ABOUT COMPOSITIONAL ANALYSIS OF π–CALCULUS PROCESSES*

Fabio Martinelli
Istituto per le Applicazioni Telematiche - C.N.R., Pisa, Italy.
Fabio.Martinelli@iat.cnr.it

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.

Keywords: π-calculus, partial model checking, ambient logic, confidentiality.

1. Introduction

The π–calculus (Milner et al., 1992) 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:

$$\begin{array}{rcl} P & \doteq & c\langle n\rangle \\ Q & \doteq & c(y).y\langle m\rangle \end{array}$$

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

*This is an extended abstract of (Martinelli, a). Work partially supported by Microsoft Research Europe (Cambridge); by MIUR project "MEFISTO"; by MIUR project " Tecniche e strumenti software per l'analisi della sicurezza delle comunicazioni in applicazioni telematiche di interesse economico e sociale"; by CNR project "Strumenti, ambienti ed applicazioni innovative per la società dell'informazione" and finally by CSP with the project "SeTAPS".

R. Baeza-Yates et al. (eds.), *Foundations of Information Technology in the Era of Network and Mobile Computing* © Springer Science+Business Media New York 2002

name m on that channel. The parallel composition of P and Q, i.e. $P \mid Q$, 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 $P \mid Q \longrightarrow n\langle m\rangle$. Thus, the process Q now is able to communicate on the 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 Andersen, 1995; Larsen and Xinxin, 1991) 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.

There are several verification problems where the compositional analysis provided by partial model checking is particularly useful. In particular, we consider here the analysis of security protocols. The verification scenario for such protocols is to check 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 Focardi et al., 2000). 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 (Martinelli, b), we can state security properties as:

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

and then apply partial model checking techniques to reduce such a 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, and properties expressed with modal logics as the Hennessy-Milner one or the μ–calculus.

In this paper, as logic for describing the process properties, we adopt a restriction of the *ambient* logic developed in (Cardelli and Gordon, 2000; Cardelli and Gordon, 2001) to the π–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.

Organization of the paper. In Section 2, we briefly recall the asynchronous (finite) π–calculus. In Section 3, we introduce the logic we use, i.e. a restriction of the ambient logic to the π–calculus. Section 4 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. In Section 6, we discuss about some further work.

2. Asynchronous π–calculus

In this section we briefly recall some basic concepts about the asynchronous π–calculus.

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:

$$P, Q ::= \mathbf{0} \mid a\langle n\rangle \mid a(n).P \mid \nu n\, P \mid P \,|\, Q$$

The name n is said bound in the terms $(\nu n)P$ and $a(n).P$. The set $fn(P)$ of free names of P is defined as usual.

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. 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 $\nu n_1..\nu n_j(P)$ 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:

$$\begin{array}{cc}
\dfrac{}{a\langle n\rangle \,|\, a(m).P \longrightarrow P[n/m]} & \dfrac{P \equiv Q, Q \longrightarrow Q', Q' \equiv P'}{P \longrightarrow P'} \\
\dfrac{P \longrightarrow P'}{P\,|\,Q \longrightarrow P'\,|\,Q} & \dfrac{P \longrightarrow P'}{\nu n(P) \longrightarrow \nu n(P')}
\end{array}$$

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 π–processes. This is a restriction of the ambient logic of Cardelli and Gordon, 2000; Cardelli and Gordon, 2001 to π–calculus[1]. The syntax of formulas is as follows:

$$\begin{array}{lcl}
A & ::= & \mathbf{T} \mid \neg A \mid A_1 \vee A_2 \mid \eta\langle\eta'\rangle A \mid \eta(\eta')A \mid \bigcirc A \mid A \cdot_R \eta \mid \eta \cdot_R A \mid \\
 & & \mathbf{0} \mid A\,|\,B \mid A \triangleright B \mid \forall x A
\end{array}$$

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

The logic permits us to express both temporal and spatial properties of processes; moreover it allows to treat with restricted names in a convenient way. The logic, besides the usual constants and operators of propositional logic, has three modalities for expressing the temporal behavior of processes:

- $\eta\langle\eta'\rangle A$ (*output*). This formula expresses a process may send the name η' on the channel η and then it satisfies A.

[1] Recently in (Caires and Cardelli, 2001), Cardelli and Caires adapted several concepts of the ambient logic to the π–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).

- $\eta(\eta')A$ (*input*). This formula expresses that a process may receive the name η' on the channel η and then it satisfies A.
- $\bigcirc A$ (*reduction*). This formula expresses that a process performs a reduction (an internal communication) and then it satisfies A[2].

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

- $\mathbf{0}$ (*zero*). This formula requires that the process is structurally equivalent to $\mathbf{0}$.
- $A \mid B$ (*composition*). 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.
- $A \triangleright B$ (*adjunct of the composition*). 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.

- $A \cdot_R n$ (*hiding*). This formula expresses that a process, after the restriction of the name n enjoys A.
- $n \cdot_R A$ (*revelation*). This formula expresses that a process is 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. 1. As usual, we define $\exists x A$ as $\neg\forall x \neg A$, $A \wedge B$ as $\neg(\neg A \vee \neg B)$, $A \implies B$ as $\neg A \vee B$ and $\mathbf{F} = \neg\mathbf{T}$. Let $Names(A)$ be the set of all names appearing in a formula A.

We give some examples of the usage of the logic for expressing properties of π–calculus processes.

Example 1 *To express that a process P emits a restricted name, we can check whether $P \models \exists y \exists x(\neg(y \cdot_R T) \wedge x \cdot_R y\langle x\rangle)$. The revelation operator picks a restricted name of P, if any, call it x and then checks if x is emitted on some channel y. Note, that we can reveal a name x only if x is not a free name of the process. Indeed, the formula $x \cdot_R \mathbf{T}$ states that x is not free in a process (and thus it can be revealed). Similarly, if we are not able to reveal y, it means that y is free in P.* ■

[2] In (Cardelli and Gordon, 2001), 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).

$P \models \mathbf{T}$		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 1. Formal semantics of the logic.

Example 2 *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:* $\forall X \quad P \mid X \models B$ *as* $P \models \mathbf{T} \triangleright B$ ■

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 3 *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* $P \mid X \models A$ *iff there exists* P' *s.t.* $P \mid X \equiv n\langle n' \rangle \mid P'$ *and* $P' \models \mathbf{T}$. *Note that* $P \mid X \equiv n\langle n' \rangle \mid P'$ *iff* P' *is* X, *or* $P' \equiv P \mid X'$ *and* $X \equiv n\langle n' \rangle \mid X'$. *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 Andersen, 1995), 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 Andersen, 1995; Larsen and Xinxin, 1991). Note that it is possible to give a semantics of the π–calculus in terms of 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 in Engelfriet and Gelsema, 1999.

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 1 *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 the 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 2 *Given a process P, we can effectively compute a finite set of processes $C^o(a\langle n\rangle, P)$ s.t.:*

1 $P \equiv a\langle n\rangle \,|\, P'$ 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 $P \equiv a\langle n\rangle \,|\, P''$. ■

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 3 *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 that 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 residual after the communication. This gives us all the possible continuations after the reception of the value n.

Lemma 4 *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 that 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 5 *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 6 *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* $R \equiv \nu N(P \mid Q')$;

5 $Q \equiv a\langle n\rangle \mid Q', P \equiv \nu N''(a(m).P' \mid P'')$, *with* $a, n \notin N''$, *and* $R \equiv \nu N(Q' \mid \nu N''(P'[n/m] \mid P''))$;

6 $Q \equiv \nu n(a\langle n\rangle \mid Q'), P \equiv \nu N''(a(m).P' \mid P'')$, *with* $a, n \notin N'', a \neq n, n \notin fn(P)$ *and* $R \equiv \nu N \cup \{n\}(Q' \mid \nu N''(P'[n/m] \mid P''))$. ■

$$
\begin{array}{lcl}
\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} \\
 & & A_1 \doteq a\langle n\rangle(A/\!/_{P,N,\phi}) \quad \text{if } \{a,n\} \subseteq \phi \\
 & & A_2 \doteq \bigvee_{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} \\
 & & A_1 \doteq a(n)(A/\!/_{P,N,\phi \cup \{n\}}) \quad \text{if } a \in \phi \\
 & & A_2 \doteq \bigvee_{(N',P',P'') \in C^i(a(-),n,P)} A/\!/_{\nu N'(P'|P''),N,\phi} \\
(\bigcirc A)/\!/_{P,N,\phi} & \doteq & \bigvee_{P':P \longrightarrow P'} A/\!/_{P',N,\phi} \\
 & \vee & \bigvee_{a \in \phi} \bigvee_{n \in fn(P)} \bigvee_{P' \in C^o(a\langle n\rangle,P)} a(n)(A/\!/_{P',N,\phi \cup \{n\}}) \\
 & \vee & \bigvee_{a \in \phi} \bigvee_{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 & \bigvee_{a,n \in \phi} \bigvee_{(N_1,P',P'') \in C^i(a(-),n,P)} a\langle n\rangle(A/\!/_{\nu N_1(P'|P''),N,\phi}) \\
 & \vee & \bigvee_{a \in \phi} \bigvee_{(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'\}}) \\
 & & \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} \\
 & & \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} \\
 & & A_1 \doteq n \cdot_R \mathbf{T} \wedge (\bigvee_{n' \in N}(A[n'/n])/\!/_{P,N \setminus \{n'\},\phi}) \text{ if } n \notin fn(P) \setminus N \\
 & & A_2 \doteq n \cdot_R (A/\!/_{P,N,\phi \cup \{n\}}) \text{ if } n \notin fn(P) \cup N \\
 & & A_3 \doteq n \cdot_R \mathbf{T} \wedge (\bigvee_{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 & \bigvee_{(P',P'') \in C^{comp}(P), fn(P') \cap N = \emptyset} \\
 & & (fn^{\neg}(N) \wedge A/\!/_{P',\emptyset,\phi \setminus N}) \mid (B/\!/_{P'',N,\phi}) \vee \\
 & & (fn^{\neg}(N) \wedge B/\!/_{P',\emptyset,\phi \setminus N}) \mid (A/\!/_{P'',N,\phi}) \\
 & & \text{where } fn^{\neg}(\{n_1,\ldots,n_k\}) \doteq \bigwedge_{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}) \\
 & & \text{where } A' = A \wedge fn^{\subseteq}(N_1)
\end{array}
$$

Table 2. 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 $\neg\mathbf{T}$.

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 $A \wedge fn^{\subseteq}(N) \rhd B$, where $fn^{\subseteq}(N)$ is a short-cut for $\forall x(x \cdot_R \mathbf{T} \vee \bigvee_{n \in N} x = n)$. 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. 2. ■

Requiring that the names of the formula A are not in N is not restrictive. Indeed, we can simply rename each name of A in N with a fresh one and then perform the analysis w.r.t. this new formula (e.g., see Lemma 2.3 of Cardelli and Gordon, 2001).

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. By definition, $\mathbf{0} \models \mathbf{T} \triangleright A$ iff $\forall P \models \mathbf{T}$ we have $\mathbf{0} \,|\, P \models A$. So, we have $\mathbf{0} \,|\, P \models A$ iff $P \models A$.* ■

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

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

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 defined as:

$$ml(\mathbf{0}) = 0; \quad ml(\nu nP) = ml(P); \quad ml(a\langle n\rangle) = 1; \quad ml(a(n).P) = 1 + ml(P);$$
$$ml(P \mid P') = ml(P) + ml(P')$$

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

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

It is possible to build a process X' s.t. if $\nu N(P \mid X) \longrightarrow^* Q \mid pub\langle v\rangle$ then also $\nu N(P \mid X') \longrightarrow^* Q' \mid pub\langle v\rangle$ but X' does not perform any internal reduction.

Lemma 7 *If $\nu N(P \mid X) \longrightarrow^* Q \mid pub\langle v\rangle$ then we can find a process X' s.t. $\nu N(P \mid X') \longrightarrow^* Q' \mid pub\langle v\rangle$, 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 8 *Assume that $\nu N'(P \mid X) \longrightarrow^* Q \mid pub\langle v\rangle$, 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.:*

$$\nu N'(P \mid X') \longrightarrow^* Q \mid pub\langle v\rangle \qquad ■$$

Moreover, we can restrict ourselves to consider the analysis of the formula $leaked_v^{ml(P)}$ only for contexts $\nu fn(P)(P \mid X)$, 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.:

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

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. A satisfiability procedure for such formulas exists.

Lemma 9 *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 (Marchignoli and Martinelli, 1999; Martinelli, c), 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 $\triangleright$ 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 (Charatonik et al., 2001), where the model checking problem for the ambient logic have been studied, no results have been given about fragments with the $\triangleright$ operator. By means of partial model checking and by Lemma 9, 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 $fn^{\subseteq}(N) \triangleright B$ 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 (moreover the hidings are removed)*[3]. ■

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. A recent work (Caires and Cardelli, 2001) 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

[3] 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 9.

model checking problem (actually for a different logic) may be solved (see Dam, 1996). 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 (Abadi and Gordon, 1999) and *ambient* calculus (Cardelli and Gordon, 1998).

Acknowledgments. We wish to thank Andy Gordon for encouraging comments about this work. Thanks are also due to the anonymous referees for their help for the presentation of the paper.

References

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

Andersen, H. R. (1995). Partial model checking (extended abstract). In *Proc. of LICS*, pages 398–407. IEEE Computer Society Press.

Caires, L. and Cardelli, L. (2001). A spatial logic for concurrency (Part I). In *Proc. of TACS*, volume 2215 of *LNCS*.

Cardelli, L. and Gordon, A. (1998). Mobile ambients. In *Proc. of FOSSACS*, volume 1378 of *LNCS*, pages 140–155.

Cardelli, L. and Gordon, A. D. (2000). Anytime, anywhere: Modal logics for mobile ambients. In *Proc. of POPL*, pages 365–377. ACM Press.

Cardelli, L. and Gordon, A. D. (2001). Logical properties of name restriction. In *Proc. of TCLA 2001*, volume 2044 of *LNCS*, pages 46–60. Springer.

Charatonik, W., Dal Zilio, S., Gordon, A. D., Mukhopadhyay, S., , and Talbot, J.-M. (2001). The complexity of model checking mobile ambients. In *Proc. of FoSSaCS*, volume 2030 of *LNCS*, pages 52–167. Springer.

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

Engelfriet, J. and Gelsema, T. (1999). Multisets and structural congruence of the π-calculus with replication. *Theoretical Computer Science*, 211(1–2):311–337.

Focardi, R., Gorrieri, R., and Martinelli, F. (2000). Non interference for the analysis of cryptographic protocols. In *Proc. of ICALP*, volume 1853 of *LNCS*, pages 354–372.

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

Marchignoli, D. and Martinelli, F. (1999). Automatic verification of cryptographic protocols through compositional analysis techniques. In *Proc. of TACAS*, volume 1579 of *LNCS*.

Martinelli, F. About compositional analysis of π–calculus processes. Technical Report IAT-B4-019, December 2001. Full version of this paper.

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

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

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