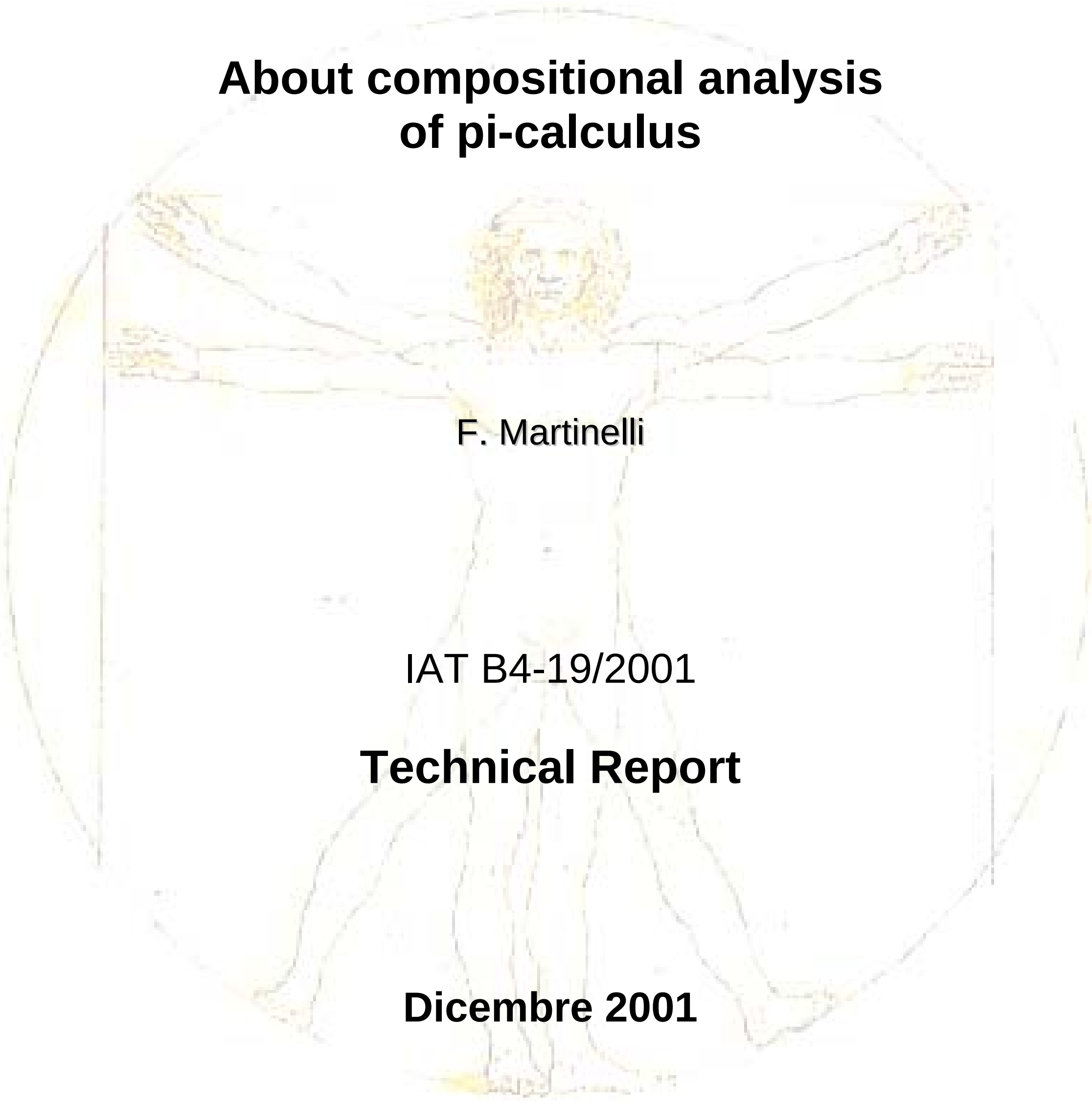
Consiglio Nazionale delle Ricerche



About compositional analysis of pi-calculus

F. Martinelli

IAT B4-19/2001

Technical Report

Dicembre 2001

Istituto per le Applicazioni Telematiche

About compositional analysis of π–calculus processes[⋆]

Fabio Martinelli

Istituto per le Applicazioni Telematiche - C.N.R., Pisa, Italy.
e-mail: Fabio.Martinelli@iat.cnr.it
Tel: +39 050 3153425; Fax: +39 050 3152593

Abstract. We set up a logical framework for the compositional analysis of finite π–calculus processes. In particular, we extend the partial model checking techniques developed for value passing process algebras to a nominal calculus, i.e. the π–calculus. The logic considered is an adaptation of the ambient logic to the π–calculus. As one of the possible applications, we show that our techniques may be used to study interesting security properties as confidentiality for (finite) π–calculus processes.

1 Introduction

The π–calculus [22] is a compact and expressive language for describing concurrent systems. This calculus is suitable for describing processes whose communication topology may change during the computation. Processes communicate by performing sending or receiving actions on channels. Actions may be performed only on channels whose name is known by the process. Thus, the notion of name plays a central role in this calculus. Processes can send and receive names. So, if P sends the name n to Q then also Q can communicate on the channel n. Consider the following two terms:

$$P \doteq c\langle n\rangle$$
$$Q \doteq c(y).y\langle m\rangle$$

denoting two π–calculus processes. The process P emits on the channel c the name n (although note that both c and n are names in the π–calculus). The process Q is willing to receive a name on the channel c and after it emits the name m on that channel. The parallel composition of P and Q, i.e. $P \,|\, Q$, evolves in the process $n\langle m\rangle$ through a reduction, i.e. an internal communication between P and Q. This fact is represented as $P \,|\, Q \longrightarrow n\langle m\rangle$. Thus, the process Q now is able to communicate on the

[⋆] Work partially supported by Microsoft Research Europe and by CSP/CNR project "SeTAPS".

channel n. Another interesting feature is the possibility for processes to create new local names, i.e. names which no other process can refer to. Consider the process P' defined as

$$P' \doteq \nu n(c\langle n\rangle \mid n(y))$$

The idea is that n is a name different from all the others outside the restriction ν. Note that also restricted (or private) names can be communicated. When this happens, the scope of the restriction changes (*scope extrusion*) by including also the receiving process, e.g.:

$$P' \mid Q = \nu n(c\langle n\rangle \mid n(y)) \mid Q \longrightarrow \nu n(n(y) \mid n\langle m\rangle)$$

This models n is a private name of P' and Q after the reduction.

In this paper, we are interested in extending the compositional analysis techniques called partial model checking (e.g., see [2, 13]) to the π–calculus. Basically, suppose we want to verify that a system $P \mid Q$ enjoys a property expressed by a formula A of a certain logic. Then, we can simply study if one of the two components, say Q, satisfies a property A' which encodes the necessary and sufficient conditions on Q s.t. $P \mid Q$ enjoys A.

The advantages of such a compositional reasoning for concurrent systems are various. In [2], partial model checking has been proposed as an efficient method for performing model checking of μ–calculus formulas w.r.t. Labeled Transition Systems (LTSs). The idea is the following. Consider to check whether the composition of k processes $P_1 \mid \ldots \mid P_k \mid \mathbf{0}$[1] satisfies a formula A. By applying once the partial model checking we obtain a formula A_1 that $P_2 \mid \ldots \mid P_k \mid \mathbf{0}$ must satisfy. Here, one could apply some equivalence reductions on the formula A_1, in order to obtain a smaller and so more tractable formula. After k applications of the partial model checking, we obtain that the stuck process $\mathbf{0}$ must satisfy a formula A_k. The model checking of a formula w.r.t. the stuck process is usually very efficient[2]. Thus, the model checking of a system which consists of k parallel-running processes is reduced, through partial model checking, to the model checking of the stuck process $\mathbf{0}$.

Another application of such compositional analysis techniques is during the system design. It is common first to delineate the general architecture of a complex system and next implement its components. In

[1] The tail $\mathbf{0}$ is the process that does nothing; in several calculi, and also in the π–calculus, $P \mid \mathbf{0}$ is considered “equivalent” to P.

[2] Unfortunately, this is not true for the full logic we adopt here, as we show in the remainder of the paper.

particular, with partial model checking, it is possible to derive the properties that specific sub-components must ensure in such a way that the whole system respects its requirements. So, assume we are designing a system that must satisfy some requirements expressed by a logical formula A. Moreover, assume that, somehow, we have already a component of this system, say P; for instance, this could be a plant which needs a control device. So, we wish to derive the description of the corresponding controller, say C, s.t. $C \mid P$ satisfies A. As a matter of fact, by means of partial model checking, we can find the necessary and sufficient conditions on the controller C s.t., whenever this is composed with the plant, the overall system enjoys its specification. Thus, we can directly work only on the specification for the controller without considering the plant.

There are also specific analysis problems, like the verification of security protocols, where the compositional analysis provided by partial model checking is particularly useful. Indeed, the verification scenario for security protocols is to check the whether the protocol participants are able to successfully complete their assigned roles even in the presence of an enemy which tries to interfere with the execution (e.g., see [11]). Let P be the process describing the behavior of honest agents of the protocol. The enemy could be whatever process one may specify in a given language, say X, possibly enjoying certain initial assumptions (e.g., the set of messages it knows). Thus, by following [16, 18, 19], we can state security properties as:

$$\forall X \quad P \mid X \models A$$

and then apply partial model checking techniques to reduce such verification problem to a validity one, i.e.:

$$\forall X \quad X \models A'$$

which may be faced by using standard results of logic. So far, this idea has been applied to the analysis of several security properties for systems which may be described through variants of the CCS process algebra [20], and properties expressed with modal logics as the Hennessy-Milner one or the $\mu-$calculus [12].

In this paper, as logic for describing the process properties, we adopt a restriction of the *ambient* logic developed by Cardelli and Gordon in [5, 6] to the $\pi-$calculus. The reason is that this logic is suitable for reasoning about free and restricted names. Furthermore, this logic is defined in terms of *structural congruence* between processes. This equivalence relation takes into account the spatial structure of processes, e.g. how many parallel processes are running and how these are related by the scope of restriction operators. To the best of our knowledge, this is the first attempt to develop a partial model checking analysis for a nominal calculus, and moreover, for a logic with operators which can express also the spatial structure of processes.

The techniques we develop here, even though present some restrictions to their application, are powerful enough to study interesting properties of the π–calculus. In particular, we obtain an effective method for the verification of confidentiality properties for finite π–calculus processes, i.e. if a (restricted) name is leaked to the external environment. As noticed in [22], restricted names may be used to control the access rights to system resources; the leakage of such names may cause unauthorized accesses to such resources.

Organization of the paper. In Section 2, we describe the version of the π–calculus we adopt. In Section 3, we introduce the logic we use, i.e. a restriction of the ambient logic to the π–calculus. Section 4 is the main one and presents the partial model checking techniques. In Section 5 we show how to apply partial model checking techniques to study security properties, in particular the so–called Dolev-Yao confidentiality [9]. In Section 6, we discuss about some further work in this topic.

2 Asynchronous π–calculus

In this section we briefly recall some basic concepts about the asynchronous π–calculus (e.g., see [3, 21]).

Given a countable set of names $\mathcal{N}$ (ranged over by $a, b, \ldots, n, m, \ldots$) the set of π–calculus processes is defined through the following BNF grammar:

$$\begin{array}{rll} P, Q ::= & \mathbf{0} & \text{(Zero)} \\ \mid & a\langle n\rangle & \text{(Output)} \\ \mid & a(n).P & \text{(Input)} \\ \mid & (\nu n)P & \text{(Restriction)} \\ \mid & P \mid Q & \text{(Parallel composition)} \end{array}$$

The name n is said bound in the terms $(\nu n)P$ and $a(n).P$. Given a term P we inductively define the set of free names of P, namely $fn(P)$, as:

$$\begin{aligned} fn(\mathbf{0}) &= \emptyset \\ fn(n\langle n'\rangle) &= \{n, n'\} \\ fn(n(n').P) &= (fn(P) \setminus \{n'\}) \cup \{n\} \\ fn(\nu n P) &= fn(P) \setminus \{n\} \\ fn(P \mid Q) &= fn(P) \cup fn(Q) \end{aligned}$$

We give an intuitive explanation of the operators of the calculus:

- $\mathbf{0}$ is the stuck process that does nothing.
- $a\langle n\rangle$ is the output process. Briefly, it denotes a communication on the channel a of the name n. Note that channel names can be communicated.
- $a(n).P$ is the input construct. A name is received on the channel a and its value is substituted to the free occurrences of the name n.
- $(\nu n)P$ is the name restriction. The idea is that n is a local name of P.
- $P \,|\, Q$ is the parallel composition of two processes P and Q.

We also define the *structural congruence* as follows[3]. Let $\equiv$ be the least congruence relation over processes closed under the following rules:

1. $P \equiv Q$, if P is obtained through α–conversion from Q.
2. $P \,|\, \mathbf{0} \equiv P$;
3. $P \,|\, Q \equiv Q \,|\, P$;
4. $P \,|(Q \,|\, R) \equiv (P \,|\, Q) \,|\, R$;
5. $\nu n \mathbf{0} \equiv \mathbf{0}$;
6. $\nu n \nu m P \equiv \nu m \nu n P$;
7. $\nu n \nu n(P) \equiv \nu n(P)$;
8. $\nu n(a\langle m\rangle) \equiv a\langle n\rangle$, if $n \notin \{a, m\}$;
9. $\nu n(a(m).P) \equiv a(m).\nu n(P)$, if $n \notin \{a, m\}$;
10. $\nu n(P \,|\, Q) \equiv P \,|\, \nu n Q$ if $n \notin fn(P)$.

For convenience, we often write $\nu N(P)$ for $\nu n_1 \ldots \nu n_j(P)$ where $N = \{n_1, \ldots, n_j\}$. When N is empty, we assume that $\nu N(P) = P$. (We do not loose information by considering $\nu N(P)$ instead of $\nu n_1..\nu n_j(P)$ because of the rules on structural congruence.)

We give the *reduction* semantics for the asynchronous π–calculus. Processes communicate among them by exchanging messages. An internal communication (or *reduction*) of the process P is denoted by $P \longrightarrow P'$. We have the following rules for calculating the reduction relation between processes:

$$\frac{}{a\langle n\rangle \,|\, a(m).P \longrightarrow P[n/m]} \tag{1}$$

$$\frac{P \equiv Q, Q \longrightarrow Q', Q \equiv P'}{P \longrightarrow P'} \tag{2}$$

[3] Several sets of axioms are given in the literature for describing such congruence. We use the definition and related results given in [10]. (Note that this set is not necessarily minimal.)

$$\frac{P \longrightarrow P'}{P \,|\, Q \longrightarrow P' \,|\, Q} \tag{3}$$

$$\frac{P \longrightarrow P'}{\nu n(P) \longrightarrow \nu n(P')} \tag{4}$$

where $P[n/m]$ denotes the process P where all the free occurrences of m are replaced with n.

3 A logic on π–calculus

In this section we describe the logic we use to express properties of $\pi-$processes. This is a restriction of the ambient logic of Cardelli and Gordon [5, 6] to π–calculus[4]. The syntax of formulas is given in Tab. 1. The logic permits us to express both temporal and spatial properties of

$A ::=$	$\mathbf{T}$	Logical constant true
	$\neg A$	Negation
	$A_1 \vee A_2$	Disjunction
	$\eta\langle\eta'\rangle A$	Output
	$\eta(\eta')A$	Input
	$\bigcirc A$	Reduction
	$A \cdot_R \eta$	Hiding
	$\eta \cdot_R A$	Revelation
	$\mathbf{0}$	Zero
	$A \,\vert\, B$	Composition
	$A \rhd B$	Adjunct of the composition
	$\forall x A$	Universal quantification

where $\eta(\eta')$ are variables $x \in \mathcal{V}$ or names $n \in \mathcal{N}$.

Table 1. Syntax of the logic.

processes; moreover it allows to treat with restricted names in a conve-

[4] Recently in [3], Cardelli and Caires adapted several concepts of the ambient logic to the $\pi-$calculus by adding also recursion. Clearly, the logic we use here is also a restriction of the one of Cardelli and Caires (although we adopt slightly different input/output modalities).

nient way. The logic, besides the usual constants and operators of propositional logic, has three modalities for expressing the temporal behavior of processes:

- $\eta\langle\eta'\rangle A$. This formula expresses a process may send the name η' on the channel η and then it satisfies A.
- $\eta(\eta')A$. This formula expresses that a process may receive the name η' on the channel η and then it satisfies A.
- $\bigcirc A$. This formula expresses that a process performs a reduction (an internal communication) and then it satisfies A[5].

Moreover, the logic permits us to represent the spatial structure of processes, in particular:

- $\mathbf{0}$[6]. This formula requires that the process is structurally equivalent to $\mathbf{0}$.
- $A \mid B$. This formula expresses that the process is (or better is structurally equivalent to) a composition of two processes. One of them satisfies A, while the other satisfies B.
- $A \triangleright B$. This formula expresses that the composition of the process with whatever process satisfying A enjoys the formula B.

But, the main feature of this logic is its treatment of the (restricted) names. The logic uses two operators for managing names.

- $A \cdot_R n$. This formula expresses that a process, after the restriction of the name n enjoys A.
- $n \cdot_R A$. This formula expresses that a process is a equivalent to another one under the restriction of n. After that the restriction is removed, the resulting process enjoys A.

We have also a universal quantifier $\forall x A$, which may be used to state that a property A always holds when one substitutes any name for the variable x.

The truth relation $\models$ for the logic is inductively defined in Tab. 2.

[5] In [6], a different operator is used, namely $\Diamond$, whose semantics is similar to $\bigcirc A$ where the reduction relation is replaced with its reflexive and transitive closure. Thus, all the reachable processes through finite sequences of reductions are inspected instead of the ones reachable with only one reduction step. However, when dealing with finite π–calculus processes, the $\Diamond$ modality may often be equivalently expressed through the $\bigcirc$ one plus the disjunction operator (e.g., see Section 5).

[6] Note that there is an overloading of the symbols $\mathbf{0}$ and $\mid$ used both in the logic and in the process calculus. However, they respectively represent the same concept in the two languages and their actual role should be clear from the context.

$P \models \mathbf{T}$		For all P
$P \models A \vee B$	iff	$P \models A$ or $P \models B$
$P \models \neg A$	iff	Not $P \models A$
$P \models a\langle n \rangle A$	iff	$P \equiv a\langle n \rangle \mid P'$ and $P' \models A$
$P \models a(n)A$	iff	$P \equiv \nu N(a(m).P' \mid P'')$, with $a, n \notin N$,
		and $\nu N(P'[n/m] \mid P'') \models A$
$P \models \bigcirc A$	iff	$P \longrightarrow P'$ and $P' \models A$
$P \models n \cdot_R A$	iff	$P \equiv \nu n P'$ and $P' \models A$
$P \models A \cdot_R n$	iff	$\nu n P \models A$
$P \models \mathbf{0}$	iff	$P \equiv \mathbf{0}$
$P \models A \mid B$	iff	$P \equiv Q' \mid Q''$ and $Q' \models A, Q'' \models B$
$P \models A \triangleright B$	iff	$\forall P' \models A$ we have $P \mid P' \models B$
$P \models \forall x A$	iff	$\forall n \in \mathcal{N}$ we have $P \models A[n/x]$

Table 2. Formal semantics of the logic.

Note 1. We added two operators, i.e. $\eta\langle \eta' \rangle A$ and $\eta(\eta')A$, to the logic in [6]. Actually, if one introduces the output operator, then the input one can be derived by using the adjunct of the composition and the output one. But, in this way, we mix two operators which denote conceptually different aspects, i.e. the temporal and the spatial structure of processes. (A similar situation arises if one defines the output modality, as $\eta\langle \eta' \rangle$ and then our output modality as derived one, i.e. $\eta\langle \eta' \rangle \mid A$.) With our choice, it is straightforward to tell apart the full logic from sub-logic which specifically deals with the temporal aspects, i.e. the full logic without $\mathbf{0}$, $\mid$ and $\triangleright$. This may be useful if one wishes to study the process equivalence induced by the logic: we make no claims but we feel that the equivalence induced by the sub-logic without spatial operators should coincide with barbed congruence while the one induced by the full logic with structural congruence (see the results of Sangiorgi in [23] for the ambient logic). However, we stress that the choice of input/ output modalities is only matter of taste; our results still hold with different input/output modalities. ■

We give some examples of the usage of the logic for expressing properties of π–calculus processes. As usual, we define $\exists x A$ as $\neg \forall x \neg A$, $A \wedge B$ as $\neg(\neg A \vee \neg B)$, $A \Longrightarrow B$ as $\neg A \vee B$ and $\mathbf{F} = \neg \mathbf{T}$.

Example 1. We can define the equality between η and η' in the logic as $((\eta\langle\eta''\rangle\mathbf{0})\,|\,(\eta'(\eta'')\mathbf{0})) \Longrightarrow \bigcirc\mathbf{T}$. Indeed, $\eta\langle\eta''\rangle\mathbf{0}$ expresses that a process is able to emit on the channel η the free name η'' and after it is equivalent $\mathbf{0}$. This also means that it cannot perform other actions. Similarly, $\eta'(\eta'')\mathbf{0}$ means that the process may only receive a message on the channel η'. The parallel composition of the previous formulas says that the process is able to send a message on η and at the same time to receive the same message on η'. Now, if we have a reduction it means that $\eta = \eta'$ (no other reductions are possible). Moreover, if $\eta = \eta'$ then clearly a reduction may be performed. ■

Example 2. To express that a process P emits a restricted name, we can check whether $P \models \exists x(n \cdot_R (x\langle n\rangle \wedge \neg x = n))$. The revelation operator picks a restricted name of P, if any, call it n and then checks if n is emitted on some channel x, when $x \neq n$. Note, that we can reveal a name n only if n is not a free name of the process. Indeed, the formula $n \cdot_R \mathbf{T}$ states that n is not free in a process (and thus it can be revealed). ■

Example 3. The adjunct of the composition is an interesting operator whose definition involves the quantification over processes. It is interesting to note that, through this operator, it is possible to encode in the logic, the schema for defining the security properties given in the introduction. In particular, we can encode:

$$\forall X \quad P\,|\,X \models B$$

as

$$P \models \mathbf{T} \triangleright B$$

■

We have the following lemma that basically tells that two processes that are structurally congruent cannot be distinguished by a formula of the logic.

Lemma 1. *Given a formula A, if $P \equiv Q$ then $P \models A$ iff $Q \models A$.* ■

Let $Names(A)$ be the set of all names appearing in a formula A.

The following lemma (see [6]) basically states that the truth relation is invariant w.r.t. the renaming of fresh names.

Lemma 2. *Consider a formula A and a process Q. Let N'' be a set of names s.t. $N'' \cap (Names(A) \cup fn(Q)) = \emptyset$. Then,*

$$Q \models A \text{ iff } Q[N''/N] \models A[N''/N]$$ ■

Corollary 1. *Under the hypothesis lemma, suppose also that* $N \cap fn(Q) = \emptyset$ *then*

$$Q \models A \text{ iff } Q \models A[N''/N] \qquad \blacksquare$$

Note 2. When $fn(P) \cap N'' = \emptyset$, we have that $P[N''/N][N/N''] \equiv P$. When $Names(A) \cap N'' = \emptyset$, we have that $A[N''/N][N/N''] = A$. ■

4 Compositional analysis of processes

In this section we provide a technique to reason compositionally about the satisfaction of the properties of the logic.

Our aim is to find the necessary and sufficient condition, expressed by a formula A', on the process X s.t.

$$P \mid X \models A \text{ iff } X \models A'$$

Thus, instead of reasoning about the whole system, we can directly work on one (or more) of its (parallel) components.

Example 4. To show how this works consider, for instance, a process $P = n\langle n'\rangle$ and a formula $A = n\langle n'\rangle\mathbf{T}$. Then, from the definition of the truth relation, we have $P \,|\, X \models A$ iff there exists P' s.t. $P \,|\, X \equiv n\langle n'\rangle \,|\, P'$ and $P' \models \mathbf{T}$. Note that $P \,|\, X \equiv n\langle n'\rangle \,|\, P'$ iff P' is X, or $P' \equiv P \,|\, X'$ and $X \equiv n\langle n'\rangle \,|\, X'$. Thus, the former case imposes no conditions on X, i.e. X could be whatever process; the latter one imposes that $X \models n\langle n'\rangle\mathbf{T}$. By putting together the conditions on X we get $X \models \mathbf{T} \vee n\langle n'\rangle\mathbf{T}$, which is equivalent to require that $X \models \mathbf{T}$. Indeed, the process P is enough to satisfy A. The situation slightly changes if we take $A = n\langle n''\rangle$, with $n'' \neq n'$. In this case, the condition on X is $X \models n\langle n''\rangle$, since P cannot contribute to the satisfaction of the formula A. ■

We must face a specific problem directly related with the restriction operator of the π–calculus and its scope extrusion mechanism. Indeed, the process $P \,|\, X$ may perform a reduction which depends on a communication of a private name of P (resp. of X) to X (resp. P). Thus, we may have $P \,|\, X \longrightarrow \nu n(P' \,|\, X')$. So, the evaluation context is changed. Note that this situation does not arise for CCS-like process algebras (e.g., see [2]), where the channel restriction is a static operator, i.e. it does not change its scope, whereas in the π–calculus is dynamic. Thus, we perform the partial model checking of more general contexts, as:

$$\nu N(P \,|\, (\_))$$

where N could be possibly empty.

Another specific problem that we encounter in this study is that the definition of the semantics of both the π–calculus and the logic heavily depend on the structural congruence. Instead, the previously defined frameworks for partial model checking usually rely on Labeled Transition Systems (LTSs) (e.g., see [2, 13]). Note that it is possible to give a semantics of the π–calculus in terms LTSs, however the spatial operators of the logic require the notion of structural congruence (while for the others it is possible to give a semantics through a suitable notion of LTSs). Thus, we develop the partial model checking techniques in a framework completely based on structural congruence.

We introduce some auxiliary lemmas that show how it is possible to decompose processes in several formats, up to structural equivalence. Most of them are straightforwardly derived from the results about structural congruence by Engelfriet and Gelsema (see [10]).

Given a name n, we can find only a finite number of processes P', up to structural equivalence, such that P is structurally equivalent to the restriction of n in P'.

Lemma 3. *Given a process P, we can effectively compute a finite set of processes $res(P, n)$ s.t.:*

1. *$P \equiv \nu n P'$ implies that there exists $P'' \in res(P, n)$ s.t. $P'' \equiv P'$;*
2. *$P'' \in res(P, n)$ implies $P \equiv \nu n P''$.* ■

Given a process $a\langle n\rangle$, we can find only a finite number of processes P', up to structural equivalence, such that P is structurally equivalent to composition of the output process $a\langle n\rangle$ with P'. This gives us all the possible continuations of the process P after an output.

Lemma 4. *Given a process P, we can effectively compute a finite set of processes $C^o(a\langle n\rangle, P)$ s.t.:*

1. *$P \equiv a\langle n\rangle \mid P'$ implies that there exists $P'' \in C^o(a\langle n\rangle, P)$ s.t. $P'' \equiv P'$;*
2. *$P'' \in C^o(a\langle n\rangle, P)$ implies $P \equiv a\langle n\rangle \mid P''$.* ■

Given a process $a\langle n\rangle$, we can find only a finite number of processes P', up to structural equivalence, such that P is structurally equivalent to the composition of the output process $a\langle n\rangle$ with P' under the restriction of n. This gives us all the possible continuations of the process P after the output of a restricted name.

Lemma 5. *Given a process P, we can effectively compute a finite set of processes $C^{ro}(a\langle n\rangle, P)$ s.t.:*

1. *$P \equiv \nu n(a\langle n\rangle \mid P')$, with $a \neq n$, implies there exists $P'' \in C^{ro}(a\langle n\rangle, P)$ s.t. $P'' \equiv P'$;*
2. *$P'' \in C^{ro}(a\langle n\rangle, P)$ implies $P \equiv \nu n(a\langle n\rangle \mid P'')$, with $a \neq n$.* ■

A process P after the receiving on a channel a of a name n may be only finitely decomposed, up to structural equivalence, as the restriction on a set of channels different from a and n of a composition of two processes s.t. one is the residue after the communication. This gives us all the possible continuations after the reception of the value n.

Lemma 6. *Given a process P, we can effectively compute a finite set of triples $C^i(a(-), n, P)$ s.t.:*

1. *$P \equiv \nu N(a(m).P' \mid P'')$, with $a, n \notin N$, implies there exists $(N_1, P_1', P_1'') \in C^i(a(-), n, P)$ s.t. $N \cap fn(a(m).P' \mid P'') = N_1, P'[n/m] \equiv P_1'$ and $P'' \equiv P_1''$;*
2. *$(N_1, P_1', P_1'') \in C^i(a(-), n, P)$ implies $P \equiv \nu N_1(a(m).P' \mid P'')$, with $a, n \notin N_1$, $P_1' = P'[n/m]$ and $P_1'' = P''$.* ■

A process P may be finitely represented as the composition of pairs of processes, up to structural equivalence.

Lemma 7. *Given a process P, we can effectively compute a finite set of pairs $C^{comp}(P)$ s.t.:*

1. *$P \equiv P' \mid P''$, implies that there exists $(P_1', P_1'') \in C^{comp}(P)$ s.t. $P' \equiv P_1'$ and $P'' \equiv P_1''$;*
2. *$(P_1', P_1'') \in C^{comp}(P)$ implies $P \equiv P_1' \mid P_1''$.* ■

Note that we can study the fact that a process performs a reduction, by considering its possible decompositions. In particular, the following lemma states the possible decompositions on P and Q s.t. $\nu N(P \mid Q) \longrightarrow R$.

Lemma 8. *We have that $\nu N(P \mid Q) \longrightarrow R$ iff one of the following cases holds:*

1. *$P \longrightarrow P'$ and $R \equiv \nu N(P' \mid Q)$;*
2. *$P \equiv a\langle n\rangle \mid P', Q \equiv \nu N''(a(m).Q' \mid Q'')$, with $a, n \notin N''$, and $R \equiv \nu N(P' \mid$
$\nu N''(Q'[n/m] \mid Q''))$;*
3. *$P \equiv \nu n(a\langle n\rangle \mid P'), Q \equiv \nu N''(a(m).Q' \mid Q'')$, with $a, n \notin N'', a \neq n, n \notin fn(Q)$ and $R \equiv \nu(N \cup \{n\})(P' \mid \nu N''(Q'[n/m] \mid Q''))$;*

4. *$Q \longrightarrow Q'$ and $R \equiv \nu N(P \,|\, Q')$;*
5. *$Q \equiv a\langle n\rangle \,|\, Q', P \equiv \nu N''(a(m).P' \,|\, P'')$, with $a, n \notin N''$, and $R \equiv \nu N(Q' \,|$
$\nu N''(P'[n/m] \,|\, P''))$;*
6. *$Q \equiv \nu n(a\langle n\rangle \,|\, Q'), P \equiv \nu N''(a(m).P' \,|\, P'')$, with $a, n \notin N'', a \neq n, n \notin fn(P)$ and $R \equiv \nu N \cup \{n\}(Q' \,|\, \nu N''(P'[n/m] \,|\, P''))$.* ∎

We make some assumptions that help us to make more tractable the partial model checking problem. In particular, we consider only components X whose set of free names is fixed *a priori*. Moreover, we consider only formulas where the adjunct of the composition has the following format $A \wedge fn^{\subseteq}(N) \triangleright B$, where $fn^{\subseteq}(N)$ is a short-cut for $\forall x(x \cdot_R \mathbf{T} \vee \vee_{n \in N} x = n)$. Basically, $fn^{\subseteq}(N)$ is satisfied by a process P iff $fn(P)$ is contained in N. We also require that the quantification is only present within the definition of $fn^{\subseteq}(N)$. Call $\mathcal{L}_{\setminus\forall}$ such sub-logic.

We have now the technical notions to state the main result of this paper.

Proposition 1. *Let A be a formula in $\mathcal{L}_{\setminus\forall}$. Consider a finite set of names ϕ, a context $\nu N(P \,|(\_))$, with $Names(A) \cap N = \emptyset$, and process X with $fn(X) \subseteq \phi$. Then, we have:*

$$\nu N(P \mid X) \models A \text{ iff } X \models A//_{P,N,\phi}$$

where $A//_{P,N,\phi}$ is the formula defined in Tab. 3. ∎

Requiring that the names of the formula A are not in N is not restrictive. Indeed, by Corollary 1, we can simply rename each name of A in N with fresh ones and then perform the analysis w.r.t. this new formula.

Remark 1. Note that with the previous proposition we are able to reduce some model checking problems for the logic to validity checking problems. For instance, checking that $P \models A' \triangleright B$ holds, where A' is equivalent to require that the free names of the process are contained in ϕ, can be reduced to checking that $A' \Longrightarrow B//_{P,\emptyset,\phi}$ is valid. Indeed, $P \models A' \triangleright B$ iff for all X with $fn(X) \subseteq \phi$ we have $P \,|\, X \models B$; by partial model checking we obtain that X must satisfy $B//_{P,\emptyset,\phi}$. On the other hand, note that validity problems may be encoded as model checking problems. For instance, in order to establish whether or not A is valid, one can simply check if $\mathbf{0} \models \mathbf{T} \triangleright A$ holds. Indeed, by definition, $\mathbf{0} \models \mathbf{T} \triangleright A$ iff $\forall P \models \mathbf{T}$ we have $\mathbf{0} \,|\, P \models A$. By Lemma 1, we have $\mathbf{0} \,|\, P \models A$ iff $P \models A$. ∎

$$
\begin{array}{ll}
\mathbf{T}/\!/_{P,N,\phi} & \doteq \mathbf{T} \\
(A \vee B)/\!/_{P,N,\phi} & \doteq A/\!/_{P,N,\phi} \vee B/\!/_{P,N,\phi} \\
(\neg A)/\!/_{P,N,\phi} & \doteq \neg(A/\!/_{P,N,\phi}) \\
(a\langle n\rangle A)/\!/_{P,N,\phi} & \doteq A_1 \vee A_2,\ \text{where} \\
 & \quad A_1 \doteq a\langle n\rangle(A/\!/_{P,N,\phi}) \quad \text{if } \{a,n\} \subseteq \phi \\
 & \quad A_2 \doteq \vee_{P' \in C^{o}(a\langle n\rangle,P)} A/\!/_{P',N,\phi} \\
(a(n)A)/\!/_{P,N,\phi} & \doteq A_1 \vee A_2,\ \text{where} \\
 & \quad A_1 \doteq a(n)(A/\!/_{P,N,\phi\cup\{n\}}) \quad \text{if } a \in \phi \\
 & \quad A_2 \doteq \vee_{(N',P',P'') \in C^{i}(a(-),n,P)} A/\!/_{\nu N'(P'|P''),N,\phi} \\
(\bigcirc A)/\!/_{P,N,\phi} & \doteq \vee_{P' : P \longrightarrow P'} A/\!/_{P',N,\phi} \\
 & \vee\ \vee_{a\in\phi} \vee_{P' \in C^{o}(a\langle n\rangle,P)}\ a(n)(A/\!/_{P',N,\phi\cup\{n\}}) \\
 & \vee\ \vee_{a\in\phi} \vee_{P' \in C^{ro}(a\langle n'\rangle,P)}\ a(n')(A/\!/_{P',N\cup\{n'\},\phi\cup\{n'\}}) \\
 & \vee\ \bigcirc(A/\!/_{P,N,\phi}) \\
 & \vee\ \vee_{a,n\in\phi} \vee_{(N_1,P',P'') \in C^{i}(a(-),n,P)}\ a\langle n\rangle(A/\!/_{\nu N_1(P'|P''),N,\phi}) \\
 & \vee\ \vee_{a\in\phi} \vee_{(N_1,P',P'') \in C^{i}(a(-),n',P)}\ n' \cdot_R a\langle n'\rangle(A/\!/_{\nu N_1(P'|P''),N\cup\{n'\},\phi\cup\{n'\}}) \\
 & \quad \text{where } n' \text{ is s.t. } n' \notin fn(P) \cup \phi \cup N \cup Names(A) \\
(A \cdot_R n)/\!/_{P,N,\phi} & \doteq (A[n'/n])/\!/_{P,N\cup\{n\},\phi} \\
 & \quad \text{where } n' \text{ is s.t. } n' \notin fn(P) \cup \phi \cup N \cup Names(A) \\
(n \cdot_R A)/\!/_{P,N,\phi} & \doteq A_1 \vee A_2 \vee A_3,\ \text{where} \\
 & \quad A_1 \doteq n \cdot_R \mathbf{T} \wedge (\vee_{n' \in N}(A[n'/n])/\!/_{P,N\setminus\{n'\},\phi}) \quad \text{if } n \notin fn(P) \setminus N \\
 & \quad A_2 \doteq n \cdot_R (A/\!/_{P,N,\phi\cup\{n\}}) \quad \text{if } n \notin fn(P) \cup N \\
 & \quad A_3 \doteq n \cdot_R \mathbf{T} \wedge (\vee_{P' \in res(P,n)} A/\!/_{P',N,\phi}) \quad \text{if } n \notin fn(P) \cup N \\
\mathbf{0}/\!/_{P,N,\phi} & \doteq \mathbf{0} \quad \text{if } P \equiv \mathbf{0} \\
A \mid B/\!/_{P,N,\phi} & \doteq \vee_{(P',P'') \in C^{comp}(P), fn(P') \cap N = \emptyset} \\
 & \quad (fn^{\neg}(N) \wedge A/\!/_{P',\emptyset,\phi\setminus N}) \mid (B/\!/_{P'',N,\phi}) \vee \\
 & \quad (fn^{\neg}(N) \wedge B/\!/_{P',\emptyset,\phi\setminus N}) \mid (A/\!/_{P'',N,\phi}) \\
 & \quad \text{where } fn^{\neg}(\{n_1,\ldots,n_k\}) \doteq \wedge_{l\in\{1,\ldots,k\}} n_l \cdot_R \mathbf{T} \\
(A' \rhd B)/\!/_{P,N,\phi} & \doteq A' \rhd (B/\!/_{P,N,\phi\cup N_1}) \\
 & \quad \text{where } A' = A \wedge fn^{\subseteq}(N_1)
\end{array}
$$

Table 3. Partial model checking function for the context $\nu N(P \,|\, (\_))$. The "else" branch in the definition of auxiliary formulas A_i, with $i = 1..3$, is always $\neg\mathbf{T}$.

5 An application to confidentiality analysis

The results of the previous section, even if deal only with a subset of the logic, are strong enough to prove interesting properties as the confidentiality one.

Consider a protocol P, which runs in a hostile environment X. We may be interested to study whether a name of P remains confined among the agents of P or it is leaked to the outside environment. (This form of confidentiality is sometimes called *Dolev-Yao* secrecy.) This leakage may be represented as the sending on an open channel, say *pub*, of the confidential value, say v.

Definition 1. *Given a context $\nu N(P \,|\, \_)$ and a process X, with $v \in fn(P), pub \in fn(X) \setminus N, v \notin fn(X) \cup N$, we say that v is leaked to X, if $\nu N(P \,|\, X) \longrightarrow^* Q \,|\, pub\langle v\rangle$. If a name v is not leaked, we say it is confidential (w.r.t. X).* ■

Remark 2. Similarly, we could define the leakage of a restricted name n of $P \equiv \nu n P'$, with $n \in fn(P')$ to a process X. But, this kind of property may be treated as an instance of the previous definition, i.e. as the leakage of the free name n of P' to a process X' s.t. $n \notin fn(X')$.

Note that if it could be possible to fix an upper bound to the number of possible interactions between whatever intruder and the system P then it would be possible to express in the logic the confidentiality property. For a while, assume that such bound is n. Then, the formula[7]

$$leaked_v^n = \vee_{1 \le i \le n} \bigcirc^i \, pub\langle v\rangle$$

may be used to state that v is leaked by P (where $pub\langle v\rangle\mathbf{T}$ is abbreviated as $pub\langle v\rangle$) and $\neg leaked_v^n$ that v is confidential. We can prove that such bound actually exists. First, we give an estimation for the maximal length of possible interactions of a system P with its environment. Let $ml(P)$ be recursively defined as:

$$\begin{aligned} ml(\mathbf{0}) &= 0 \\ ml(\nu n P) &= ml(P) \\ ml(a\langle n\rangle) &= 1 \\ ml(a(n).P) &= 1 + ml(P) \\ ml(P \,|\, P') &= ml(P) + ml(P') \end{aligned}$$

[7] We use $\bigcirc^i F$ to denote $\overbrace{\bigcirc \ldots \bigcirc}^{i} F$.

Thus, $\nu N(P \,|\, X) \longrightarrow^*$ may consists of at most $ml(P)$ interactions between P and X plus the interactions internal to the process X. Consider now the situation where X does not contribute to the computation with internal actions. Thus, we have:

$$\begin{array}{c}
\nu N(P \,|\, X) \longrightarrow^* Q \,|\, pub\langle v\rangle \\
\text{iff} \\
\nu N(P \,|\, X) \overbrace{\longrightarrow \ldots \longrightarrow}^{n'} Q \,|\, pub\langle v\rangle \text{ with } n' \leq ml(P) \\
\text{iff} \\
\nu N(P \,|\, X) \models leaked_v^{ml(P)}
\end{array}$$

We show how to build a process X' s.t. if $\nu N(P \,|\, X) \longrightarrow^* Q \,|\, pub\langle v\rangle$ then also $\nu N(P \,|\, X') \longrightarrow^* Q' \,|\, pub\langle v\rangle$ but X' does not perform internal reductions.

Lemma 9. *If $\nu N(P \,|\, X) \longrightarrow^* Q \,|\, pub\langle v\rangle$ then we can find a process X' s.t. $\nu N(P \,|\, X') \longrightarrow^* Q' \,|\, pub\langle v\rangle$, with $fn(X) = fn(X')$ and X' during this computation does not perform any internal reduction.* ■

Note also that we can consider as intruders only processes X s.t. $fn(X) \subseteq (fn(P) \cup \{pub\}) \setminus \{v\}$.

Lemma 10. *Assume that $\nu N'(P \,|\, X) \longrightarrow^* Q \,|\, pub\langle v\rangle$, with $pub, v \notin N'$ and $v \notin fn(X)$. Then, there exists X', with $fn(X') \subseteq (fn(P) \cup \{pub\}) \setminus \{v\}$, s.t.:*

$$\nu N'(P \,|\, X') \longrightarrow^* Q \,|\, pub\langle v\rangle.$$

■

Moreover, we can restrict ourselves to consider the analysis of the formula $leaked_v^{ml(P)}$ only for contexts $\nu fn(P)(P \,|\, X)$, where $fn(X) \subseteq fn(P) \cup \{pub\}$. Thus, we can study whether the value $v \in fn(P)$ is confidential in P, by requiring that there is no process X, with $fn(X) \subseteq (fn(P) \cup pub) \setminus \{v\}$, s.t.:

$$P \,|\, X \models leaked_v^{ml(P)}$$

By partial model checking, we can find

$$F = leaked_v^{ml(P)} //_{P,\emptyset,(fn(P)\cup\{pub\})\setminus\{v\}}$$

that is satisfiable by some process (whose set of free names is contained in $(fn(P) \cup pub) \setminus \{v\}$) if and only if v can be leaked. Alternatively, v is confidential iff F is not satisfiable. Note that the formula obtained, after the partial model checking, only consists of logical constants, disjunctions, revelations, outputs, inputs and reductions. We can produce a satisfiability procedure for such kind of formulas.

Lemma 11. *Let A be a formula which consists only of logical constants, disjunctions, inputs, outputs, revelations and reductions. Then, the problem of establishing whether or not there exists a process X, with $fn(X) \subseteq \phi$, s.t. $X \models A$ is decidable.* ■

Thus, as a simple application of our theory we have that the confidentiality analysis for our processes is decidable.

Proposition 2. *The confidentiality analysis for finite π–calculus processes is decidable.* ■

It is worthy noticing that, from results in [15, 17], several authentication properties can be encoded as properties of the intruder knowledge, and ultimately as confidentiality properties. Thus our results may be used to deal also with authentication properties of finite π–calculus processes.

Remark 3. In Remark 1, it has been shown how by using $\rhd$ operator one can encode validity checking problems as model checking ones. This feature makes the model checking problem for the full logic rather difficult. Indeed, in [7], where the model checking problem for the ambient logic have been studied, no positive results have been given about fragments with the $\rhd$ operator. By means of partial model checking and by Lemma 11, we are able to give a decision procedure for a small class of properties defined in the logic with this operator. In particular, we are able to perform the model checking of formulas like $fn^{\subseteq}(N) \rhd B$ where N is a finite set of names and B is a formula which consists only of logical constants, disjunctions, inputs, outputs, revelations, hidings and reductions. The key point is that the partial model checking of the formula B does not introduce other operators (and moreover the hidings are removed)[8]. ■

5.1 An example

We consider as example the analysis of the protocol where a value is transmitted on a private channel between two users. We formally prove that a hostile environment X which does not know the value v , i.e.

[8] Actually, the partial model checking for revelation introduces some conjunctions with simple revelations. However, these can be simply treated by modifying the satisfiability procedure defined in the proof of Lemma 11.

$v \notin fn(X)$, is not able to retrieve it in communication with the users in the system. Consider the following processes:

$$\begin{aligned} P &= c_{PQ}\langle v\rangle \\ Q &= c_{PQ}(n).\mathbf{0} \\ Sys &= \nu c_{PQ}(P \,|\, Q) \end{aligned}$$

with $fn(Sys) = v$. We can prove that there exists no process X, with $v \notin fn(X)$ and $pub \in fn(X)$, s.t. $Sys \,|\, X \longrightarrow^* T \,|\, pub\langle v\rangle$. First, we compute $ml(Sys) = 2$. Then, we build $leaked_v^2$ and we perform the partial model checking. We illustrate only the partial model for the disjunct $\bigcirc^2 pub\langle v\rangle //_{Sys,\emptyset,\{pub\}}$, since the others are analyzed as particular instances of this one. Thus, we have

$$\begin{gathered} (\bigcirc \bigcirc pub\langle v\rangle)//_{Sys,\emptyset,\{pub\}} \doteq \\ \bigcirc pub\langle v\rangle //_{\nu C_{PQ}\mathbf{0},\emptyset,\{pub\}} \vee \bigcirc(\bigcirc pub\langle v\rangle //_{Sys,\emptyset,\{pub\}}) \end{gathered}$$

since the two unique possibilities are that Sys performs and internal reduction otherwise is X that perform such reduction. Then, we have

$$\begin{aligned} \bigcirc pub\langle v\rangle //_{\nu C_{PQ}\mathbf{0},\emptyset,\{pub\}} &\doteq \bigcirc(pub\langle v\rangle //_{\nu C_{PQ}\mathbf{0},\emptyset,\{pub\}}) \\ &\doteq \bigcirc \neg \mathbf{T} \end{aligned}$$

and

$$\begin{aligned} \bigcirc pub\langle v\rangle //_{Sys,\emptyset,\{pub\}} &\doteq \bigcirc(pub\langle v\rangle //_{\nu C_{PQ}\mathbf{0},\emptyset,\{pub\}}) \vee \bigcirc(pub\langle v\rangle //_{Sys,\emptyset,\{pub\}}) \\ &\doteq \bigcirc(\neg \mathbf{T}) \vee \bigcirc(\neg \mathbf{T}) \end{aligned}$$

since $pub\langle v\rangle //_{\nu C_{PQ}\mathbf{0},\emptyset,\{pub\}}$ is $\neg\mathbf{T}$ because $v \notin \{pub\}$ and $pub\langle v\rangle //_{Sys,\emptyset,\{pub\}}$ $\doteq \neg\mathbf{T}$ because Sys is not structurally equivalent to $pub\langle v\rangle \,|\, P'$ for any P' and $v \notin \{pub\}$.

Note that a formula like $\bigcirc(\neg\mathbf{T})$ is never satisfiable and thus it follows that $leaked_v^{ml(P)}$ is not satisfiable and we get the thesis.

6 Further work

In this paper, we performed some preliminary steps towards a compositional analysis framework for a nominal calculus, i.e. the π–calculus. So far, we have considered only finite π–calculus processes and a simple logic without recursion and without universal quantification. This clearly limits the range of application of such techniques. For dealing with universal quantification one could try to resort to infinite conjunctions or to a symbolic semantics for the π–calculus ([14]). A recent work [3] of Caires and Cardelli shows that the interplay between new name generation and recursion is rather complex and interesting. Whether it is possible to extend the partial model checking techniques to a calculus and a logic with some form of recursion deserves further investigation. However, we argue that the results in this paper and the semantics of Cardelli and Caires for the logic with recursion may be considered as building blocks for such a study. In particular, we plan to consider the partial model checking problem for finite-control processes which may have infinite behavior but whose model checking problem (actually for a different logic) may be solved (see [8]). On the other hand, some preliminary investigations show that it is possible to extend the same ideas applied in this paper to other nominal calculi such as the spi–calculus [1] and *ambient* calculus [4].

References

[1] M. Abadi and A. D. Gordon. A calculus for cryptographic protocols: The spi calculus. *Information and Computation*, 148(1):1–70, 1999.

[2] H. R. Andersen. Partial model checking (extended abstract). In *Proceedings of 10th Annual IEEE Symposium on Logic in Computer Science*, pages 398–407. IEEE Computer Society Press, 1995.

[3] L. Caires and L. Cardelli. A spatial logic for concurrency (Part I). In *Proc. Fourth International Symposium on Theoretical Aspects of Computer Science*, volume 2215 of *Lecture Notes in Computer Science*, 2001. To appear.

[4] L. Cardelli and A. Gordon. Mobile ambients. In *Proc. Foundations of Software Science and Computation Structures*, volume 1378 of *LNCS*, pages 140–155, 1998.

[5] L. Cardelli and A. D. Gordon. Anytime, anywhere: Modal logics for mobile ambients. In *Proceedings of the 27th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL-00)*, pages 365–377, N.Y., Jan. 19–21 2000. ACM Press.

[6] L. Cardelli and A. D. Gordon. Logical properties of name restriction. In *International Conference on Typed Lambda Calculi and Applications (TCLA 2001, Krakow, Poland)*, volume 2044 of *LNCS*, pages 46–60. Springer, 2001.

[7] W. Charatonik, S. Dal Zilio, A. D. Gordon, S. Mukhopadhyay, , and J.-M. Talbot. The complexity of model checking mobile ambients. In F. Honsell and M. Miculan, editors, *Proceedings of the 4th International Conference on Foundations of Software Science and Computation Structures (FoSSaCS 2001)*, volume 2030 of *LNCS*, pages 52–167. Springer, 2001.

[8] M. Dam. Model checking mobile processes. *Information and Computation*, 129(1):35–51, 1996.

[9] D. Dolev and A. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29(12):198–208, 1983.

[10] J. Engelfriet and T. Gelsema. Multisets and structural congruence of the π-calculus with replication. *Theoretical Computer Science*, 211(1–2):311–337, 1999. Previously published as Leiden University Report 95-02, 1995.

[11] R. Focardi, R. Gorrieri, and F. Martinelli. Non interference for the analysis of cryptographic protocols. In *Proceedings of 27th International Colloquium in Automata, Languages and Programming*, volume 1853 of *Lectures Notes in Computer Science*, pages 354–372, 2000.

[12] D. Kozen. Results on the propositional $\mu-$calculus. *Theoretical Computer Science*, 27(3):333–354, 1983.

[13] K. G. Larsen and L. Xinxin. Compositionality through an operational semantics of contexts. *Journal of Logic and Computation*, 1(6):761–795, 1991.

[14] H. Lin. Symbolic bisimulations and proof systems for the pi-calculus. Technical Report 94:07, School of Cognitive and Computing Sciences, University of Sussex, 1994.

[15] D. Marchignoli and F. Martinelli. Automatic verification of cryptographic protocols through compositional analysis techniques. In *Proceedings of the International Conference on Tools and Algorithms for the Construction and the Analysis of Systems (TACAS'99)*, volume 1579 of *Lecture Notes in Computer Science*, 1999.

[16] F. Martinelli. Analysis of security protocols as open systems. Technical Report IAT-B4-006, July 2001. Accepted for publication on TCS (under minor revisions).

[17] F. Martinelli. Encoding several security properties as properties of the intruder's knowledge. Technical Report IAT-B4-020, December 2001. Submitted for publication.

[18] F. Martinelli. Languages for description and analysis of authentication protocols. In *Proceedings of 6th ICTCS*, pages 304–315. World Scientific, 1998.

[19] F. Martinelli. Partial model checking and theorem proving for ensuring security properties. In *Proceedings of 11th Computer Security Foundations Workshop*, pages 44–52. IEEE Computer Society Press, 1998.

[20] R. Milner. *Communication and Concurrency.* International Series in Computer Science. Prentice Hall, 1989.

[21] R. Milner. *Communicating and Mobile Systems: the π-Calculus.* Cambridge University Press, 1999.

[22] R. Milner, J. Parrow, and D. Walker. A calculus of mobile processes. *Information and Computation*, 100(1):1–77, 1992.

[23] D. Sangiorgi. Extensionality and intensionality of the ambient logics. In *The 28th ACM Symposium on Principles of Programming Languages*, pages 4–13, 2001.

A Proofs

The *reduction* semantics is an elegant and compact way to describe the behavior of systems. However, the presence of *structural* congruence in its definition makes it difficult to perform the proofs with this technical machinery. Thus, many authors prefer to resort to the so-called *commitment* relation, which is the counterpart of the *labeled transition* semantics in the π–calculus. We adapt the description given in [1] to the asynchronous π–calculus. We need some new syntactic forms, in particular: *Abstractions*, *concretions*, and *agents*. An *abstraction* is term of the form $(x).P$ where x is bound and P is a process. Roughly, $(x).P$ waits a message y and behaves as $P[y/x]$. A *concretion*, is a term of the form $\nu N\langle m\rangle P$, where P is a process. Roughly, this is a process willing to send a message m (possibly restricted if $m \in N$) and then behaves as P. Finally, *agents* are either processes, abstractions, or concretions. We extend the restriction and composition operators to arbitrary agents as follows:

$$\begin{aligned}\nu m((x).P) &\doteq (x').\nu m(P[x'/x])\\ R\,|(x).P &\doteq (x').(R\,|\,P[x'/x])\end{aligned}$$

assuming that $x' \notin fn(R) \cup fn(P) \cup \{m\}$. For a concretion, we set:

$$\begin{aligned}\nu m\nu N\langle m'\rangle P &\doteq \begin{cases}\nu(\{m\} \cup N)\langle m'\rangle P & \text{if } m = m' \text{ or } m \in N\\ \nu N\langle m'\rangle \nu m P & \text{otherwise}\end{cases}\\ R\,|\,\nu N\langle m'\rangle P &\doteq \nu N'\langle m'[N'/N]\rangle(R\,|\,P[N'/N])\end{aligned}$$

assuming that $N' \cap (fn(R) \cup fn(P) \cup \{m'\}) = \emptyset$. We can define the dual composition $A\,|\,R$ symmetrically. Abstractions and concretions are complementary, thus, it is natural to define their composition, or *interaction* @, as follows:

$$\begin{aligned}F@C &\doteq \nu N(P[m'/x]\,|\,P')\\ C@F &\doteq \nu N(P'\,|\,P[m'/x])\end{aligned}$$

where $F = (x).P$ and $C = \nu N\langle m'\rangle P'$. An action is a name a or a co-name $\overline{a}$, or a distinguished *silent* action τ. The commitment relation is written as $P \xrightarrow{\alpha} A$, where P is a closed process, α is an action and A is a closed agent. We define this relation inductively as follows:

$$\frac{}{a(x).P \xrightarrow{a} (x).P} \tag{5}$$

$$\frac{}{a\langle m'\rangle \xrightarrow{\overline{a}} \nu\emptyset\langle m'\rangle} \tag{6}$$

$$\frac{P \xrightarrow{a} F \quad Q \xrightarrow{\overline{a}} C}{P\,|\,Q \xrightarrow{\tau} F@C} \tag{7}$$

$$\frac{P \xrightarrow{\overline{a}} C \quad Q \xrightarrow{a} F}{P\,|\,Q \xrightarrow{\tau} C@F} \tag{8}$$

$$\frac{P \xrightarrow{\alpha} A}{P\,|\,Q \xrightarrow{\alpha} A\,|\,Q} \tag{9}$$

$$\frac{Q \xrightarrow{\alpha} A}{P \,|\, Q \xrightarrow{\alpha} P \,|\, A} \tag{10}$$

$$\frac{P \xrightarrow{\alpha} A \quad \alpha \notin \{a, \overline{a}\}}{\nu a(P) \xrightarrow{\alpha} \nu a(A)} \tag{11}$$

$$\tag{12}$$

Note that $P \xrightarrow{a} F$ represents the fact that the process P is committed to receive a message through an action a. $Q \xrightarrow{\overline{a}} C$ represents the fact that the process Q is committed to send a message through an action $\overline{a}$ (which is the complementary action of a). Thus, the two processes may interact (this interaction is denoted by the silent action τ) by obtaining a process which consists of the abstraction F where x is replaced with m (the abstraction has been instantiated with the received value m) and the continuation of Q.

Lemma 12. *1. If $P \xrightarrow{a} (x).P^*$ then there are P_1, P_2, and names N_1 s.t. $a \notin N_1$, $P \equiv \nu N_1(a(x).P_1 \,|\, P_2)$, and $P^*[n/x] \equiv (\nu N_1(P_1 \,|\, P_2))[n/x]$ for any n.*

2. If $P \xrightarrow{\overline{a}} \nu N_1\langle n\rangle P^*$, then there is P_1 and names N_2 s.t. $a, n \notin N_2$, $a \notin N_1$ $P \equiv \nu N_1 \nu N_2(P_1 \,|\, a\langle n\rangle)$ and $P^* \equiv \nu N_2(P_1)$.

3. If $P \xrightarrow{\tau} P'$ then $P \longrightarrow P'$. ■

An interesting proposition relates the silent actions of the commitment relation and reductions.

Proposition 3. *$P \longrightarrow P''$ iff there exists P' s.t. $P \xrightarrow{\tau} P'$ and $P' \equiv P''$.* ■

This proposition is useful since permits to investigate the possible reductions of P, whose calculation involves the notion of structural congruence, to the calculations of the commitment relation for the silent actions, which can be calculated more easily.

Indeed, we show the proof of Lemma 8.

Lemma 8 *We have that $\nu N(P \,|\, Q) \longrightarrow R$ iff one of the following cases holds:*

1. *$P \longrightarrow P'$ and $R \equiv \nu N(P' \,|\, Q)$;*
2. *$P \equiv a\langle n\rangle \,|\, P', Q \equiv \nu N''(a(x).Q' \,|\, Q'')$, with $a, n \notin N''$, and $R \equiv \nu N(P' \,|$
 $\nu N''(Q'[n/x] \,|\, Q''))$;*

3. *$P \equiv \nu n(a\langle n\rangle \,|\, P'), Q \equiv \nu N''(a(x).Q' \,|\, Q'')$, with $a, n \notin N'', a \neq n, n \notin fn(Q)$ and $R \equiv \nu(N \cup \{n\})(P' \,|\, \nu N''(Q'[n/x] \,|\, Q''))$;*
4. *$Q \longrightarrow Q'$ and $R \equiv \nu N(P \,|\, Q')$;*
5. *$Q \equiv a\langle n\rangle \,|\, Q', P \equiv \nu N''(a(x).P' \,|\, P'')$, with $a, n \notin N''$, and $R \equiv \nu N(Q' \,|$
   $\nu N''(P'[n/x] \,|\, P''))$;*
6. *$Q \equiv \nu n(a\langle n\rangle \,|\, Q'), P \equiv \nu N''(a(x).P' \,|\, P'')$, with $a, n \notin N'', a \neq n, n \notin fn(P)$ and $R \equiv \nu N \cup \{n\}(Q' \,|\, \nu N''(P'[n/x] \,|\, P''))$.*

Proof. The *if* direction is easy and thus we concentrate on the *only if* one.

We can note that $\nu N(P \,|\, Q) \longrightarrow R$, then, by Proposition 3, there exists R_1 s.t. $\nu N(P \,|\, Q) \xrightarrow{\tau} R_1$ and $R_1 \equiv R$. Thus, by inspecting the operational rules for the commitment relation, it follows that $\nu N(P \,|\, Q) \xrightarrow{\tau} R_1$ iff $P \,|\, Q \xrightarrow{\tau} R_2$ and $R_1 = \nu N(R_2)$. We may have several cases depending on the last rule applied to infer the reduction:

- Rule 9. Then, we have $P \xrightarrow{\tau} P'$. By Lemma 12 (3), we have $P \longrightarrow P'$. Thus, the thesis follows (see case (1)).
- Rule 10. As above (see case (4)).
- Rule 7. Then, we have $P \xrightarrow{a} F = (x).P^*$ and $Q \xrightarrow{\overline{a}} C = \nu N_3\langle n\rangle Q^*$, with $F@C = \nu N_3(P^*[n/x] \,|\, Q^*)$.
  By Lemma 12 (1), we have $P \equiv \nu N_1(a(x).P_1 \,|\, P_2)$, with $a \notin N_1$, $P^*[n/x] = (\nu N_1(P_1 \,|\, P_2))[n/x]$. We can also assume that $x \notin N_1$, since $x \notin fn(P_2)$, as follows from the definition of abstraction for parallel composition. Thus, we have $P^*[n/x] \equiv \nu N_1(P_1[n/x] \,|\, P_2)$.
  By Lemma 12 (2) $Q \equiv \nu N_3 \nu N_2(Q_1 \,|\, a\langle n\rangle)$, with $a, n \notin N_2$, and $Q^* \equiv \nu N_2(Q_1)$.
  Thus, we have
  $$R_2 = F@C = \nu N_3(P^*[n/x] \,|\, Q^*)$$
  and
  $$R_1 = \nu N(R_2) = \nu N \nu N_3(P^*[n/x] \,|\, Q^*).$$
  We have two cases depending whether $n \in N_3$ or not:
  - $n \notin N_3$. Then N_3 can be considered as $\emptyset$. This situation is take into account by case (5). Indeed, let Q' be $\nu N_2(Q_1)$, P' be P_1 and P'' be P_2. As requested, we have $P \equiv \nu N_1(a(x).P' \,|\, P'')$ and:
  $$\begin{aligned} Q &\equiv \nu N_2(Q_1 \,|\, a\langle n\rangle) \\ &\equiv (\nu N_2(Q_1) \,|\, a\langle n\rangle) \\ &= Q' \,|\, a\langle n\rangle \end{aligned}$$

We finally need to show that $R \equiv \nu N(Q' \,|\, \nu N_1(P'[n/x] \,|\, P''))$. Note that $R \equiv R_1 = \nu N(R_2)$ and so:

$$\begin{aligned} R \equiv R_1 \equiv \nu N(R_2) \equiv &\ \text{def. of } R_2 \\ \nu N \nu N_3(P^*[n/x] \,|\, Q^*) \equiv &\ \text{def. of } P^*, Q^* \\ \nu N(\nu N_1(P_1[n/x] \,|\, P_2) \,|\, \nu N_2 Q_1) \equiv &\ \text{def. of } P', P'' \\ \nu N(\nu N_1(P'[n/x] \,|\, P'') \,|\, Q') &\end{aligned}$$

- $n \in N_3$. Thus, we can freely assume, by definition of restriction over concretions, that $N_3 = \{n\}$. This situation is take into account by case (6). Indeed, let Q' be $\nu N_2(Q_1)$, P' be P_1 and P'' be P_2. As requested, we have:

$$\begin{aligned} Q &\equiv \nu n \nu N_2(Q_1 \,|\, a\langle n \rangle) \\ &\equiv \nu n(\nu N_2(Q_1) \,|\, a\langle n \rangle) \\ &= \nu n(Q' \,|\, a\langle n \rangle) \end{aligned}$$

and P as above. We finally need to show that

$$R \equiv \nu N \nu n(Q' \,|\, \nu N_1(P'[n/x] \,|\, P''))$$

Note that $R \equiv R_1 = \nu N(R_1)$ and so:

$$\begin{aligned} R \equiv R_1 \equiv \nu N(R_2) \equiv &\ \text{def. of } R_2 \\ \nu N \nu N_3(P^*[n/x] \,|\, Q^*) \equiv &\ \text{def. of } P^*, Q^* \\ \nu N \nu n(\nu N_1(P_1[n/x] \,|\, P_2) \,|\, \nu N_2 Q_1) \equiv &\ \text{def. of } P', P'' \\ \nu N \nu n(\nu N_1(P'[n/x] \,|\, P'') \,|\, Q') &\end{aligned}$$

– Rule (8). As above (see cases (2) and (3)). ■

Proposition 1 *Let A be a formula in $\mathcal{L}_{\setminus\forall}$. Consider a finite set of names ϕ, a context $\nu N(P \,|\, (\_))$, with $Names(A) \cup N = \emptyset$, and process X with $fn(X) \subseteq \phi$. Then, we have:*

$$\nu N(P \mid X) \models A \text{ iff } X \models A//_{P,N,\phi}$$

where $A//_{P,N,\phi}$ is the formula defined in Tab. 3.

Proof. By induction on the structure of A:

– **T**. For every process X we have that $\nu N(P \,|\, X) \models \mathbf{T}$. Thus, **T** is the necessary and sufficient condition on X s.t. the whole composition satisfies **T**.

- $A \vee B$. Given a process X, we have that $\nu N(P \,|\, X) \models A \vee B$ iff $\nu N(P \,|\, X) \models A$ or $\nu N(P \,|\, X) \models B$. Thus, by structural induction, we have $\nu N(P \,|\, X) \models A$ ($\nu N(P \,|\, X) \models B$) iff $X \models A//_{P,N,\phi}$ ($X \models B//_{P,N,\phi}$). Thus, we have $\nu(P \,|\, X) \models A \vee B$ iff $X \models A//_{P,N,\phi} \vee B//_{P,N,\phi}$.
- $\neg A$. Analogous to the previous reasoning.
- $a\langle n\rangle A$. Given a process X, we have that $\nu N(P \,|\, X) \models a\langle n\rangle A$ iff $\nu N(P \,|\, X) \equiv a\langle n\rangle \,|\, Q'$ and $Q' \models A$. We might only consider two different cases: 1) The subterm $a\langle n\rangle$ belongs to X; 2) otherwise it belongs to P, i.e:
  - In the case 1), we have that $X \equiv a\langle n\rangle \,|\, X'$ and $\nu N(P \,|\, X') \models A$. By structural induction, we have that $\nu N(P \,|\, X') \models A$ iff $X' \models A//_{P,N,\phi}$. By putting together the conditions on X we get $X \models a\langle n\rangle(A//_{P,N,\phi})$.
    On the other hand, if $X \models a\langle n\rangle(A//_{P,N,\phi})$ then $\nu N(P \,|\, X) \models a\langle n\rangle A$.
  - In the case 2), we have that $P \equiv a\langle n\rangle \,|\, P'$ and $\nu N(P' \,|\, X) \models A$. By Lemma 4 (1), we know that for some $P'' \in C^o(a\langle n\rangle, P)$ we have $P'' \equiv P'$. Thus, by Lemma 1, $\nu N(P' \,|\, X) \models A$ iff $\nu N(P'' \,|\, X) \models A$. By structural induction, we have that $\nu N(P'' \,|\, X) \models A$ iff $X \models A//_{P'',N,\phi}$. Thus, $X \models A//_{P'',N,\phi}$ for some $P'' \in C^o(a\langle n\rangle, P)$. On the contrary if $X \models A//_{P'',N,\phi}$ for some $P'' \in C^o(a\langle n\rangle, P)$, then by structural induction, $\nu(P'' \,|\, X) \models A$. Moreover, by Lemma 4 (2), we have that $P \equiv P'' \,|\, a\langle n\rangle$, and so $\nu N(P \,|\, X) \models a\langle n\rangle A$.
- $a(n)A$. Given a process X, we have that $\nu N(P \,|\, X) \models a(n)A$ iff $\nu N(P \,|\, X) \equiv \nu N^*(a(m).Q' \,|\, Q'')$, with $a, n \notin N^*$, and $\nu N^*(Q'[n/m] \,|\, Q'') \models A$ holds. We may consider only the following two different cases: 1) The subterm $a(m).Q'$ belongs to X; 2) otherwise it belongs to P.
  - In the case 1), we have that $X \equiv \nu N_X(a(m).X' \,|\, X'')$, with $a, n \notin N_X$ and $a \in \phi$, for some X', X'' and possibly $n \in fn(X'[n/m])$. We can freely assume that $fn(P) \cap N_X = \emptyset$. As a matter of fact, we could always choose N_X s.t. the previous equality holds, because of $\alpha-$conversion of structural congruence and by the fact that $fn(P)$ is a finite set. Thus, we get:

$$\begin{aligned} &\nu N(P \,|\, X) \equiv \\ \nu N(P \,|\, &\nu N_X(a(m).X' \,|\, X'')) \equiv \\ \nu N \nu N_X&(a(m).X' \,|(P \,|\, X'')) \end{aligned}$$

and it must be that $\nu N \nu N_X(X'[n/m] \,|(P \,|\, X'')) \models A$. By structural induction, we know that

$$\begin{gathered}\nu N(P \,|\, \nu N_X(X'[n/m] \,|\, X'')) \models A \text{ iff} \\ \nu N_X(X'[n/m] \,|\, X'') \models A/\!/_{P,N,\phi\cup\{n\}}\end{gathered}$$

By putting together the conditions on X we get

$$X \models a(n)(A/\!/_{P,N,\phi\cup\{n\}})$$

On the other hand, if $X \models a(n)(A/\!/_{P,N,\phi\cup\{n\}})$, we have $X \equiv \nu N_X(a(m).X' \,|\, X'')$, with $a, n \notin N_X$ and $\nu N_X(X'[n/m] \,|\, X'') \models A/\!/_{P,N,\phi\cup\{n\}}$. By structural induction, we know that

$$\nu N(P \,|\, \nu N_X(X'[n/m] \,|\, X'') \models A$$

So, $\nu N(P \,|\, X) \equiv \nu N(P \,|\, \nu N_X(a(m).X' \,|\, X'')) \models a(n)A$ (recall that $a, n \notin N$).

- In the case 2), we have that $P \equiv \nu N'(a(m).P' \,|\, P'')$, with $a, n \notin N'$. We can freely assume that $N' \cap \phi = \emptyset$. We have $\nu N \nu N'((P'[n/m] \,|\, P'') \,|\, X) \models A$. By Lemma 6 (1), there exists $(N_1, P_1', P_2'') \in C^i(a(-), n, P)$ s.t.

$$\nu N'(P'[n/m] \,|\, P'') \equiv \nu N_1(P_1' \,|\, P_2'')$$

Thus, by Lemma 1, we have

$$\nu N(\nu N'(P'[n/m] \,|\, P'') \,|\, X) \models A \text{ iff } \nu N(\nu N_1(P_1' \,|\, P_2'') \,|\, X) \models A$$

and, by structural induction, it follows

$$\nu N(\nu N_1(P_1' \,|\, P_2'') \,|\, X) \models A \text{ iff } X \models A/\!/_{\nu N_1(P_1' \,|\, P_2''),N,\phi}$$

for some $(N_1, P_1', P_2'') \in C^i(a(-), n, P)$. On the other hand, if $X \models A/\!/_{\nu N_1(P_1' \,|\, P_2''),N,\phi}$ for some $(N_1, P_1', P_2'') \in C^i(a(-), n, P)$ then, by structural induction, we have $\nu N(\nu N_1(P_1' \,|\, P_2'') \,|\, X) \models A$. Moreover, by Lemma 6 (2), we have $P \equiv \nu N_1(a(m).P' \,|\, P'')$, with $a, n \notin N_1, P'[n/m] \equiv P_1'$ and $P'' \equiv P_2''$. Thus, $\nu N(P \,|\, X) \models a(n)A$ holds.

– $\bigcirc A$. Given a process X, we have $\nu N(P \,|\, X) \models A$ iff $\nu(P \,|\, X) \longrightarrow P_1$ and $P_1 \models A$. By Lemma 8, we may only have the following cases:
  1. The reduction is due to an internal reduction of P. Thus, consider a process P' s.t. $P \longrightarrow P'$. Then, $\nu N(P \,|\, X) \longrightarrow \nu N(P' \,|\, X)$ and $\nu N(P' \,|\, X) \models A$. By structural induction, we have $\nu N(P' \,|\, X) \models A$ iff $X \models A/\!/_{P',N,\phi}$. On the other hand, if $X \models A/\!/_{P',N,\phi}$ for some P' s.t. $P \longrightarrow P'$, then we have $\nu N(P \,|\, X) \models \bigcirc A$.

2. The reduction is due to an output of P. Thus, we have $P \equiv a\langle n\rangle \,|\, P', X \equiv \nu N_X(a(m).X' \,|\, X'')$, with $a, n \notin N_X$ and possibly $n \in fn(X'[n/m])$, and $\nu N(P' \,|\, \nu N_X(X'[n/m] \,|\, X'')) \models A$. By Lemma 4, we know that there exists $P_1 \in C^o(a\langle n\rangle, P)$ s.t. $P' \equiv P_1$. Thus, by Lemma 1, we have

$$\begin{array}{c}\nu N(P' \,|\, \nu N_X(X'[n/m] \,|\, X'')) \models A \ \text{ iff} \\ \nu N(P_1 \,|\, \nu N_X(X'[n/m] \,|\, X'')) \models A\end{array}$$

and, by structural induction, we have

$$\begin{array}{c}\nu N(P_1 \,|\, \nu N_X(X'[n/m] \,|\, X'')) \models A \ \text{ iff} \\ \nu N_X(X'[n/m] \,|\, X'') \models A/\!/_{P_1, N, \phi\cup\{n\}}\end{array}$$

By putting together the conditions on X we get that, for some $P_1 \in C^o(a\langle n\rangle, P)$, $X \models a(n)(A/\!/_{P_1,N,\phi\cup\{n\}})$. On the other hand, if $X \models a(n)(A/\!/_{P_1,N,\phi\cup\{n\}})$, for some $P_1 \in C^o(a\langle n\rangle, P)$, then $\nu N(P \,|\, X) \models a(n)A$ holds.
3. The reduction is due to an output of a restricted name of P. Thus, we have $P \equiv \nu n(a\langle n\rangle \,|\, P'), X \equiv \nu N_X(a(m).X' \,|\, X'')$, with $n \notin fn(X)$, $a \neq n$, and

$$\nu N \cup \{n\}(P' \,|\, \nu N_X(X'[n/m] \,|\, X'')) \models A$$

We can freely assume that $n \notin (fn(P) \cup \phi \cup N \cup Names(A))$, since n is a restricted name. By Lemma 5, we know that there exists $P_1 \in C^{ro}(a\langle n\rangle, P)$ s.t. $P' \equiv P_1$. Thus, by Lemma 1, we have:

$$\begin{array}{c}\nu N \cup \{n\}(P' \,|\, \nu N_X(X'[n/m] \,|\, X'')) \models A \ \text{ iff} \\ \nu N \cup \{n\}(P_1 \,|\, \nu N_X(X'[n/m] \,|\, X'')) \models A\end{array}$$

and, by structural induction, we have:

$$\begin{array}{c}\nu N \cup \{n\}(P_1 \,|\, \nu N_X(X'[n/m] \,|\, X'')) \models A \ \text{ iff} \\ \nu N_X(X'[n/m] \,|\, X'') \models A/\!/_{P_1, N\cup\{n\}, \phi\cup\{n\}}\end{array}$$

By grouping the conditions on X we get that, for some $P_1 \in C^{ro}(a\langle n\rangle, P)$, $X \models a(n)(A/\!/_{P_1,N\cup\{n\},\phi\cup\{n\}})$. Note that we need to consider a single name n with the previous properties.
On the other hand, if

$$X \models a(n)(A/\!/_{P_1,N\cup\{n\},\phi\cup\{n\}})$$

for some $P_1 \in C^{ro}(a\langle n\rangle, P)$, then $\nu N(P \,|\, X) \models \bigcirc A$ holds.

4. The reduction is due to an internal reduction of X. Thus, consider that $X \longrightarrow X'$ and $\nu(P \mid X') \models A$. By structural induction, we have $\nu(P \mid X') \models A$ iff $X' \models A/\!/_{P,N,\phi}$. Thus, the condition on X is expressed by the formula $X \models \bigcirc(A/\!/_{P,N,\phi})$.
5. The reduction is due to an output of X. Thus, we have $X \equiv a\langle n\rangle \mid X', P \equiv \nu N'(a(m).P' \mid P'')$, with $a, n \notin N'$, and $\nu N(\nu N'(P'[n/m] \mid P'') \mid X') \models A$. By Lemma 6 (1), there exists $(N_1, P_1', P_2'') \in C^i(a(-), n, P)$ s.t. $P'[n/m] \equiv P_1', P'' \equiv P_2''$ and $\nu N'(P'[n/m] \mid P'') \equiv \nu N_1(P_1' \mid P_2'')$. By Lemma 1 (1), we have:
$$\nu N(\nu N'(P'[n/m] \mid P'') \mid X') \models A \text{ iff } \nu N(\nu N_1(P_1' \mid P_2'') \mid X') \models A$$
and, by structural induction, we get
$$\nu N(\nu N_1(P_1' \mid P_2'') \mid X') \models A \text{ iff } X' \models A/\!/_{\nu N_1(P_1', P_2''),N,\phi}$$
By grouping the conditions on X we get that, for some $(N_1, P_1', P_2'') \in C^i(a(-), n, P)$, $X \models a\langle n\rangle(A/\!/_{\nu N_1(P_1', P_2''),N,\phi})$. On the other hand, if $X \models a\langle n\rangle(A/\!/_{\nu N_1(P_1', P_2''),N,\phi})$, then we have that $\nu N(P \mid X) \models \bigcirc A$ holds.
6. The reduction is due to an output of a restricted name of X. Thus, we have $X \equiv \nu n(a\langle n\rangle \mid X')$, with $a \neq n$ and possibly $n \in fn(X'), P \equiv \nu N'(a(m).P' \mid P'')$, with $a, n \notin N'$, and
$$\nu N \cup \{n\}(\nu N'(P'[n/m] \mid P'') \mid X') \models A$$
Note that we can freely assume that $n \notin (fn(P) \cup \phi \cup N \cup Names(A))$.
By Lemma 6 (1), there exists $(N_1, P_1', P_2'') \in C^i(a(-), n, P)$ s.t. $P'[n/m] \equiv P_1', P'' \equiv P_2''$ and $\nu N'(P'[n/m] \mid P'') \equiv \nu N_1(P_1' \mid P_2'')$. By Lemma 1 (1), we have:
$$\begin{gathered}\nu N \cup \{n\}(\nu N'(P'[n/m] \mid P'') \mid X') \models A \text{ iff} \\ \nu N \cup \{n\}(\nu N_1(P_1' \mid P_2'') \mid X') \models A\end{gathered}$$
and, by structural induction, we know that:
$$\begin{gathered}\nu N \cup \{n\}(\nu N_1(P_1' \mid P_2'') \mid X') \models A \text{ iff} \\ X' \models A/\!/_{\nu N'(P_1', P_2''),N\cup\{n\},\phi\cup\{n\}}\end{gathered}$$
By grouping the conditions on X we get that, for some $(N_1, P_1', P_2'') \in C^i(a(-), n, P)$, $X \models n \cdot_R a\langle n\rangle(A/\!/_{\nu N_1(P_1', P_2''),N\cup\{n\},\phi\cup\{n\}})$. On the contrary if $X \models n \cdot_R a\langle n\rangle(A/\!/_{\nu N_1(P_1', P_2''),N\cup\{n\},\phi\cup\{n\}})$, we can check that $\nu N(P \mid X) \models \bigcirc A$.

- $A{\cdot}_R n$. By definition we have $\nu N(P \mid X) \models A{\cdot}_R n$ iff $\nu n \nu N(P \mid X) \models A$. Let n' be a name s.t. $n' \notin (fn(P) \cup \phi \cup N \cup Names(A))$. Then, by Lemma 2, we have

$$\nu n \nu N(P \mid X) \models A \text{ iff } \nu n \nu N(P \mid X) \models A[n'/n]$$

  By induction hypothesis, it follows that $\nu n \nu N(P \mid X) \models A[n'/n]$ iff $X \models A([n'/n])//_{P,N\cup\{n\},\phi}$.
  On the other hand, if $X \models A([n'/n])//_{P,N\cup\{n\},\phi}$ then, by induction hypothesis, we have that $\nu N \cup \{n\}(P \mid X) \models A[n'/n]$. Since, n and n' are not a free names of $\nu N \cup \{n\}(P \mid X)$, we get

$$\begin{aligned} \nu N \cup \{n\}(P \mid X) \models A[n'/n] &\text{ iff} \\ \nu N \cup \{n\}(P \mid X)[n/n'] \models A[n'/n][n/n'] &\text{ iff} \\ \nu N \cup \{n\}(P \mid X) \models A &\text{ iff} \\ \nu N(P \mid X) \models A \cdot_R n & \end{aligned}$$

- $n \cdot_R A$. By definition we have $\nu N(P \mid X) \models n \cdot_R A$ iff $\nu N(P \mid X) \equiv \nu n(P')$ and $P' \models A$. Note that n cannot be a free name of $\nu N(P \mid X)$, i.e. $n \notin (fn(P) \cup fn(X)) \setminus N$ (recall that n does not occur in N). We only need to consider three different cases:
  1. n is a name in N (up to renaming since, by assumption, in $n \cdot_R A$ cannot be names in N). Thus, we have $P' \equiv (\nu N \setminus \{n'\}(P \mid X))[n/n']$ and $(\nu N \setminus \{n'\}(P \mid X))[n/n'] \models A$. (We are performing an α–renaming of n' with n.)
     By Lemma 2, we know that:

$$\nu N\setminus\{n'\}(P \mid X)[n/n'] \models A \text{ iff } \nu N\setminus\{n'\}(P \mid X)[n/n'][n'/n] \models A[n'/n]$$

     since $n' \in N$ and, by assumption, in $n \cdot_R A$ there are no names also in N. Note that $\nu N \setminus \{n'\}(P \mid X)[n/n'][n'/n] \equiv \nu N \setminus \{n'\}(P \mid X)$. By structural induction, we have:

$$\nu N \setminus \{n'\}(P \mid X) \models A[n'/n] \;\text{ iff }\; X \models A[n'/n]//_{P,N\setminus\{n'\},\phi}$$

     We need to consider every possible name n' in N. We must have also that $n \notin fn(X) \setminus N$; since $n \notin N$ by assumption, we can simply require that $n \notin fn(X)$, which can be logically expressed.
  2. n is a restricted name of X. Thus, we must have $X \equiv \nu n X'$, with $n \notin fn(P)$, and $P' \equiv \nu N(P \mid X') \models A$ (since $n \notin N$) with possibly $n \in fn(X')$. By structural induction we have $X' \models A//_{P,N,\phi\cup\{n\}}$ and so $X \models n \cdot_R (A//_{P,N,\phi\cup\{n\}})$. On the other hand, if $X \models n \cdot_R$

$(A/\!/_{P,N,\phi\cup\{n\}})$, and $n \notin fn(P) \cup N$, then it must be that $X \equiv \nu n X'$, and $X' \models (A/\!/_{P,N,\phi\cup\{n\}})$. By structural induction, we have $\nu N(P \,|\, X') \models A$ and so $\nu n \nu N(P \,|\, X') \equiv \nu N(P \,|\, X) \models n \cdot_R A$, since $n \notin fn(P) \cup N$.

3. n is a restricted name of P. First, it must be $P \equiv \nu n P_1$, with $n \notin fn(X)$, and $P' \equiv \nu N(P_1 \,|\, X)$. By Lemma 3, we know that there exists $P_1' \in C^{res}(P, n)$ s.t. $P_1 \equiv P_1'$. Thus, by Lemma 1, we have $\nu N(P_1 \,|\, X) \models A$ iff $\nu N(P_1' \,|\, X) \models A$. By structural induction, we have:
$$\nu N(P_1' \,|\, X) \models A \ \text{ iff } X \models A/\!/_{P_1',N,\phi}$$
On the other hand, if $X \models A/\!/_{P_1',N,\phi}$, for some $P_1' \in C^{res}(P, n)$ and $n \notin fn(P) \cup N$, then it follows $\nu n \nu N(P_1' \,|\, X) \equiv \nu N(P \,|\, X) \models n \cdot_R A$.

– $\mathbf{0}$. By definition we have $\nu N(P \,|\, X) \models \mathbf{0}$ iff $\nu N(P \,|\, X) \equiv \mathbf{0}$. This is possible if and only if $P \,|\, X \equiv \mathbf{0}$. Thus, it must be $P \equiv \mathbf{0}$ and $X \equiv 0$.

– $A \,|\, B$. By definition, we have $\nu N(P \,|\, X) \models A \,|\, B$ iff $\nu N(P \,|\, X) \equiv P' \,|\, P''$ and $P' \models A$ and $P'' \models B$. It must be that $P \,|\, X \equiv Q' \,|\, Q''$ and $fn(Q') \cap N = \emptyset$ (otherwise it is not possible that $\nu N(P \,|\, X) \equiv P' \,|\, P''$). Q' may consist of parts of both P and X. Thus $P \equiv P_1 \,|\, P_2, X \equiv X_1 \,|\, X_2$ and $fn(P_1) \cap N = \emptyset = fn(X_1) \cap N$. Thus, $\nu N(P \,|\, X) \equiv (P_1 \,|\, X_1) \,|\, \nu N(P_2 \,|\, X_2)$. We have to consider that either:
  - $P_1 \,|\, X_1 \models A$ and $\nu N(P_2 \,|\, X_2) \models B$, or
  - $P_1 \,|\, X_1 \models B$ and $\nu N(P_2 \,|\, X_2) \models A$.

  Consider a pair (P_1', P_2') in $C^{comp}(P)$ s.t. $P_1 \equiv P_1'$ and $P_2 \equiv P_2'$. Then, by Lemma 1, we have either:
  - $P_1' \,|\, X_1 \models A$ and $\nu N(P_2' \,|\, X_2) \models B$, or
  - $P_1' \,|\, X_1 \models B$ and $\nu N(P_2' \,|\, X_2) \models A$.

  By structural induction, we have either:
  - $X_1 \models A/\!/_{P_1',\emptyset,\phi\setminus N}$ and $X_2 \models B/\!/_{P_2',N,\phi}$, or
  - $X_1 \models B/\!/_{P_1',\emptyset,\phi\setminus N}$ and $X_2 \models A/\!/_{P_2',N,\phi}$.

  Note also that it must be $X_1 \models fn^{\neg}(N)$. Thus, by grouping together the previous conditions on X we obtain:
$$\begin{aligned} X \models\ & (fn^{\neg}(N) \wedge A/\!/_{P_1',\emptyset,\phi\setminus N} \,|\, B/\!/_{P_2',N,\phi}) \vee \\ & (fn^{\neg}(N) \wedge B/\!/_{P_1',\emptyset,\phi\setminus N} \,|\, A/\!/_{P_2',N,\phi}) \end{aligned}$$

– $A \wedge fn^{\subseteq}(N_1) \triangleright B$. Recall that $N_1 \cap N = \emptyset$. By definition, we have $\nu N(P \,|\, X) \models A \triangleright B$ iff for all Q s.t. $Q \models A$ and $fn(Q) \subseteq N_1$

we have that $\nu N(P \,|\, X) \,|\, Q \models B$ holds. We have $\nu N(P \,|\, X) \,|\, Q \equiv \nu N(P \,|(X \,|\, Q))$. We apply the induction hypothesis and obtain that:

$$\begin{aligned} \nu N(P \,|(X \,|\, Q)) \models B &\text{ iff} \\ X \,|\, Q \models B//_{P,N,\phi \cup N_1} &\text{ iff} \\ X \models (A \wedge fn^{\subseteq}(N_1)) \lhd (B//_{P,N,\phi \cup N_1}) & \end{aligned}$$

Lemma 9 *If $\nu N(P \,|\, X) \longrightarrow^* Q \,|\, pub\langle v \rangle$ then we can find a process X' s.t. $\nu N(P \,|\, X') \longrightarrow^* Q' \,|\, pub\langle v \rangle$, with $fn(X) = fn(X')$ and X' during this computation does not perform any internal reduction.*

Proof. By induction on the length n of the computation.

- $n = 0$. We are done, let X' be X.
- *Induction step.* Assume that

$$\nu N(P \,|\, X) \longrightarrow \nu N_1(P_1 \,|\, X_1) \longrightarrow^* Q \,|\, pub\langle v \rangle$$

  We may have three different cases depending on the nature of the first internal communication:
  1. The first communication is internal to P. Then we have $X_1 = X$. By inductive hypothesis we have that there exists a process X_1' s.t. $\nu N_1(P_1 \,|\, X_1') \longrightarrow^* Q \,|\, pub\langle v \rangle$. So, simply consider X' as X_1'.
  2. The first communication is due to an interaction between P and X. This interaction may be due only for the following reasons:
     - $X \equiv c\langle m \rangle \,|\, X_2$ and $X_1 = X_2$. We have that $\nu N_1(P_1 \,|\, X_2) \longrightarrow^* Q \,|\, pub\langle v \rangle$. Thus, by inductive hypothesis, we have that there exists X_2' s.t. $\nu N_1(P_1 \,|\, X_2') \longrightarrow^* Q' \,|\, pub\langle v \rangle$. We consider X' as $c\langle m \rangle \,|\, X_2'$.
     - The case when m is restricted is similar.
     - $X \equiv \nu N_X(c(m).X' \,|\, X_2), P \equiv c\langle n \rangle \,|\, P_1$, with $c, n \notin N_X$, and $X_1 = \nu N_X(X'[n/m] \,|\, X_2)$. Thus, by inductive hypothesis, there exists a process X_1' s.t. $\nu N_1(P_1 \,|\, X_1') \longrightarrow^* Q' \,|\, pub\langle v \rangle$. Now consider as $X' = c(m).X_1''$, where $X_1'' = X_1'[m/n]$. Then we get the thesis.
     - The case when n is restricted is similar.
  3. The first communication is due only to an action of X. Then, by inductive hypothesis, we have that there exists X_1' s.t. $\nu N(P \,|\, X_1') \longrightarrow^* Q' \,|\, pub\langle v \rangle$. Consider X' as X_1'. ■

Lemma 10 *Assume that $\nu N'(P \,|\, X) \longrightarrow^* Q \,|\, pub\langle v\rangle$, with $pub, v \notin N'$ and $v \notin fn(X)$. Then, there exists X', with $fn(X') \subseteq (fn(P) \cup \{pub\}) \setminus \{v\}$, s.t.:*

$$\nu N'(P \,|\, X') \longrightarrow^* Q \,|\, pub\langle v\rangle.$$

Proof. Indeed, assume that $\nu N'(P \,|\, X) \longrightarrow^* Q \,|\, pub\langle v\rangle$; then to study the confidentiality of the value v in P is equivalent to study the confidentiality of $\nu N \nu N'(P \,|\, X) \longrightarrow^* Q' \,|\, pub\langle v\rangle$, provided that $pub, v \notin N$. Now, recall that v is not a free name of X. So, let N be $fn(X) \setminus \{pub\}$. Thus, also $T = \nu fn(X) \setminus \{pub\} \nu N'(P \,|\, X) \longrightarrow^* Q'' \,|\, pub\langle v\rangle$. Now, by applying the rule for the structural congruence, we get that $T \equiv T' = \nu fn(P) \nu N'(P \,|\, \nu fn(X) \setminus \{pub\} \setminus fn(P) X) \longrightarrow^* Q'' \,|\, pub\langle v\rangle$. Thus, let X' be $\nu fn(X) \setminus \{pub\} \setminus fn(P) X$. Note that $fn(X') \subseteq (fn(P) \cup \{pub\}) \setminus \{v\}$. ∎

Lemma 11 *Let A be a formula which consists only of logical constants, disjunctions, inputs, outputs, revelations and reductions. Then, the problem of establishing whether or not there exists a process X, with $fn(X) \subseteq \phi$, s.t. $X \models A$ is decidable.*

Proof. By induction on the depth of A. If A is satisfiable we are able to construct a model of it, call it X_A.

- $A = \mathbf{T}$. Every process X is a model of this formula. Let X_A be $\mathbf{0}$.
- $A = \mathbf{F}$. No process X is a model of this formula.
- $A = a\langle n\rangle A'$. We have that A is satisfiable by a process X with free names in ϕ iff $X \equiv a\langle n\rangle \,|\, X'$ iff $a, n \in \phi$ and A' is satisfiable by a process X' whose free names are contained in ϕ. By induction hypothesis, we are able to establish whether or not such process X' exists. If so, let $X_A = a\langle n\rangle \,|\, X_{A'}$.
- $A = a(n)A'$. We have that A is satisfiable by a process X with free names in ϕ iff $X \equiv \nu N(a(m)X_1 \,|\, X_2)$, with $a, n \notin N$, and $\nu N(X_1[n/m] \,|\, X_2) \models A'$. Thus, we have A is satisfiable iff $a \in \phi$ and A' is satisfiable by a process X' with free names in $\phi \cup \{n\}$. By induction hypothesis, we are able to establish whether or not such process X' exists. If so, it must be that $X_{A'} \equiv \nu N(X_1' \,|\, X_2')$, where $n \in fn(X_1')$ and $n \notin N$. Thus, consider $X_A = \nu N(a(m).X_1'[m/n] \,|\, X'')$.

- $A = n \cdot_R A'$. We have that A is satisfiable by a process X whose free names are contained in ϕ iff $X \equiv \nu n X'$ and $X' \models A'$. Note that possibly $fn(X') \subseteq \phi \cup \{n\}$. By induction hypothesis, we are able to establish whether or not such process X' exists. If so, let $X_A = \nu n X_{A'}$. Then, we have that $X_A \models A$ and $fn(X_A) \subseteq \phi$.
- $A = \bigcirc A'$. We have that A is satisfiable by a process X whose free names are contained in ϕ iff $X \longrightarrow X'$ and $X' \models A'$. Note that $fn(X') \subseteq fn(X) \subseteq \phi$. By induction hypothesis, we are able to establish whether or not such process X' exists. On the one hand, assume that $X_{A'}$ is a model of A'. Then, let n be s.t. $n \notin \phi$. Then, $X_A = \nu n(n\langle n\rangle \mid n(n).X_{A'})$ is able to perform a reduction by reaching a process structurally equivalent to $X_{A'}$. On the other hand, if A' is not satisfiable, then also A is not satisfiable.
- $A = A_1 \vee A_2$. A is satisfiable iff either A_1 or A_2 are satisfiable. The proof proceeds by induction induction. ■