the rules induced by the the prefix `in`, the *immigration* of ambients. A typical example of an ambient m migrating into an ambient n follows:

$$(\boldsymbol{\nu}m)(m[\,\mathtt{in\_}n.P_1 \mid P_2\,] \mid M) \mid n[Q] \;\twoheadrightarrow\; (\boldsymbol{\nu}m)(M \mid n[\,m[\,P_1 \mid P_2\,] \mid Q])$$

The driving force behind the migration is the activation of the prefix $\mathtt{in\_}n$, within the ambient m. It induces a capability in the ambient m to migrate into n, that we formalise as a new action $\mathtt{enter\_}n$. Thus, an application of (π Enter) gives

$$m[\mathtt{in\_}n.P_1 \mid P_2] \xrightarrow{\mathtt{enter\_}n} \langle m[P_1 \mid P_2]\rangle\mathbf{0}$$

and more generally, using the structural rules (π Res) and (π Par),

$$(\boldsymbol{\nu}m)(m[\mathtt{in\_}n.P_1 \mid P_2] \mid M) \xrightarrow{\mathtt{enter\_}n} (\boldsymbol{\nu}m)\langle m[P_1 \mid P_2]\rangle M.$$

This means that the ambient $m[\mathtt{in\_}n.P_1 \mid P_2]$ has the capability to enter an ambient n; if the capability is exercised, the ambient $m[P_1 \mid P_2]$ will enter n, while M will be the residual at the original location. Of course, the action can only be executed if there is an ambient n in parallel. The rule (π Amb) allows to check for the presence of ambients. So we have

$$n[Q] \xrightarrow{\mathtt{amb\_}n} \langle Q\rangle\mathbf{0}.$$

Table 2 Pre-actions, Env-actions, Actions, Concretions, and Outcomes

$$\begin{array}{llll} \textit{Pre-actions:} & \pi & ::= & \texttt{in\_}n \mid \texttt{out\_}n \mid \texttt{open\_}n \mid \texttt{enter\_}n \mid \texttt{amb\_}n \mid \texttt{exit\_}n \\ \textit{Env-actions:} & \sigma & ::= & k.\texttt{enter\_}n \mid k.\texttt{exit\_}n \mid *.\texttt{enter\_}n \mid *.\texttt{exit\_}n \mid n.\overline{\texttt{enter}}\texttt{\_}k \mid k.\texttt{open\_}n \\ \textit{Outcomes:} & O & ::= & P \mid K \\ \textit{Concretions:} & K & ::= & (\boldsymbol{\nu}\tilde{m})\langle P\rangle Q \\ \textit{Actions:} & \alpha & ::= & \sigma \cup \tau \end{array}$$

Table 3 Labelled Transition System - Pre-actions

$$(\pi\ \text{Pfx})\ \frac{-}{\pi.P \xrightarrow{\pi} P} \qquad (\pi\ \text{Repl Pfx})\ \frac{-}{!\pi.P \xrightarrow{\pi} P \mid !\pi.P} \qquad (\pi\ \text{Exit})\ \frac{P \xrightarrow{\texttt{out\_}n} P_1}{m[P] \xrightarrow{\texttt{exit\_}n} \langle m[P_1]\rangle\mathbf{0}}$$

$$(\pi\ \text{Amb})\ \frac{-}{n[P] \xrightarrow{\texttt{amb\_}n} \langle P\rangle\mathbf{0}} \qquad (\pi\ \text{Enter})\ \frac{P \xrightarrow{\texttt{in\_}n} P_1}{m[P] \xrightarrow{\texttt{enter\_}n} \langle m[P_1]\rangle\mathbf{0}} \qquad (\pi\ \text{Res})\ \frac{P \xrightarrow{\pi} O \quad n \notin \text{fn}(\pi)}{(\boldsymbol{\nu}n)P \xrightarrow{\pi} (\boldsymbol{\nu}n)O}$$

$$(\pi\ \text{Par})\ \frac{P \xrightarrow{\pi} O}{\begin{array}{c} P \mid Q \xrightarrow{\pi} O \mid Q \\ Q \mid P \xrightarrow{\pi} Q \mid O \end{array}}$$

Table 4 Labelled Transition System - τ-actions

$$(\tau\ \text{Enter})\ \frac{P \xrightarrow{\texttt{enter\_}n} (\boldsymbol{\nu}\tilde{p})\langle P_1\rangle P_2 \quad Q \xrightarrow{\texttt{amb\_}n} (\boldsymbol{\nu}\tilde{q})\langle Q_1\rangle Q_2{}^{(*)}}{\begin{array}{c} P \mid Q \xrightarrow{\tau} (\boldsymbol{\nu}\tilde{p})(\boldsymbol{\nu}\tilde{q})(n[P_1 \mid Q_1] \mid P_2 \mid Q_2) \\ Q \mid P \xrightarrow{\tau} (\boldsymbol{\nu}\tilde{q})(\boldsymbol{\nu}\tilde{p})(n[Q_1 \mid P_1] \mid Q_2 \mid P_2) \end{array}} \qquad (\tau\ \text{Par})\ \frac{P \xrightarrow{\tau} P'}{\begin{array}{c} P \mid Q \xrightarrow{\tau} P' \mid Q \\ Q \mid P \xrightarrow{\tau} Q \mid P' \end{array}}$$

$$(\tau\ \text{Exit})\ \frac{P \xrightarrow{\texttt{exit\_}n} (\boldsymbol{\nu}\tilde{m})\langle k[P_1]\rangle P_2}{n[P] \xrightarrow{\tau} (\boldsymbol{\nu}\tilde{m})(k[P_1] \mid n[P_2])} \qquad (\tau\ \text{Amb})\ \frac{P \xrightarrow{\tau} Q}{n[P] \xrightarrow{\tau} n[Q]}$$

$$(\tau\ \text{Open})\ \frac{P \xrightarrow{\texttt{open\_}n} P_1 \quad Q \xrightarrow{\texttt{amb\_}n} (\boldsymbol{\nu}\tilde{m})\langle Q_1\rangle Q_2}{\begin{array}{c} P \mid Q \xrightarrow{\tau} P_1 \mid (\boldsymbol{\nu}\tilde{m})(Q_1 \mid Q_2) \\ Q \mid P \xrightarrow{\tau} (\boldsymbol{\nu}\tilde{m})(Q_1 \mid Q_2) \mid P_1 \end{array}} \qquad (\tau\ \text{Res})\ \frac{P \xrightarrow{\tau} P'}{(\boldsymbol{\nu}n)P \xrightarrow{\tau} (\boldsymbol{\nu}n)P'}$$

(*) We require $((\text{fn}(P_1) \cup \text{fn}(P_2)) \cap \{\tilde{q}\}) = ((\text{fn}(Q_1) \cup \text{fn}(Q_2)) \cap \{\tilde{p}\}) = \emptyset$

Table 5 Labelled Transition System - Env-actions

$$(\text{Enter})\ \frac{P \xrightarrow{\texttt{enter\_}n} (\boldsymbol{\nu}\tilde{m})\langle k[P_1]\rangle P_2{}^{(\dagger)}}{P \xrightarrow{k.\texttt{enter\_}n} (\boldsymbol{\nu}\tilde{m})(n[\circ \mid k[P_1]] \mid P_2)} \qquad (\text{Exit})\ \frac{P \xrightarrow{\texttt{exit\_}n} (\boldsymbol{\nu}\tilde{m})\langle k[P_1]\rangle P_2{}^{(\dagger)}}{P \xrightarrow{k.\texttt{exit\_}n} (\boldsymbol{\nu}\tilde{m})(k[P_1] \mid n[\circ \mid P_2])}$$

$$(\text{Co-Enter})\ \frac{P \xrightarrow{\texttt{amb\_}n} (\boldsymbol{\nu}\tilde{m})\langle P_1\rangle P_2{}^{(\dagger)}}{P \xrightarrow{n.\overline{\texttt{enter}}\texttt{\_}k} (\boldsymbol{\nu}\tilde{m})(n[P_1 \mid k[\circ]] \mid P_2)} \qquad (\text{Open})\ \frac{P \xrightarrow{\texttt{amb\_}n} (\boldsymbol{\nu}\tilde{m})\langle P_1\rangle P_2}{P \xrightarrow{k.\texttt{open\_}n} k[\circ \mid (\boldsymbol{\nu}\tilde{m})(P_1 \mid P_2)]}$$

$$(\text{Enter Shh})\ \frac{P \xrightarrow{\texttt{enter\_}n} (\boldsymbol{\nu}\tilde{m})\langle k[P_1]\rangle P_2{}^{(\ddagger)}}{P \xrightarrow{*.\texttt{enter\_}n} (\boldsymbol{\nu}\tilde{m})(n[\circ \mid k[P_1]] \mid P_2)} \qquad (\text{Exit Shh})\ \frac{P \xrightarrow{\texttt{exit\_}n} (\boldsymbol{\nu}\tilde{m})\langle k[P_1]\rangle P_2{}^{(\ddagger)}}{P \xrightarrow{*.\texttt{exit\_}n} (\boldsymbol{\nu}\tilde{m})(k[P_1] \mid n[\circ \mid P_2])}$$

(†) We require $k \notin \tilde{m}$. (‡) We require $k \neq n$ and $k \in \tilde{m}$

Here, the concretion $\langle Q\rangle\mathbf{0}$ says that Q is in n, while $\mathbf{0}$ is outside. Finally, the communication (τ Enter) allows these two complementary actions to occur simultaneously, executing the migration of the ambient $m[P_1 \mid P_2]$ from its current computational space into the ambient n, giving rise to the original move above:

$$(\boldsymbol{\nu} m)(m[\,\texttt{in\_}n.P_1 \mid P_2\,] \mid M) \mid n[Q] \xrightarrow{\tau} (\boldsymbol{\nu} m)(M \mid n[\,m[\,P_1 \mid P_2\,] \mid Q]).$$

Env-actions model the interaction of mobile agents with their environment. For instance, using the rule (Enter Shh), we derive from

$$(\boldsymbol{\nu} m)(m[\texttt{in\_}n.P_1 \mid P_2] \mid M) \xrightarrow{\texttt{enter\_}n} (\boldsymbol{\nu} m)\langle m[P_1 \mid P_2]\rangle M$$

the transition

$$(\boldsymbol{\nu} m)(m[\texttt{in\_}n.P_1 \mid P_2] \mid M) \xrightarrow{*.\texttt{enter\_}n} (\boldsymbol{\nu} m)(n[\circ \mid m[P_1 \mid P_2]] \mid M).$$

This transition denotes a private (*secret*) ambient entering an ambient n provided by the environment. The computation running at n can be specified later by instantiating the placeholder $\circ$. Had the ambient name m not been restricted, we would have used the rule (Enter) to derive

$$m[\texttt{in\_}n.P_1 \mid P_2] \mid M \xrightarrow{m.\texttt{enter\_}n} n[\circ \mid m[P_1 \mid P_2]] \mid M$$

to model a public ambient m that enters an ambient n provided by the environment.

The rules for *emigration* and *opening* follow the same lines. Finally, whenever a system offers a public ambient n at top-level, a context can interact with the system by providing an ambient entering n. The rule Co-Enter captures this interaction between system and environment.

The LTS based semantics coincides with the reduction semantics of Section 1.

Theorem 1. *If $P \xrightarrow{\tau} P'$ then $P \rightarrow P'$. If $P \rightarrow P'$ then $P \xrightarrow{\tau}\equiv P'$.*

Remark 1. From the result above, it is easy to establish that if $M \cong N$ then (i) $M \Downarrow n$ iff $N \Downarrow n$ and (ii) $M \Longrightarrow M'$ implies there is N' such that $N \Longrightarrow N'$ and $M' \cong N'$. In the sequel we will use these properties without comment.

3 Characterising Reduction Barbed Congruence

In this section we define a labelled bisimilarity for MA that completely characterises reduction barbed congruence.

Since we are interested in *weak bisimilarities*, that abstract over τ-actions, we introduce the notion of weak action. The definition is standard: $\Longrightarrow$ denotes the reflexive and transitive closure of $\xrightarrow{\tau}$; $\stackrel{\alpha}{\Longrightarrow}$ denotes $\Longrightarrow\xrightarrow{\alpha}\Longrightarrow$; $\stackrel{\hat{\alpha}}{\Longrightarrow}$ denotes $\Longrightarrow$ if $\alpha = \tau$ and $\stackrel{\alpha}{\Longrightarrow}$ otherwise.

In the previous section we said that actions (and more precisely env-actions) introduce a special process $\circ$ to pinpoint those ambients whose content will be specified in the bisimilarity. The $\bullet$ operator instantiates the placeholder with a process, as defined below.

Definition 2. *Let T and T_i be either systems or processes. Then, for a process P, we define:*

$$\begin{array}{lll}
\mathbf{0} \bullet P \stackrel{\text{def}}{=} \mathbf{0} & (T_1 \mid T_2) \bullet P \stackrel{\text{def}}{=} (T_1 \bullet P) \mid (T_2 \bullet P) & \circ \bullet P \stackrel{\text{def}}{=} P \\
n[R] \bullet P \stackrel{\text{def}}{=} n[R \bullet P] & (\boldsymbol{\nu} n)T \bullet P \stackrel{\text{def}}{=} (\boldsymbol{\nu} n)(T \bullet P) \text{ if } n \notin \text{fn}(P) & \\
!C.R \bullet P \stackrel{\text{def}}{=} !C.(R \bullet P) & C.R \bullet P \stackrel{\text{def}}{=} C.(R \bullet P) &
\end{array}$$

Everything is in place to define our bisimilarity over systems.

Definition 3 (Bisimilarity). *A symmetric relation $\mathcal{R}$ is a* bisimulation *if $M \mathrel{\mathcal{R}} N$ implies:*

- *if $M \xrightarrow{\alpha} M'$, $\alpha \notin \{*.\mathtt{enter\_}n, *.\mathtt{exit\_}n\}$, then there is a system N' such that $N \stackrel{\hat{\alpha}}{\Longrightarrow} N'$ and for all processes P it holds $M' \bullet P \mathrel{\mathcal{R}} N' \bullet P$;*
- *if $M \xrightarrow{*.\mathtt{enter\_}n} M'$ then there is a system N' such that $N \mid n[\circ] \Longrightarrow N'$ and for all processes P it holds $M' \bullet P \mathrel{\mathcal{R}} N' \bullet P$;*
- *if $M \xrightarrow{*.\mathtt{exit\_}n} M'$ then there is a system N' such that $n[\circ \mid N] \Longrightarrow N'$ and for all processes P it holds $M' \bullet P \mathrel{\mathcal{R}} N' \bullet P$.*

Systems M and N are bisimilar*, written $M \approx N$, if $M \mathrel{\mathcal{R}} N$ for some bisimulation $\mathcal{R}$.*

The bisimilarity above has a universal quantification over the process P provided by the environment. This process instantiates the special process $\circ$ generated via env-actions. The bisimilarity is defined in a *late* style as the existential quantification precedes the universal one. Another possibility would be to define the bisimilarity in *early* style where the universal quantification over the environment's contribution P precedes that over the derivative N'. In [15] we prove that, as in HOπ [21], late and early bisimilarity coincide.

Finally, notice that actions $*.\mathtt{enter\_}n$ and $*.\mathtt{exit\_}n$ are treated apart asking for weaker matching requirements. This is because both actions are not observable. Somehow, this is very similar to what happens with input actions in the asynchronous π-calculus [1].

We prove that $\approx$ is a proof techniques for reduction barbed congruence.

Theorem 2. *Bisimilarity is contextual and is an equivalence relation*

Notice that when proving transitivity we use the fact that $\approx$ is preserved by parallel composition and ambient nesting. These two properties do not rely on the transitivity of $\approx$, and are necessary to deal with the env-actions $*.\mathtt{enter\_}n$ and $*.\mathtt{exit\_}n$.

We can finally state our soundness result.

Theorem 3 (Soundness). *Bisimilarity is contained in reduction barbed congruence.*

Proof By Theorem 2 and the fact that bisimilarity is reduction closed and barb-preserving. $\Box$

We now prove that $\approx$ is more than a proof techniques. It actually characterises the reduction barbed congruence. The main challenge here is to design the contexts capable to observe our visible actions. The definition of these contexts, $C_\alpha[-]$, for every visible action α, is given in Table 6. The special ambient name done is used as *fresh* barb to signal the consumption of actions.

To prove our characterisation result it suffices to show that reduction barbed congruence is contained in bisimilarity. At this end we must prove a correspondence between visible actions α and their corresponding contexts $C_\alpha[-]$.

The following lemma says that the defining contexts are sound, that is, they can successfully mimic the execution of visible actions.

Lemma 1. *Let M be a system, and let $\alpha \in \{k.\mathtt{enter\_}n, k.\mathtt{exit\_}n, n.\overline{\mathtt{enter}}\mathtt{\_}k, k.\mathtt{open\_}n\}$. For all processes P, if $M \xrightarrow{\alpha} M'$, then $C_\alpha[M] \bullet P \Longrightarrow\cong M' \bullet P \mid \mathsf{done}[\,]$.*

To complete the correspondence proof between actions α and their contexts $C_\alpha[-]$, we have to prove the converse of Lemma 1, formalised in Lemma 2. Such result requires a few technical definitions given in Table 7. The symbol $\oplus$ denotes a form of internal choice, whereas the context $\mathrm{SPY}_\alpha\langle i, j, -\rangle$ is a technical tool to guarantee that the process P provided by the environment does not perform any action. This is essential to the proof of the completeness result because it guarantees that the contribution of P is the same on both sides. The ability of $\mathrm{SPY}_\alpha\langle i, j, P\rangle$ to "spy" on P stems from the fact that one of the two fresh barbs i and j is lost when P performs any action.

Table 6 Contexts for visible actions

$\alpha = k.\mathtt{enter}\_n$	$C_\alpha[-] = n[\circ \mid \mathsf{done}[\mathtt{in}\_k.\mathtt{out}\_k.\mathtt{out}\_n]] \mid -$
$\alpha = k.\mathtt{exit}\_n$	$C_\alpha[-] = (\boldsymbol{\nu} a)a[\mathtt{in}\_k.\mathtt{out}\_k.\mathsf{done}[\mathtt{out}\_a]] \mid n[\circ \mid -]$
$\alpha = n.\overline{\mathtt{enter}}\_k$	$C_\alpha[-] = (\boldsymbol{\nu} a)a[\mathtt{in}\_n.k[\mathtt{out}\_a.(\circ \mid (\boldsymbol{\nu} b)b[\mathtt{out}\_k.\mathtt{out}\_n.\mathsf{done}[\mathtt{out}\_b]])]] \mid -$
$\alpha = k.\mathtt{open}\_n$	$C_\alpha[-] = k[\circ \mid (\boldsymbol{\nu} a,b)(\mathtt{open}\_b.\mathtt{open}\_a.\mathsf{done}[\mathtt{out}\_k] \mid a[- \mid \mathtt{open}\_n.b[\mathtt{out}\_a]])]$
	where a and b are fresh.

Table 7 Auxiliary contexts and processes

$$-_1 \oplus -_2 \quad = (\boldsymbol{\nu} o)(o[\,] \mid \mathtt{open}\_o.-_1 \mid \mathtt{open}\_o.-_2)$$

$$\mathrm{SPY}_\alpha\langle i,j,-\rangle = (i[\mathtt{out}\_n] \mid -) \oplus (j[\mathtt{out}\_n] \mid -)$$
$$\text{if } \alpha \in \{k.\mathtt{enter}\_n, k.\mathtt{exit}\_n, k.\mathtt{open}\_n, *.\mathtt{enter}\_n, *.\mathtt{exit}\_n\}$$

$$\mathrm{SPY}_\alpha\langle i,j,-\rangle = (i[\mathtt{out}\_k.\mathtt{out}\_n] \mid -) \oplus (j[\mathtt{out}\_k.\mathtt{out}\_n] \mid -) \text{ if } \alpha \in \{n.\overline{\mathtt{enter}}\_k\}$$

Lemma 2. *Let M be a system, let $\alpha \in \{k.\mathtt{enter}\_n, k.\mathtt{exit}\_n, n.\overline{\mathtt{enter}}\_k, k.\mathtt{open}\_n\}$, and let i,j be fresh names for M. For all processes P with $\{i,j\} \cap \mathrm{fn}(P) = \emptyset$, if $C_\alpha[M] \bullet \mathrm{SPY}_\alpha\langle i,j,P\rangle \Rightarrow\cong O \mid$ $\mathsf{done}[\,]$ and $O \Downarrow_{i,j}$, then there exists a system M' such that $O \cong M' \bullet \mathrm{SPY}_\alpha\langle i,j,P\rangle$ and $M \stackrel{\alpha}{\Longrightarrow} M'$.*

Theorem 4 (Completeness). *Reduction barbed congruence is contained in bisimilarity.*

Proof [Sketch] We prove that the relation $\mathcal{R} = \{(M,N) \mid M \cong N\}$ is a late bisimulation. Suppose $M \mathrel{\mathcal{R}} N$ and $M \xrightarrow{\alpha} M'$. Suppose also $\alpha \in \{k.\mathtt{enter}\_n, k.\mathtt{exit}\_n, n.\overline{\mathtt{enter}}\_k, k.\mathtt{open}\_n\}$. We must find a system N' such that $N \stackrel{\alpha}{\Longrightarrow} N'$ and for all P, $M' \bullet P \cong N' \bullet P$.

The idea of the proof is to use a particular context which mimics the effect of the action α, and also allows us to subsequentely compare the residuals of the two systems. This context has the form

$$D_\alpha\langle P\rangle[-] \;=\; (C_\alpha[-] \mid \mathsf{Flip}) \bullet \mathrm{SPY}_\alpha\langle i,j,P\rangle$$

where $C_\alpha[-]$ are the contexts in Table 6 and Flip is the system:

$$(\boldsymbol{\nu} k)k[\mathtt{in}\_\mathsf{done}.\mathtt{out}\_\mathsf{done}.(\mathsf{succ}[\mathtt{out}\_k] \oplus \mathsf{fail}[\mathtt{out}\_k])]$$

with succ and fail fresh names. Intuitively, the existence of the fresh barb fail indicates that the action α has not yet happened, whereas the presence of succ together with the absence of fail ensures that the action α has been performed, and has been reported via done.

As $\cong$ is contextual, $M \cong N$ implies that, for all processes P, it holds $D_\alpha\langle P\rangle[M] \;\cong\; D_\alpha\langle P\rangle[N]$. By Lemma 1, and by inspecting the reduction of the Flip process, we observe that:

$$D_\alpha\langle P\rangle[M] \;\;\Rightarrow\cong\;\; M' \bullet \mathrm{SPY}_\alpha\langle i,j,P\rangle \mid \mathsf{done}[\,] \mid \mathsf{Flip} \;\;\Rightarrow\cong\;\; M' \bullet \mathrm{SPY}_\alpha\langle i,j,P\rangle \mid \mathsf{done}[\,] \mid \mathsf{succ}[\,]$$

where $M' \bullet \mathrm{SPY}_\alpha\langle i,j,P\rangle \mid \mathsf{done}[\,] \mid \mathsf{succ}[\,] \Downarrow_{i,j,\mathsf{succ}} \not\Downarrow_{\mathsf{fail}}$. Call this outcome O_1. This reduction must be matched by a corresponding reduction $D_\alpha\langle P\rangle[N] \;\Rightarrow\; O_2$, where $O_1 \cong O_2$. However, the possible matching reductions are constrained by the barbs of O_1, because it must hold $O_2 \Downarrow_{i,j,\mathsf{succ}} \not\Downarrow_{\mathsf{fail}}$.

As $O_2 \Downarrow_{\mathsf{succ}} \not\Downarrow_{\mathsf{fail}}$, it must be $O_2 \cong \hat{N} \mid \mathsf{done}[\,] \mid \mathsf{succ}[\,]$, for some systems $\hat{N}$. As $O_2 \Downarrow_{i,j}$, the previous observation can be combined with Lemma 2 to derive the existence of a system (over the extended process syntax) N' such that $\hat{N} \cong N' \bullet \mathrm{SPY}_\alpha\langle i,j,P\rangle$ and a weak action $N \stackrel{\alpha}{\Longrightarrow} N'$.

To conclude we must establish that for all P, it holds $M' \bullet P \;\cong\; N' \bullet P$. As barbed congruence is preserved by restriction, we have $(\boldsymbol{\nu}\mathsf{done},\mathsf{succ})O_1 \cong (\boldsymbol{\nu}\mathsf{done},\mathsf{succ})O_2$. As $(\boldsymbol{\nu}\mathsf{done})\mathsf{done}[\,] \cong$ $(\boldsymbol{\nu}\mathsf{succ})\mathsf{succ}[\,] \cong \mathbf{0}$, it follows that $M' \bullet \mathrm{SPY}_\alpha\langle i,j,P\rangle \;\cong\; N' \bullet \mathrm{SPY}_\alpha\langle i,j,P\rangle$. Again, $\cong$ is preserved by restriction and $(\boldsymbol{\nu} i,j)\mathrm{SPY}_\alpha\langle i,j,P\rangle \cong P$. So, we can finally derive $M' \bullet P \;\; \mathcal{R} \;\; N' \bullet P$, for all processes P.

To complete the proof we need to consider the actions $*.\mathtt{enter\_}n$ and $*.\mathtt{exit\_}n$: as they are not observable, these cases are much easier. $\square$

By Theorems 3 and 4 we conclude that *bisimilarity and reduction barbed congruence coincide.*

Synchronous communication of capabilities can be added to MA: the *output process*, $\langle E\rangle.P$, outputs the message E and then continues as P, and the input process $(x).Q$ receives a message and binds it to x in Q, which then executes. As discussed in [28, 23], synchrony is not unrealistic because communication in MA is always local. Our LTS needs to be extended with rules analogues to the rules that deal with communication in [14]. The proof of Theorem 1 can be easily completed. More interestingly, in our framework, communication capabilities cannot be observed at top-level: this in turn implies that our bisimulations can be applied to the extended calculus, and all the results of Section 3 and Section 4 hold without modifications.

4 Up-to Proof Techniques

In this section we adapt well-known *up-to* proof techniques [18, 22] to our setting. These techniques allow us to reduce the size of the relation to exhibit to prove that two processes are bisimilar. We focus on the *up-to expansion* [24] and the *up-to context* techniques [22]. As in π-calculus, these can be merged: for lack of space we only report the resulting technique.

Roughly, the expansion, written $\lesssim$, is an asymmetric variant of the bisimilarity that allows us to count the number of silent moves performed by a process.

Definition 4 (Expansion). *A relation $\mathcal{R}$ over systems is an* expansion *if $M \mathrel{\mathcal{R}} N$ implies:*

- *if $M \xrightarrow{\alpha} M'$, $\alpha \notin \{*.\mathtt{enter\_}n, *.\mathtt{exit\_}n\}$, then there exists a system N' such that $N \stackrel{\hat{\alpha}}{\Longrightarrow} N'$ and for all processes P it holds $M' \bullet P \mathrel{\mathcal{R}} N' \bullet P$;*
- *if $M \xrightarrow{*.\mathtt{enter\_}n} M'$ then there is a system N' such that $N \mid n[\circ\,] \Rightarrow N'$ and for all processes P it holds $M' \bullet P \mathrel{\mathcal{R}} N' \bullet P$;*
- *if $M \xrightarrow{*.\mathtt{exit\_}n} M'$ then there is a system N' such that $n[\circ \mid N] \Rightarrow N'$ and for all processes P it holds $M' \bullet P \mathrel{\mathcal{R}} N' \bullet P$;*
- *if $N \xrightarrow{\alpha} N'$, $\alpha \notin \{*.\mathtt{enter\_}n, *.\mathtt{exit\_}n\}$, then there exists a system M' such that $M \xrightarrow{\hat{\alpha}} M'$ and for all processes P it holds $M' \bullet P \mathrel{\mathcal{R}} N' \bullet P$;*
- *if $N \xrightarrow{*.\mathtt{enter\_}n} N'$ then $(M \mid n[P]) \mathrel{\mathcal{R}} N' \bullet P$, for all processes P;*
- *if $N \xrightarrow{*.\mathtt{exit\_}n} N'$ then $n[M \mid P] \mathrel{\mathcal{R}} N' \bullet P$, for all processes P.*

We write $M \lesssim N$, if $M \mathrel{\mathcal{R}} N$ for some expansion $\mathcal{R}$.

Definition 5 (Bisimulation up to context and up to $\gtrsim$). *A symmetric relation $\mathcal{R}$ is a* bisimulation up to context and up to $\gtrsim$ *if $P \mathrel{\mathcal{R}} Q$ implies:*

- *if $M \xrightarrow{\alpha} M''$, $\alpha \notin \{*.\mathtt{enter\_}n, *.\mathtt{exit\_}n\}$, then there exists a system N'' such that $N \stackrel{\hat{\alpha}}{\Longrightarrow} N''$, and for all processes P there is a system context $C[-]$ and systems M' and N' such that $M'' \bullet P \gtrsim C[M']$, $N'' \bullet P \gtrsim C[N']$, and $M' \mathrel{\mathcal{R}} N'$;*
- *if $M \xrightarrow{*.\mathtt{enter\_}n} M''$ then there exists a system N'' such that $N \mid n[\,\circ\,] \Rightarrow N''$, and for all processes P there is a system context $C[-]$ and systems M' and N' such that $M'' \bullet P \gtrsim C[M']$, $N'' \bullet P \gtrsim C[N']$, and $M' \mathrel{\mathcal{R}} N'$;*
- *if $M \xrightarrow{*.\mathtt{exit\_}n} M''$ then there exist a system N'' such that $n[\circ \mid N] \Rightarrow N''$, and for all processes P there is a system context $C[-]$ and systems M' and N' such that $M'' \bullet P \gtrsim C[M']$, $N'' \bullet P \gtrsim C[N']$, and $M' \mathrel{\mathcal{R}} N'$.*

Theorem 5. *If $\mathcal{R}$ is a bisimulation up to context and up to $\gtrsim$, then $\mathcal{R} \subseteq \approx$.*

5 Algebraic Theory

Here we prove a collection of algebraic laws using our bisimulation proof methods, and the correctness of a protocol for controlling access through a *firewall*, first proposed in [4]. We recall that M, N range over systems and P, Q, R over processes. The first two laws are examples of local communication within private ambients without interference. The third law is the well-known perfect firewall law. The following four laws represent non-interference properties about movements of private ambients. Finally, the last two laws say when opening cannot be interfered.

Theorem 6.

1. $(\boldsymbol{\nu} n)n[\langle W\rangle.P \mid (x).Q \mid M] \;\cong\; (\boldsymbol{\nu} n)n[P \mid Q\{^W/_x\} \mid M]$ *if* $n \notin \mathrm{fn}(M)$
2. $(\boldsymbol{\nu} n)n[\langle W\rangle.P \mid (x).Q \mid \prod_{j\in J} \mathtt{open\_}k_j.R_j] \;\cong\; (\boldsymbol{\nu} n)n[P \mid Q\{^W/_x\} \mid \prod_{j\in J} \mathtt{open\_}k_j.R_j]$
3. $(\boldsymbol{\nu} n)n[P] \;\cong\; \mathbf{0}$ *if* $n \notin \mathrm{fn}(P)$
4. $(\boldsymbol{\nu} n)((\boldsymbol{\nu} m)m[\mathtt{in\_}n.P] \mid n[M]) \;\cong\; (\boldsymbol{\nu} n)n[(\boldsymbol{\nu} m)m[P] \mid M]$ *if* $n \notin \mathrm{fn}(M)$
5. $(\boldsymbol{\nu} m, n)(m[\mathtt{in\_}n.P] \mid n[\prod_{j\in J} \mathtt{open\_}k_j.R_j]) \;\cong\; (\boldsymbol{\nu} m, n)n[m[P] \mid \prod_{j\in J} \mathtt{open\_}k_j.R_j]$
6. $(\boldsymbol{\nu} n)n[(\boldsymbol{\nu} m)m[\mathtt{out\_}n.P] \mid M] \;\cong\; (\boldsymbol{\nu} n)((\boldsymbol{\nu} m)m[P] \mid n[M])$ *if* $n \notin \mathrm{fn}(M)$
7. $(\boldsymbol{\nu} n)n[m[\mathtt{out\_}n.P] \mid \prod_{j\in J} \mathtt{open\_}k_j.R_j] \;\cong\; (\boldsymbol{\nu} n)(m[P] \mid n[\prod_{j\in J} \mathtt{open\_}k_j.R_j])$ *if* $m \neq k_j$, *for* $j \in J$
8. $n[(\boldsymbol{\nu} m)(\mathtt{open\_}m.P \mid m[N]) \mid Q] \;\cong\; n[(\boldsymbol{\nu} m)(P \mid N) \mid Q]$ *if* $Q \equiv M \mid \prod_{j\in J}\langle W_j\rangle.R_j$ *and* $m \notin \mathrm{fn}(N)$
9. $(\boldsymbol{\nu} n)n[(\boldsymbol{\nu} m)(\mathtt{open\_}m.P \mid m[Q]) \mid R] \;\cong\; (\boldsymbol{\nu} n)n[(\boldsymbol{\nu} m)(P \mid Q) \mid R]$ *if* $R \equiv \prod_{i\in I}\langle W_i\rangle.S_i \mid \prod_{j\in J} \mathtt{open\_}k_j.R_j$ *and* $m, n \notin \mathrm{fn}(Q)$.

Proof [Sketch] The laws above are proved by exhibiting the appropriate bisimulation, possibly up to context. We illustrate the proof of the law (3). Let $\mathcal{S} = \{((\boldsymbol{\nu} n)n[Q], \mathbf{0}) \mid \forall Q \text{ s.t. } n \notin \mathrm{fn}(Q)\}$. We show that $\mathcal{S}$ is a bisimulation up to context and up to structural congruence. The most delicate cases are those regarding the silent moves $*.\mathtt{enter\_}k$ and $*.\mathtt{exit\_}k$. For instance, if

$$(\boldsymbol{\nu} n)n[P] \xrightarrow{*.\mathtt{enter\_}k} (\boldsymbol{\nu} n)k[\circ \mid n[P']] \equiv k[\circ \mid (\boldsymbol{\nu} n)n[P']] \qquad \text{then} \qquad \mathbf{0} \mid k[\,\circ\,] \Longrightarrow\equiv k[\circ \mid \mathbf{0}]$$

and up to context and structural congruence we are still in $\mathcal{S}$. $\square$

Crossing a firewall Consider the following protocol in MA:

$$AG \stackrel{\text{def}}{=} m[\mathtt{open\_}k.(x).x.Q] \qquad\qquad FW \stackrel{\text{def}}{=} (\boldsymbol{\nu} w)w[\mathtt{open\_}m.P \mid k[\mathtt{out\_}w.\mathtt{in\_}m.\langle \mathtt{in\_}w\rangle]]$$

The ambient w represents the firewall; the ambient m is a trusted agent containing a process Q, supposed to cross the firewall. The firewall ambient sends into the agent a pilot ambient k with the capability $\mathtt{in\_}w$ for entering the firewall. The agent acquires the capability by opening k. The process Q carried by the agent is finally liberated inside the firewall by the opening of ambient m. Names m and k act like passwords that guarantee the access only to authorised agents. The correctness (of a slight variant) of the protocol above is shown in [4] for may-testing [6] proving that $(\boldsymbol{\nu} m, k)(AG \mid FG) \;\cong\; (\boldsymbol{\nu} w)w[Q \mid P]$ under the conditions that $w \notin \mathrm{fn}(Q)$, $x \notin \mathrm{fv}(Q)$, $\{m, k\} \cap (\mathrm{fn}(P) \cup \mathrm{fn}(Q)) = \emptyset$. The proof relies on non-trivial contextual reasonings.

The system on the right can be obtained from that one on the left by executing six τ-actions. We prove that $\cong$ is insensitive to all these τ-actions.

Lemma 3. *Let P, Q, and R be processes. Then*

1. $(\boldsymbol{\nu} k, m, w)(k[\mathtt{in\_}m.P] \mid m[\mathtt{open\_}k.Q] \mid w[\mathtt{open\_}m.R]) \cong (\boldsymbol{\nu} k, m, w)(m[k[P] \mid \mathtt{open\_}k.Q] \mid w[\mathtt{open\_}m.R])$
2. $(\boldsymbol{\nu} m, w)(m[\langle \mathtt{in\_}w \rangle \mid (x).P] \mid w[\mathtt{open\_}m.Q]) \cong (\boldsymbol{\nu} m, w)(m[P\{^{\mathtt{in\_}w}/_x\}] \mid w[\mathtt{open\_}m.Q])$

Theorem 7. *If $w \notin \mathrm{fn}(Q)$, $x \notin \mathrm{fv}(Q)$, and $\{m, k\} \cap (\mathrm{fn}(P) \cup \mathrm{fn}(Q)) = \emptyset$, then $(\boldsymbol{\nu} m, k)(AG \mid FG) \cong (\boldsymbol{\nu} w)w[Q \mid P]$.*

Proof The result follows from transitivity of $\approx$, because of Law (7) of Theorem 6, Law (1) of Lemma 3, Law (9) of Theorem 6, Law (2) of Lemma 3, and Laws (5) and (9) of Theorem 6. $\square$

6 Related Work

Higher-order LTSs for Mobile Ambients can be found in [3, 10, 27, 7]. But we are not aware of any form of bisimilarity defined using these LTSs. A simple first-order LTS for MA without restriction is proposed by Sangiorgi in [23]. Using this LTS the author defines an *intensional* bisimilarity for MA that separates terms on the basis of their internal structure.

Our work is the natural prosecution of [14] where it was proposed an LTS and a labelled characterisation of reduction barbed congruence for a variant of Levi and Sangiorgi's Safe Ambients, called SAP. The main differences with respect to [14] are the following:

– SAP differs from MA for having co-capabilities and passwords, both features are essential to prove the characterisation result in SAP.
– Our env-actions, unlike those in [14], are truly late, as they do not mention the process provided by the environment. This process can be added *later*, when playing the bisimulation game.
– Our actions for ambient's movement, unlike those in SAP, report the name of the migrating ambient. For instance, in $k.\mathtt{enter\_}n$ we say that ambient k enters n. The knowledge of k is necessary to make the action observable for the environment. This is not needed in SAP, because movements can be observed by means of co-capabilities.
– Co-capabilities also allow the observation of the movement of an ambient whose name is private. As a consequence, the perfect firewall equation does not hold neither in SAP, nor in Safe Ambients. In MA, the movements of an ambient whose name is private cannot be observed. This is why the perfect firewall equation holds.

Apart from [14], other forms of bisimilarity for higher-order calculi, such as Distributed π-calculus [11], Seal [28], Nomadic Pict [26], and a Calculus for Mobile Resources [9], can be found in [16, 5, 26, 9, 2], but only [16, 9, 2] prove labelled characterisations of a contextually defined notion of equivalence. The perfect firewall equation as already been proved for Morris-style contextual equivalence in [10] using a context lemma.

Finally, we believe that interesting labelled characterisations of typed reduction barbed congruence for MA can be derived along the lines of [16], enhancing the algebraic laws of Section 5.

Acknowledgements Vladimiro Sassone spotted a problem in an early draft of the paper. The anonymous referees contributed useful comments. Francesco Zappa Nardelli is grateful to the Foundations of Computing Group of University of Sussex for the kind hospitality and support. He is partly funded by 'MyThS: Models and Types for Security in Mobile Distributed Systems', EU FET-GC IST-2001-32617.

References

1. R. Amadio, I. Castellani, and D. Sangiorgi. On bisimulations for the asynchronous π-calculus. *Theoretical Computer Science*, 195:291–324, 1998.

2. M. Bugliesi, S. Crafa, M. Merro, and V. Sassone. Communication interference in mobile boxed ambients. Forthcoming Technical Report. An extended abstract appeared in Proc. FSTTCS'02, LNCS, Springer-Verlag.
3. L. Cardelli and A. Gordon. A commitment relation for the ambient calculus. Unpublished notes, 1996.
4. L. Cardelli and A. Gordon. Mobile ambients. *Theoretical Computer Science*, 240(1):177–213, 2000. An extended abstract appeared in *Proc. of FoSSaCS '98*.
5. G. Castagna and F. Zappa Nardelli. The seal calculus revisited: Contextual equivalence and bisimilarity. In *Proc. 22nd FSTTCS '02*, volume 2556 of *LNCS*. Springer-Verlag, 2002.
6. R. De Nicola and M. Hennessy. Testing equivalences for processes. *Theoretical Computer Science*, 34:83–133, 1984.
7. G. Ferrari, U. Montanari, and E. Tuosto. A LTS semantics of ambients via graph synchronization with mobility. In *Proc. ICTCS*, volume 2202 of *LNCS*, 2001.
8. C. Fournet and G. Gonthier. A hierarchy of equivalences for asynchronous calculi. In *Proc. 25th ICALP*, pages 844–855, 1998.
9. J.C. Godskesen, T. Hildebrandt, and V. Sassone. A calculus of mobile resources. In *Proc. 10th CONCUR '02*, volume 2421 of *LNCS*, 2002.
10. A. D. Gordon and L. Cardelli. Equational properties of mobile ambients. *Journal of Mathematical Structures in Computer Science*, 12:1–38, 2002.
11. M. Hennessy and J. Riely. A typed language for distributed mobile processes. In *Proc. 25th POPL*. ACM Press, 1998.
12. K. Honda and N. Yoshida. On reduction-based process semantics. *Theoretical Computer Science*, 152(2):437–486, 1995.
13. F. Levi and D. Sangiorgi. Controlling interference in ambients. In *Proc. 27th POPL*. ACM Press, 2000.
14. M. Merro and M. Hennessy. Bisimulation congruences in safe ambients. In *Proc. 29th POPL '02*. ACM Press, 2002.
15. M. Merro and F. Zappa Nardelli. Bisimulation proof methods for mobile ambients. Computer Science Report 2003:01, `http://cogslib.cogs.susx.ac.uk/csr_abs.php?cs`, University of Sussex, 2003.
16. M. Hennessy M. Merro and J. Rathke. Towards a behavioural theory of access and mobility control in distributed system. To appear in Proc. 5th FoSSaCS '03, LNCS, 2003, Springer-Verlag.
17. R. Milner, J. Parrow, and D. Walker. A calculus of mobile processes, (Parts I and II). *Information and Computation*, 100:1–77, 1992.
18. R. Milner and D. Sangiorgi. Barbed bisimulation. In *Proc. 19th ICALP*, volume 623 of *LNCS*, pages 685–695. Springer Verlag, 1992.
19. D.M. Park. Concurrency on automata and infinite sequences. In P. Deussen, editor, *Conf. on Theoretical Computer Science*, volume 104 of *LNCS*. Springer Verlag, 1981.
20. 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.
21. D. Sangiorgi. Bisimulation for Higher-Order Process Calculi. *Information and Computation*, 131(2):141–178, 1996.
22. D. Sangiorgi. On the bisimulation proof method. *Journal of Mathematical Structures in Computer Science*, 8:447–479, 1998.
23. D. Sangiorgi. Extensionality and intensionality of the ambient logic. In *Proc. 28th POPL*. ACM Press, 2001.
24. D. Sangiorgi and R. Milner. The problem of "Weak Bisimulation up to". In *Proc. CONCUR '92*, volume 630 of *LNCS*, pages 32–46. Springer Verlag, 1992.
25. D. Sangiorgi and D. Walker. *The π-calculus: a Theory of Mobile Processes*. Cambridge University Press, 2001.
26. A. Unyapoth and P. Sewell. Nomadic Pict: Correct communication infrastructures for mobile computation. In *Proc. 28th POPL*. ACM, January 2001.
27. M. G. Vigliotti. Transition systems for the ambient calculus. Master thesis, Imperial College of Science, Technology and Medicine (University of London), September 1999.
28. J. Vitek and G. Castagna. Seal: A framework for secure mobile computations. In *Internet Programming Languages*, 1999.