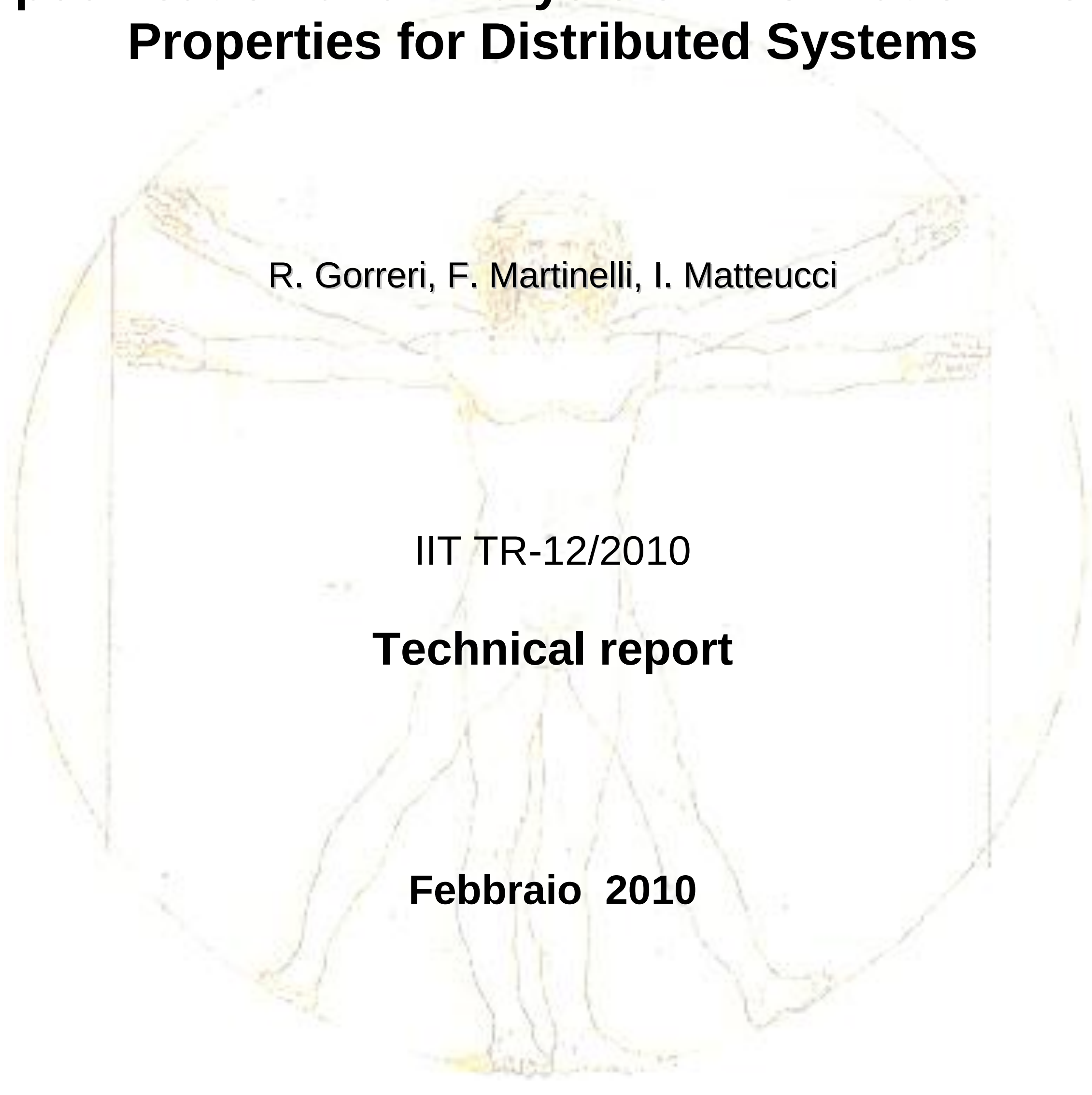
Consiglio Nazionale delle Ricerche



# Specification and Analysis of Information Flow Properties for Distributed Systems

R. Gorreri, F. Martinelli, I. Matteucci

IIT TR-12/2010

**Technical report**

**Febbraio 2010**

**Istituto di Informatica e Telematica**

# Specification and Analysis of Information Flow Properties for Distributed Systems*

Roberto Gorrieri
Università di Bologna, Bologna, Italy
gorrieri@cs.unibo.it

Fabio Martinelli
IIT CNR, Pisa, via Moruzzi, 1 - 56125 Pisa, Italy
fabio.martinelli@iit.cnr.it

Ilaria Matteucci
IIT CNR, Pisa, via Moruzzi, 1 - 56125 Pisa, Italy
ilaria.matteucci@iit.cnr.it

February 2, 2010

**Abstract**

We present a framework for the specification and the analysis of information flow properties in partially specified distributed systems, *i.e.*, systems in which there are several unspecified components located in different places. First we consider the notion of *Non Deducibility on Composition* (*NDC* for short) originally proposed for nondeterministic systems and based on trace semantics. We study how this information flow property can be extended in order to deal also with distributed partially specified systems. In particular, we develop two different approaches: the *centralized NDC* (*CNDC*) and the *decentralized NDC* (*DNDC*). According to the former, there is just one unspecified global component that has complete control of the $n$ distributed locations where interaction occurs between the system and the unspecified component. According to *DNDC*, there is one unspecified component for each distributed location, and the $n$ unspecified components are completely independent, *i.e.*, they cannot coordinate their efforts or cooperate. Surprisingly enough, we prove that *centralized NDC* is as discriminating as *decentralized NDC*. However, when we move to *Bisimulation-based Non-Deducibility on Composition*, *BNDC* for short, the situation is completely different. We prove that *centralized BNDC* (*CBNDC* for short) is strictly finer than *decentralized*

*Work partially supported by EU-funded project "Software Engineering for Service-Oriented Overlay Computers"(SENSORIA) and by EU-funded project "CONSEQUENCE". Preliminary results reported in the [12].

*BNDC* (*DBNDC for short*), hence proving the quite expected fact that a system that can resist to coordinated attacks is also able to resist to simpler attacks performed by independent entities. Hence, by exploiting a variant of the modal $\mu$-calculus that permits to manage tuples of actions, we present a method to analyze when a system is *CBNDC* and/or *DBNDC*, that is based on the theory of *decomposition* of formulas and compositional analysis.

**Keyword:** Information flow properties, distributed systems, compositional theory, contexts.

## 1 Introduction

Information flow analysis is considered one of the main techniques for studying confidentiality in computer systems. Information flow properties aim at defining the way the information may flow among different entities of a compound system. For instance, these properties may define constraints on the kind of information flow that can be set among different groups of entities with different security levels (*e.g.*, *high* and *low*). Usually, the goal is to prevent any possible flow from the confidential (*high*) level to the public (*low*) one.

Several formalizations have been proposed in the literature to capture the intuitive idea of flow of information. Most of them originate from the basic idea of *Non Interference*, *NI* for short, proposed in [11] on deterministic machines with outputs: Basically one wants that *low* output variables do not depend on *high* inputs. This intuitive notion has been then extended to trace based models. Assume there are two groups of users, $G$ and $G'$, and, given any input sequence of actions $\gamma$, let $\gamma'$ be its subsequence obtained by deleting all the actions of users in $G$. $G$ is *non interfering with* $G'$ if and only if for every input sequence $\gamma$, users in $G'$ obtain the same outputs after the execution of $\gamma$ and $\gamma'$.

This basic notion has been also adapted and generalized to the richer setting of nondeterministic labelled transition systems. Among the many definitions derived from *NI* (see *e.g.*, [5, 7, 19, 26]), here we consider the *Non Deducibility on Composition* property (*NDC*, [5]). Intuitively, a system is *NDC* if, by *interacting* with every possible high level user, it always *appears* the same to low level users, so that no information at all can be deduced by low level users. The above idea can be instantiated in a lot of ways, by choosing a particular way of interacting between systems and various criteria of equivalence. First we consider trace equivalence: A system $E$ is *NDC* if $E \setminus H$ (*i.e.*, $E$ where all high level actions are prevented) is trace equivalent to $E$ in parallel with any high level process $\Pi$ where all the high actions in $H$ are restricted (hence cooperation on high actions is forced). In other words, the low view of the behavior of system $E$ is not modified by the presence of process $\Pi$, that can be considered as an intruder that tries to break the system. Observe that this definition is given by considering at most one possible intruder (*high* user).

We can obtain a bisimulation based *NDC* by simply replacing trace equivalence with bisimulation equivalence. We consider the notion of *Bisimulation*

*Non Deducibility on Compositions*, *BNDC* for short, that was proposed in [5, 7]. Also in this case the definition is given by considering at most one possible *high* user.

The goal of this work is to extend the idea of *NDC* and *BNDC* also to distributed systems, where the possible intruders can be more than one and may also coordinate their efforts. In particular, the framework we present in this paper is composed of a first part in which we describe our approach for the specification of *NDC* and *BNDC* for distributed systems, and of a second part, in which we present a technique for analysis of *BNDC* based properties.

A distributed system is modelled as a *transducer* [17], *i.e.*, a context which can receive in input, say, $n$ actions in different locations and which may produce a tuple of outputs. Intuitively, a context of the form $C(X_1, \ldots, X_n)$ can be seen as a *distributed partially specified system* [20, 21, 22, 25], *i.e.*, a system $C$ with *holes*, where the components $X_1 \ldots X_n$ are not specified. These unspecified components are meant to be the potential intruders of the system.

We want to study whether such contexts respect information flow properties, in particular those based on the *NDC* and *BNDC* idea, whatever the possible intruders are. In both cases, we proceed by following two different approaches: A *centralized* approach and a *decentralized* one.

We first consider the *NDC* property. Given the context $C(X_1, \ldots, X_n)$, where each $X_i$ denotes a hole in the system, we may define *decentralized NDC* (*DNDC* for short) as the *NDC* property where the $n$ intruders act independently, without communicating or coordinating their activities. A system satisfying decentralided *NDC* should resist to distributed attacks conducted by $n$ independent intruders. Then, we introduce the concept of *centralized NDC* (*CNDC* for short) in which the $n$ intruders are centrally controlled and thus considered as a unique context which performs a vector of $n$ actions, $\tilde{a} = (a_1, \ldots, a_n)$. A system satisfying *CNDC* should resist to distributed attacks conducted by $n$ cooperating intruders (or by one single intruder that has complete control of the $n$ locations in which interaction with the system is possible).

Interestingly enough we prove the quite surprising result that *CNDC* is as discriminating as *DNDC*. Then, we consider the *BNDC* property and we discover that, when trace semantics is replaced by the more discriminating bisimulation semantics, the results above are completely different.

As expected, centralized *BNDC* (*CBNDC*, for short) is proved to be finer than decentralized *BNDC* (*DBNDC*, for short). Still, the weaker notion is meaningful because, as a matter of fact, a system that is centralized *BNDC* is able to resist also to strongly coordinated attacks that, in a real-life distributed environment, might not be possible. We provide a simple counterexample showing that the reverse implication does not hold, *i.e.*, a context which is decentralized *BNDC*, but not centralized *BNDC*.

Once we have specified our problem, we extend the analysis framework for *BNDC* described in [20, 21, 22, 25] to *CBNDC* and *DBNDC* properties. Indeed, by using a variant of the modal $\mu$-calculus, we characterize these properties by logic formulas. Then, by using *partial model checking* [1], that is a compositional analysis techniques, we are able to analyze these properties.

More in detail, we use the *characteristic formula* $\tilde{\phi}$, *i.e.*, a logical formula that characterizes the behavior of the system without high users, to describe the expected correct behavior of the system. Then we check that the system, even when composed with its inner high processes, always enjoys this formula. As a matter of fact, to analyze the $CBNDC$ property we have to solve the following problem:

$$\forall X \in C^n_{0,H} \quad \backslash_{H^m}(C(X)) \models \tilde{\phi} \tag{1}$$

where $\backslash_{H^m}$ denotes that all $n$-tuple of high level actions are prevented.

Instead, to analyze the *DBNDC* property, being $\tilde{\varphi}$ the characteristic formula of the system without high users, we have to solve the following problem:

$$\forall \{X_1, \ldots, X_n\} \subseteq C^1_{0,H} \quad \backslash_{H^m}(C(X_1, \ldots, X_n)) \models \tilde{\varphi} \tag{2}$$

In both cases we have an universal quantification: In (1) on all possible $n$-ary contexts and in (2) on all possible products of unary contexts. Using the *property transformer* function $\mathcal{W}$ (see [17]), *i.e.*, a function that permits to evaluate a given temporal logic with respect to a known context, we isolate the property the unspecified part of the system has to satisfy in order to be sure that the whole distributed system is $CBNDC$ and/or $DBNDC$. Indeed the problem in (1) is reduced as follows:

$$\forall X \in C^n_{0,H} \quad X \models \mathcal{W}(\backslash_{H^m} \circ C, \tilde{\phi}) \tag{3}$$

where $\circ$ is the *composition* operation between context. In a similar way, the problem in (2) is reduced by using the property transformer as follows:

$$\forall \{X_1, \ldots, X_n\} \subseteq C^1_{0,H} \quad X_1 \times \ldots \times X_n \models \mathcal{W}(\backslash_{H^m} \circ C, \tilde{\varphi}) \tag{4}$$

where $\times$ is the *product* operation between contexts.

We provide two different analysis methods for solving the problems (3) and (4). In the centralized case, a temporal logic formula has to be valid for all possible $n$-ary contexts, since we have a unique global component that controls all distributed locations. Such global component can be seen as a process that performs $n$-tuple of actions. Hence, to solve this validity problem, we exploit a validity procedure for modal logic formulas (*e.g.*, [3, 27]) as it has been already done in [20, 21, 22, 25] for studying *BNDC* property. As a matter of fact, solving a validity problem means proving that a given formula is always true, *i.e.*, every possible system satisfies it.

On the other hand, in the decentralized case, we appeal to the procedure of *decomposition of formulas* in [17]: Finding a decomposition of a formula $\tilde{\varphi}$ means finding a *product formula*, *i.e.*, a product of unary formulas $\phi_1 \times \ldots \times \phi_n$, which is equivalent to $\tilde{\varphi}$. In particular, we are interested in finding a weak decomposition, called *safe* decomposition, *i.e.*, a *product formula* which logically implies $\tilde{\varphi}$. Once we obtain this decomposition, to verify $DBNDC$ we verify if each unknown component of the system satisfies one of the formulas of the product, *i.e.*, $\forall i = 1, \ldots, n \; \forall X \; X \models \phi_i$. In this way we have $n$ validity problems that can be solved, as before, by exploiting a validity procedure for modal logic formulas. If all the formulas are satisfied then also $\tilde{\varphi}$ is so.

*This paper is organized as follows.* Section 2 recalls some notions about contexts, logic and compositional theory. Section 3 presents our framework for the specification of information flow properties for distributed systems and Section 4 shows our framework for the analysis of such properties. In Section 5 a comparison with related work is reported. In Section 6 we draw some conclusions.

## 2 Background

In this section we recall some preliminary notions about contexts theory and modal logic referring to [17].

### 2.1 Context

First of all, we recall the definition of context.

**Definition 2.1** *A* context system $\mathcal{C}$ *is a structure* $\mathcal{C} = (\langle C_n^m \rangle_{n,m}, Act, \langle \rightarrow_{n,m} \rangle_{n,m})$ *where* $C_n^m$ *is a set of* $n$*-to-*$m$ contexts*;* $Act$ *is a set of actions;* $Act_0 = Act \cup \{0\}$ *where* $0 \notin Act$ *is a distinguished no-action symbol,* $Act_0^k$ *is a tuple of* $k$ *actions in* $Act_0$*, and* $\rightarrow_{n,m} \subseteq C_n^m \times (Act_0^n \times Act_0^m) \times C_n^m$ *is the* transduction-relation *for the* $n$*-to-*$m$ contexts *satisfying* $(C, \tilde{a}, \tilde{0}, D) \in \rightarrow_{n,m}$ *if and only if* $C = D$ *and* $\tilde{a} = \tilde{0}$ *for all contexts* $C, D \in C_n^m$*.*

For $(C, \tilde{a}, \tilde{b}, C') \in \rightarrow_{n,m}$ we usually write $C \overset{\tilde{b}}{\underset{\tilde{a}}{\rightarrow}} C'$, leaving the indices of $\rightarrow$ to be determined by the context, and we interpret this as:

*Consuming the actions $\tilde{a}$, the context $C$ can produce the actions $\tilde{b}$ becoming into $C'$.*

In a transduction $C \overset{\tilde{b}}{\underset{\tilde{a}}{\rightarrow}} C'$, certain components in $\tilde{a}$ and/or $\tilde{b}$ can be 0 indicating that the corresponding internal process and/or external observer is not involved in the transduction. In particular the last condition of $\rightarrow_{n,m}$ means that a context can always and only produce nothing without consuming anything.

In order to give some example of contexts, we present here how it is possible to see process algebra operators as contexts.

**Example 2.1** *Finite CCS process algebra [24] can be seen as a context system with the following contexts:* prefix $a^* \in C_1^1$ *for* $a \in Act$*,* restriction $\backslash L \in C_1^1$ *where* $L \subseteq Act$*.* Choice *and* parallel *context* $+, \| \in C_2^1$*;* inaction $\tilde{Nil} \in C_n^m$ *for any* $n$ *and* $m$*, with* $Nil$ *abbreviating the inaction process in* $C_0^1$*. There are also the* identity *context* $I_n \in C_n^n$ *and the* projection $\Pi_n^i \in C_n^1$*. The semantics definition of finite CCS contexts is reported in Table 1.*

Inaction:

$$C \overset{\tilde{0}}{\underset{\tilde{0}}{\longrightarrow}} C \text{ for all } C$$

Prefix:

$$a^* \overset{a}{\underset{0}{\longrightarrow}} I_1$$

Restriction:

$$\backslash_L \overset{a}{\underset{a}{\longrightarrow}} \backslash_L \quad a \notin L$$

Choice:

$$(1) + \xrightarrow[(a,0)]{a} \Pi_2^1 \quad (2) + \xrightarrow[(0,a)]{a} \Pi_2^2 \text{ for } a \in Act$$

Projection:

$$\Pi_n^i \xrightarrow[i(a)]{a} \Pi_n^i$$

Identity:

$$I_n \overset{\tilde{a}}{\underset{\tilde{a}}{\longrightarrow}} I_n$$

Parallel:

$$(1) \parallel \xrightarrow[(a,\bar{a})]{\tau} \parallel \quad (2) \parallel \xrightarrow[(a,0)]{a} \parallel \quad (3) \parallel \xrightarrow[(0,a)]{a} \parallel$$

where $i(a) \in Act_0^n$ with the $i^{th}$ component being $a$ and all the others being 0.

Table 1: Semantics of $CCS$ context system.

### 2.1.1 Operations between contexts

Several operations are allowed between contexts.

**Composition of contexts.**

**Definition 2.2** *Let $\mathcal{C} = (\langle C_n^m \rangle_{n,m}, Act, \langle \rightarrow_{n,m} \rangle_{n,m})$ be a context system. A* composition *on $\mathcal{C}$ is a dyadic operation $\circ$ on contexts such that whenever $C \in C_n^m$ and $D \in C_m^r$ then $D \circ C \in C_n^r$. Furthermore, the transductions for a context $D \circ C$ with $C \in C_n^m$ and $D \in C_m^r$ are fully characterized by the following rule:*

$$\frac{C \overset{\tilde{b}}{\underset{\tilde{a}}{\longrightarrow}} C' \quad D \overset{\tilde{c}}{\underset{\tilde{b}}{\longrightarrow}} D'}{D \circ C \overset{\tilde{c}}{\underset{\tilde{a}}{\longrightarrow}} D' \circ C'}$$

*where $\tilde{a} = (a_1, \ldots, a_n)$, $\tilde{b} = (b_1, \ldots, b_m)$ and $\tilde{c} = (c_1, \ldots, c_r)$ are vectors of actions.*

We often write $C(P)$ to denote a composed context $C \circ P$.

**Example 2.2** *In order to explain how the composition between contexts works, we recall an example given in [17].*

*Let $a.b.Nil$ be a standard $CCS$ term. It can be built from the constructs as $a^* \circ b^* \circ Nil$. Using the inference rule for composition, we can infer the following transitions:*

$$\frac{a^* \overset{a}{\underset{0}{\rightarrow}} I_1 \quad b^* \circ Nil \overset{0}{\underset{0}{\rightarrow}} b^* \circ Nil}{a^* \circ b^* \circ Nil \overset{a}{\underset{0}{\rightarrow}} I_1 \circ b^* \circ Nil}$$

*and*

$$\frac{\dfrac{-}{I_1 \overset{b}{\underset{b}{\rightarrow}} I_1} \qquad \dfrac{b^* \overset{b}{\underset{0}{\rightarrow}} I_1 \quad Nil \xrightarrow{0} Nil}{b^* \circ Nil \xrightarrow{b} I_1 \circ Nil}}{I_1 \circ b^* \circ Nil \xrightarrow{b} I_1 \circ I_1 \circ Nil}$$

*Obviously, a composed context of the form $I_m \circ C$ has the same behavior as $C$ itself.* ■

**Product of contexts.** In order to represent a system with $n$ holes, we use a $n$-to-1 context $C \in C_n^1$. If $C$ is combined with a context $D \in C_m^n$, we obtain $C \circ D \in C_m^1$. If $m = 0$ then we obtain a process. The context $D$, in this case, provides a simultaneous expansion of the $n$ holes in $C$. To allow the expansion of the $n$ holes to be carried out independently, it is defined an *independent combination* of $n$ contexts as $D_1 \times \ldots \times D_n$, where $D_i \in C_{m_i}^1$, $i = 1, \ldots, n$ and $D_i$ is intended as an expansion of the $i$'th hole of $C$ in such a way that $m = \sum_{i=1}^n m_i$. This motivates the following construct of (independent) products of contexts.

**Definition 2.3** *Let $\mathcal{C} = (\langle C_n^m \rangle_{n,m}, Act, \langle \rightarrow_{n,m} \rangle_{n,m})$ be a context system. A* product *on $\mathcal{C}$ is a dyadic operation $\times$ on contexts, such that whenever $C \in C_n^m$ and $D \in C_r^s$ then $C \times D \in C_{n+r}^{m+s}$. Furthermore the transduction for a context $C \times D$ are fully characterized by the following rule:*

$$\frac{C \overset{\tilde{b}}{\underset{\tilde{a}}{\rightarrow}} C' \quad D \overset{\tilde{d}}{\underset{\tilde{c}}{\rightarrow}} D'}{C \times D \overset{\tilde{b}\tilde{d}}{\underset{\tilde{a}\tilde{c}}{\rightarrow}} C' \times D'}$$

*where juxtaposition of vectors $\tilde{a} = (a_1, \ldots, a_n)$ and $\tilde{c} = (c_1, \ldots, c_r)$ is the vector $\tilde{a}\tilde{c} = (a_1, \ldots, a_n, c_1, \ldots, c_r)$ and juxtaposition of vectors $\tilde{b} = (b_1, \ldots, b_m)$ and $\tilde{d} = (d_1, \ldots, d_s)$ is the vector $\tilde{b}\tilde{d} = (b_1, \ldots, b_m, d_1, \ldots, d_s)$.*

We usually write the combined process $C(P_1, \ldots, P_n)$ as a shorthand for $C \circ (P_1 \times \ldots \times P_n)$. Since we consider *asynchronous* contexts, it is not required that all the components $P_1, \ldots, P_n$ contribute in a transition of the combined process $C(P_1, \ldots, P_n)$, *i.e.*, some of the $P_i$ could perform a 0 action.

**Example 2.3** *Also in this case, since in the rest of the paper composition and product operations are the most useful for our purposes, we recall an example already presented in [17].*

*Let $a.Nil + b.Nil$ be a standard CCS term. It can be composed from the constructs as $+ \circ (a^* \circ Nil \times b^* \circ Nil)$. Using the inference rules we have the following transitions:*

$$\frac{\dfrac{a^* \overset{a}{\underset{0}{\rightarrow}} I_1 \quad Nil \xrightarrow{0} Nil}{a^* \circ Nil \xrightarrow{a} I_1 \circ Nil} \qquad \dfrac{b^* \overset{b}{\underset{0}{\rightarrow}} I_1 \quad Nil \xrightarrow{0} Nil}{b^* \circ Nil \xrightarrow{b} I_1 \circ Nil}}{a^* \circ Nil \times b^* \circ Nil \xrightarrow{(a,b)} I_1 \circ Nil \times I_1 \circ Nil}$$

*Composing $(a^* \circ Nil \times b^* \circ Nil)$ with context $+$ we have:*

$$\frac{+ \overset{a}{\overrightarrow{(a,b)}}\, \Pi_2^1 \quad a^* \circ Nil \times b^* \circ Nil \xrightarrow{(a,b)} I_1 \circ Nil \times I_1 \circ Nil}{+ \circ (a^* \circ Nil \times b^* \circ Nil) \xrightarrow{a} \Pi_2^1 \circ (I_1 \circ Nil \times I_1 \circ Nil)}$$

*where the behavior of $\Pi_2^1 \circ (I_1 \circ Nil \times b^* \circ Nil)$ is behavioral equivalent to $Nil$. Hence, in accordance with the standard CCS transition relation, $a.Nil + b.Nil \xrightarrow{a} Nil$.* ■

**Feed-back on contexts.** In order to deal with *iteration*, a construction of *feed-back* on contexts is defined, such that whenever $C \in C_n^n$ then $C^\dagger \in C_0^n$ and $C^\dagger$ is equivalent to $C \circ C^\dagger$. Formally, we have the following definition.

**Definition 2.4** *Let $\mathcal{C} = (\langle C_n^m \rangle_{n,m}, Act, \langle \rightarrow_{n,m} \rangle_{n,m})$ be a context system. A* feed-back *on $\mathcal{C}$ is a unary operation $^\dagger$ on contexts of $\mathcal{C}$ such that, whenever $C \in C^{n,n}$ then $C^\dagger \in C_0^n$. Furthermore, the transduction for a context $C^\dagger$ with $C \in C_n^n$ is fully characterized by the rule:*

$$\frac{C \overset{\bar{b}}{\underset{\bar{a}}{\rightarrow}} C' \quad C^\dagger \xrightarrow{\bar{a}} D}{C^\dagger \xrightarrow{\bar{b}} C' \circ D}$$

In order to understand the meaning of this operator we recall the following example in [17].

**Example 2.4** *Let us consider the CCS process defined as $X = a.X$. It can be realized as the context $(a^*)^\dagger$. Indeed, using the inference rule of the feed-back, we obtain the following transition:*

$$\frac{a^* \overset{a}{\underset{0}{\rightarrow}} I_1 \quad (a^*)^\dagger \xrightarrow{0} (a^*)^\dagger}{(a^*)^\dagger \xrightarrow{a} I_1 \circ (a^*)^\dagger}$$

*and in fact this is the only transition for $(a^*)^\dagger$.*

■

### 2.1.2 Open systems as transducers

According to [17], the theory of contexts is useful to model and analyze *distributed partially specified systems*, *i.e.*, systems in which more than one component is unspecified. As a matter of fact, a partially specified system can be formalized by contexts as a system of the form $C(X_1, \ldots, X_n)$, where $C$ denotes the known part of the system and $X_1, \ldots, X_n$ denote components still remaining unknown. In process algebra, the partial implementation $C(X_1, \ldots, X_n)$ can be described as an expression with $X_1, \ldots, X_n$ as free variables that may be replaced by closed expression $P_1, \ldots, P_n$. By syntactically replacing each $X_i$ by the corresponding $P_i$, we get the closed expression $C(P_1, \ldots, P_n)$.

In this way we have a general framework that permits us to consider a complex and general scenario. Indeed, when we consider a partially specified system, several scenarios can be considered in order to obtain a closed system. In the easiest case, the number of the unspecified components is one. In this case there is a unique hole that a process can fill in. On the other hand, if we consider a system in which there are several holes, we may distinguish from two different approaches for the analysis of such systems: By considering all $n$ holes as a unique hole of cardinality $n$ (in this case we put a central process that performs a vector of $n$ actions), or by considering several independent unary processes whose product closes the expression.

**Example 2.5** *Let $C_1 = h_1^* \circ l_1^* \circ h_1^* \circ Nil$ and $C_2 = h_2^* \circ l_2^* \circ Nil$ be two contexts. Let us consider a context $C \in C_2^1$ built as a composition of several contexts as follows:*

$$C = \| \circ (\backslash_{\{h_1, h_2, \bar{h_1}, \bar{h_2}\}} \circ \|(C_1 \times X_1)) \times (\backslash_{\{h_1, h_2, \bar{h_1}, \bar{h_2}\}} \circ \|(C_2 \times X_2))$$

*Here we have explicitly denoted by $X_1$ and $X_2$, in $C_0^1$, the free variables in $C_0^1$ of the contexts in order to make the context readable. Being $\| \in C_2^1$, $C_1$ and $C_2$ in $C_0^1$ it is not difficult to see that $C$ is effectively in $C_2^1$. $X_1$ and $X_2$ represent respectively the first and the second unspecified component of the context $C \in C_2^1$.*

*Informally, this context forces the synchronization on actions $h_1, h_2$. This permits us, as we will see in the following, to control sequences of executions of low actions, such as $l_1$ and $l_2$.*

*In order to obtain a closed expression we have to combine $C$ with a context in $C_0^2$. Such a context could be a binary context such as, for instance, $P = (\bar{h_1}, 0)^* \circ (\bar{h_1}, 0)^* \circ (0, \bar{h_2})^* \circ \tilde{Nil}$. Or the product of a couple of contexts in $C_0^1$, for instance $P_1 = \bar{h_1}^* \circ \bar{h_1}^* \circ Nil$ and $P_2 = \bar{h_2}^* \circ Nil$. As a matter of fact, the product $P_1 \times P_2$ is a context in $C_0^2$ as required. In this case the closed expression is $C \circ (P_1 \times P_2)$.*

*Referring to Definition 2.2, the context $C \circ P$ behaves according to the following rule:*

$$\frac{P \xrightarrow{(\bar{h_1},0)} P' \quad C \overrightarrow{(\bar{h_1},0)}^{\tau} C'}{C(P) \xrightarrow{\tau} C'(P')} \tag{5}$$

*where* $P' = (\bar{h_1}, 0)^* \circ (0, \bar{h_2})^* \circ \tilde{Nil}$ *and* $C' = \| \circ (\backslash_{\{h_1,h_2\bar{h_1},\bar{h_2}\}} \circ \|(C_1' \times X_1) \times \backslash_{\{h_1,h_2,\bar{h_1},\bar{h_2}\}} \circ \|(C_2 \times X_2))$ *where* $C_1' = l_1^* \circ h_1^* \circ Nil$. *Obviously this is the first transition. The rest of the transduction is omitted, but it is possible to calculate the following steps by applying the composition rule. At the end we obtain that* $C(P)$ *has the same behavior as* $\tau^* \circ l_1^* \circ \tau^* \circ \tau^* \circ l_2^* \circ Nil$. *This is the only maximal sequence of actions allowed for this composition of contexts. As a matter of fact, informally, if* $C_2$ *pretends to perform the first action, it is forbidden by the restriction* $\backslash_{\{h_1,h_2,\bar{h_1},\bar{h_2}\}}$.

*Let us now consider the distributed case. Referring to how the contexts are built, the first transition may be one of the following two:*

$$\frac{\dfrac{P_1 \xrightarrow{\bar{h_1}} P_1' \qquad P_2 \xrightarrow{0} P_2}{P_1 \times P_2 \xrightarrow{(\bar{h_1},0)} P_1' \times P_2} \qquad \dfrac{-}{C \xrightarrow[(\bar{h_1},0)]{\tau} C'}}{C(P_1, P_2) \xrightarrow{\tau} C'(P_1', P_2)}$$

$$\frac{\dfrac{P_2 \xrightarrow{\bar{h_2}} Nil' \qquad P_1 \xrightarrow{0} P_1}{P_1 \times P_2 \xrightarrow{(\bar{h_2},0)} P_1 \times Nil} \qquad \dfrac{-}{C \xrightarrow[(\bar{h_2},0)]{\tau} C'}}{C(P_1, P_2) \xrightarrow{\tau} C'(P_1, Nil)}$$

*It is possible to note that, in the distributed case, there exists the possibility that the first step is performed by* $P_2$ *and* $C_2$, *then we can also obtain a sequence starting with* $\tau^* \circ l_2 \circ \ldots$. *This cannot happen in the centralized case.* $\blacksquare$

### 2.1.3 Behavioral equivalences

In order to have a method to compare behaviors of contexts, we recall some definitions of behavioral equivalences. We start with the definition of *trace equivalence* for contexts.

Let us start by giving some notations used in the following.

Let us consider $\tilde{\tau}$ as a tuple of actions in which there are no actions different from $\tau$ or 0. Let $\tilde{a} = (a_1, \ldots, a_n)$ be a vector of actions in $Act_0^n$. Then $\check{\tilde{a}} = \tilde{a}$ when $a_i \neq \tau$ for all $i = 1, \ldots, n$, or $\check{\tilde{a}} = \tilde{a}[0\backslash\tau]$ where all the occurrence of the $\tau$ actions in the vector are replaced by the no action 0. In particular $\check{\tilde{\tau}} = \tilde{\tau}$, *i.e.*, if the vector is composed only by $\tau$ or 0 actions, the substitution is not performed.

Then, notation $C \overset{\tilde{\tau}}{\Longrightarrow} C'$ denotes that $C$ and $C'$ belongs to the reflexive and transitive closure of $\xrightarrow{\tilde{\tau}}$. Also, $C \overset{\check{\tilde{a}}}{\Longrightarrow} C''$ iff $C \overset{\tilde{\tau}}{\Longrightarrow} \xrightarrow{\check{\tilde{a}}} \overset{\tilde{\tau}}{\Longrightarrow} C''$.

Let $\gamma = \tilde{a_1} \ldots \tilde{a_n} \in Act_0^n \backslash \{\tilde{\tau}\}$ be a *sequence* of vectors of actions, *i.e.*, a *trace*, where $Act_0^n \backslash \{\tilde{\tau}\}$ is the set of vectors of actions of length $n$ without the $n$-tuple $\tilde{\tau}$.

Let $\check{\gamma} = \check{\tilde{a_1}} \ldots \check{\tilde{a_n}}$ be a sequence of vectors of actions in which each occurrences of $\tau$ actions in each vector has been replaced by 0. $C \overset{\check{\gamma}}{\Longrightarrow} C'$ iff $C \overset{\check{\tilde{a_1}}}{\Longrightarrow} \ldots \overset{\check{\tilde{a_n}}}{\Longrightarrow} C'$

Let $C \in C_0^n$ be a context, then the set of traces of $C$ is $Tr(C) = \{\check{\gamma} | \exists C' \; C \overset{\check{\gamma}}{\Longrightarrow} C'\}$.

**Definition 2.5** *Let $C, D \in C_0^n$ be two contexts. We define the relation of* trace inclusion*, denoted by $\leq_T$ as follows:*

$$C \leq_T D \textit{ iff } Tr(C) \subseteq Tr(D)$$

*Moreover, $C$ and $D$ are* trace equivalent*, denoted by $\approx_T$, iff $Tr(C) = Tr(D)$.*

Other behavioral equivalences are defined for contexts. We just recall the definitions of *simulation* and *bisimulation* equivalences [24] by distinguishing between a *strong* version and a *weak* one.

**Definition 2.6** *Let $\mathcal{C} = (\langle C_n^m \rangle_{n,m}, Act, \langle \rightarrow_{n,m} \rangle_{n,m})$ be a context system. Then an $n$-to-$m$* strong simulation *$\mathcal{R}$ is a binary relation on $C_n^m$ such that, whenever $(C, D) \in \mathcal{R}$ and $\tilde{a} \in Act_0^n$, $\tilde{b} \in Act_0^m$, then the following holds:*

$$\textit{if } C \overset{\tilde{b}}{\underset{\tilde{a}}{\rightarrow}} C', \textit{ then } D \overset{\tilde{b}}{\underset{\tilde{a}}{\rightarrow}} D' \textit{ for some } D' \textit{ with } (C', D') \in \mathcal{R}$$

*We write $C \prec D$ in case $(C, D) \in \mathcal{R}$ for some $n$-to-$m$ simulation $\mathcal{R}$.*

*A* strong bisimulation *is a relation $\mathcal{R}$ such that both $\mathcal{R}$ and $\mathcal{R}^{-1}$ are strong simulations. We represent with $\sim$ the union of all the strong bisimulations.*

Now we are able to give the definition of *weak bisimulation* by considering that, in the rest of the paper, we use it with respect to contexts in $C_0^n$. Hence we consider a definition that is an extension of the definition of weak bisimulation given by Milner in [24] for processes, that, as we have already said, can be seen as contexts in $C_0^1$.

**Definition 2.7** *Let $\mathcal{C} = (\langle C_0^n \rangle_{0,n}, Act, \langle \rightarrow_{0,n} \rangle_{0,n})$ be a context system. Then a $0$-to-$n$* weak simulation *$\mathcal{R}$ is a binary relation on $C_0^n$ such that, whenever $(C, D) \in \mathcal{R}$ and $\tilde{a} \in Act_0^n$, then the following holds:*

$$\textit{if } C \xrightarrow{\tilde{a}} C', \textit{ then } D \overset{\hat{\tilde{a}}}{\Longrightarrow} D' \textit{ for some } D' \textit{ with } (C', D') \in \mathcal{R}.$$

*We write $C \preceq D$ in case $(C, D) \in \mathcal{R}$ for some $0$-to-$n$ simulation $\mathcal{R}$.*

*A* weak bisimulation *is a relation $\mathcal{R}$ such that both $\mathcal{R}$ and $\mathcal{R}^{-1}$ are weak simulations. We denote with $\approx$ the union of all the weak bisimulations.*

**Theorem 2.1 ([17])** *$\sim$ is preserved by composition, product and feed-back of contexts.*

## 2.2 Modal language and property transformer for contexts

In order to express properties of contexts, we recall a variant of the modal $\mu$-calculus, proposed in [17]. This variant allows minimum and maximum fixed points to be used freely and interchangeably.

$$\begin{array}{l}
[\![\mathbf{T}]\!]_\rho = \Gamma \quad [\![\mathbf{F}]\!]_\rho = \emptyset \quad [\![X]\!]_\rho = \rho(X) \\
[\![\phi_1 \vee \phi_2]\!]_\rho = [\![\phi_1]\!]_\rho \cup [\![\phi_2]\!]_\rho \; [\![\phi_1 \wedge \phi_2]\!]_\rho = [\![\phi_1]\!]_\rho \cap [\![\phi_2]\!]_\rho \\
[\![\langle a \rangle \phi]\!]_\rho = \{\gamma \in \Gamma | \exists \gamma' : \gamma \xrightarrow{a} \gamma' \wedge \gamma' \in [\![\phi]\!]_\rho\} \\
[\![[a]\phi]\!]_\rho = \{\gamma \in \Gamma | \forall \gamma' : \gamma \xrightarrow{a} \gamma' \Rightarrow \gamma' \in [\![\phi]\!]_\rho\} \\
[\![\textsc{let max}\, D \,\textsc{in}\, \phi]\!]_\rho = [\![\phi]\!](D_\nu [\![D]\!]_\rho) \\
[\![\textsc{let min}\, D \,\textsc{in}\, \phi]\!]_\rho = [\![\phi]\!](D_\mu [\![D]\!]_\rho)
\end{array}$$

and

$$\begin{array}{l}
D_\nu [\![X_1 = \phi_1 \ldots X_n = \phi_n]\!]_\rho = \nu\rho' \rho\{[\![\phi_1]\!]_{\rho'} \backslash X_1, \ldots, [\![\phi_n]\!]_{\rho'} \backslash X_n\} \\
D_\mu [\![X_1 = \phi_1 \ldots X_n = \phi_n]\!]_\rho = \mu\rho' \rho\{[\![\phi_1]\!]_{\rho'} \backslash X_1, \ldots, [\![\phi_n]\!]_{\rho'} \backslash X_n\}
\end{array}$$

Table 2: Semantics clauses.

**Definition 2.8** *Let $L$ be a set of labels (or actions), ranged over by $a$; let $V$ be a set of variables, ranged over by $X$. The sets of* formulas *$\mathcal{F}_{V,L}$ (ranged over by $\phi$) and* declarations *$\mathcal{D}_{V,L}$ (ranged over by $D$) over $V$ relative to $L$ are built up according to the following, mutually recursive, abstract syntax:*

$$\begin{array}{l}
\phi ::= \mathbf{T} \mid \mathbf{F} \mid X \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \mid \langle a \rangle \phi \mid [a]\phi \mid \\
\qquad \textsc{let max}\ D\ \textsc{in}\ \phi \mid \textsc{let min}\ D\ \textsc{in}\ \phi \\
D ::= X_1 = \phi_1 \ldots X_n = \phi_n
\end{array}$$

The above logic is a propositional modal logic with $\langle a \rangle \phi$ and $[a]\phi$ providing the two relativized modalities. The declaration in the LET-constructs, introduces simultaneous recursively specified properties. The concepts of free and bound variables are defined as usual; in particular we call a formula *closed* if it contains no free variables. We shall use standard notation $\phi[\psi/X]$ to describe the substitution of $\psi$ for all free occurrences of the variable $X$ in $\phi$.

The interpretation of the introduced logic is given with respect to a *Labelled Transition System*, *LTS* for short, that is a structure $(\Gamma, L, \rightarrow)$, where $\Gamma$ is the set of states, $L$ is the set of actions and $\rightarrow \subseteq \Gamma \times L \times \Gamma$ is the *transition relation.* The interpretation of a closed formula is given as the set of states *satisfying* the formula. The semantics of formulas is given with respect to an environment $\rho = V \rightarrow \mathcal{P}(\Gamma)$. The semantics definition is given inductively on the structure of formulas and declarations as in Table 2, with $\nu$ and $\mu$ being the *maximum* and *minimum* fixed point operators, respectively.

This logic is useful for expressing interesting properties. In particular minimum fixed point constructs allow for expressing *liveness* properties ("something good happens"), whereas maximum fixed point constructs allow for expressing *safety* properties ("nothing bad can happen").

**Example 2.6** *The following formula expresses that any configuration that is reached by an $a$-transition satisfies the property $\phi$:*

$$\textsc{letmax}\ X = \phi \wedge [a]X\ \textsc{in}\ X$$

*On the other hand, if we are interested to express that some configuration reachable by a-transition satisfies $\phi$ we use the following formula:*

$$\textsc{letmin}\ X = \phi \vee \langle a \rangle X\ \textsc{in}\ X$$

■

### 2.2.1 Characteristic formulas

The main theorem of Hennessy and Milner in [16] shows that the non-recursive part of the logic characterizes bisimilarity in the sense that two closed processes are bisimilar just in case they satisfy the same (non-recursive) formulas. The theorem is subject to the assumption that the underlying transition system is *image-finite*, *i.e.*, for any process $\gamma$ and action $a$ the set $\{\gamma' | \gamma \xrightarrow{a} \gamma'\}$ is finite. However, one half of the characterization theorem is valid for any transition system: whenever two configurations are bisimilar they will satisfy the same formulas. Even with the addition of recursion, no formula is able to distinguish between bisimilar configurations.

In this paper we are interested to the existence of characteristic formula $\phi$ for a $n$-ary process $P \in C_0^n$. Hence, here, we extend the definition of characteristic formulas with respect to weak bisimulation, as given in *e.g.*, [21], by considering $n$-ary processes. Indeed we consider vectors of actions instead of single actions.

Considering weak bisimulation, internal actions of a process can be matched by zero or more internal steps of the other process.

Let $\langle\langle \tilde{a} \rangle\rangle$ be a weak version of the modality $\langle \tilde{a} \rangle$, introduced as abbreviation and defined as follows, *e.g.*, [21]:

$$\langle\langle \tilde{\epsilon} \rangle\rangle \phi \overset{def}{=} \textsc{let min}\ Z = \phi \vee \langle \tilde{\tau} \rangle Z\ \textsc{in}\ Z \qquad \langle\langle \tilde{a} \rangle\rangle \phi \overset{def}{=} \langle\langle \tilde{\epsilon} \rangle\rangle \langle \tilde{a} \rangle \langle\langle \tilde{\epsilon} \rangle\rangle \phi$$

Moreover, let $[[\tilde{a}]]$ be a weak version of the modality $[\tilde{a}]$, introduced as abbreviation as follows:

$$[[\tilde{\epsilon}]] \phi \overset{def}{=} \textsc{let max}\ Z = \phi \wedge [\tilde{\tau}] Z\ \textsc{in}\ Z \qquad [[\tilde{a}]] \phi \overset{def}{=} [[\tilde{\epsilon}]] [\tilde{a}] [[\tilde{\epsilon}]] \phi$$

Now we are able to give the following definition.

**Definition 2.9** *Given a $n$-ary finite state process $P$, its characteristic formula with respect to weak bisimulation is given by the formula $\phi =$ LET MAX $D_P$ IN $Z_P$ where $D_P$ is the declaration system associated to the process $P$ defined with respect to the variable $Z_P$. It consists of a system of equations of the form $Z_{P'} = \phi'$, one for each $P' \in Der(P)$, where each $Z_{P'}$ is a variable and $\phi'$ is the characteristic formula of $P'$ defined recursively as follows:*

$$Z_{P'} = \Big( \bigwedge_{\tilde{a};P'':P' \overset{\tilde{a}}{\Rightarrow} P''} \langle\langle \check{\tilde{a}} \rangle\rangle Z_{P''} \Big) \wedge \Big( \bigwedge_{\tilde{a}} \big( [[\tilde{a}]] \big( \bigvee_{P'':P' \overset{\check{\tilde{a}}}{\Rightarrow} P''} Z_{P''} \big) \big) \Big)$$

$$\mathcal{W}(C,\mathbf{T}) = \mathbf{T} \quad \mathcal{W}(C,\mathbf{F}) = \mathbf{F} \quad \mathcal{W}(C,X) = X^C$$
$$\mathcal{W}(C,\phi_1 \wedge \phi_2) = \mathcal{W}(C,\phi_1) \wedge \mathcal{W}(C,\phi_2) \quad \mathcal{W}(C,\phi_1 \vee \phi_2) = \mathcal{W}(C,\phi_1) \vee \mathcal{W}(C,\phi_2)$$
$$\mathcal{W}(C,\langle\tilde{b}\rangle\phi) = \bigvee_{C \xrightarrow[\tilde{a}]{\tilde{b}} D} \langle\tilde{a}\rangle\mathcal{W}(D,\phi) \quad \mathcal{W}(C,[\tilde{b}]\phi) = \bigwedge_{C \xrightarrow[\tilde{a}]{\tilde{b}} D} [\tilde{a}]\mathcal{W}(D,\phi)$$
$$\mathcal{W}(C,\text{LET MAX } D \text{ IN } \phi) = \text{ LET MAX } D^T \text{ IN } \mathcal{W}(C,\phi)$$
$$\mathcal{W}(C,\text{LET MIN } D \text{ IN } \phi) = \text{ LET MIN } D^T \text{ IN } \mathcal{W}(C,\phi)$$

where
$$D^T = \{X^C = \mathcal{W}(C,\phi) | C \in C_n^m, X = \phi \in D\}$$

Table 3: Definition of property transformer.

### 2.2.2 Property transformer function

According to [17], it is possible to define the *property transformer function* $\mathcal{W}$ for contexts as in Table 3. The property transformer is introduced in order to isolate which is the necessary and sufficient conditions that the unspecified part of the system has to satisfy. The following theorem holds.

**Theorem 2.2 ([17])** *Let $\mathcal{C} = (\langle C_n^m\rangle_{n,m}, Act, \langle\rightarrow_{n,m}\rangle_{n,m})$ be a context system. Let $\phi$ be a closed formula and let $C \in C_n^m$. Then for any $Q \in C_0^n$ the following equivalence hold:*
$$C(Q) \models \phi \Leftrightarrow Q \models \mathcal{W}(C,\phi)$$

Hence, this theorem allows us to find, given the context $C$ and the property $\phi$, the weakest property $\mathcal{W}(C,\phi)$ that has to be satisfied by the combination of the unspecified components of the system.

## 2.3 Some information flow properties

*Information flow properties* are a particular class of security properties which aim at controlling the way information may flow among different entities. They have been first proposed as a means to ensure confidentiality, in particular to verify if access control policies are sufficient to guarantee the secrecy of (possibly classified) information. Indeed, even if access control is a well studied technique for system security, it may be impossible to find an access control policy which guarantees that no information leak.

In the literature, there are many different security definitions reminiscent of the information flow idea, each based on some system model (see [5, 7, 19, 26]). The central property for this paper is *Non Deducibility on Composition* (*NDC*, see [5]).

Exploiting the relation of bisimulation, we obtain the notion of *Bisimulation Non Deducibility on Compositions*, *BNDC* for short, proposed in [5, 7] as a generalization of the classical idea of *Non-Interference* [11] to nondeterministic systems.

### 2.3.1 *NDC* and *BNDC* properties

To describe information flow properties, we can consider two users, *High* and *Low* interacting with the same computer system. We wonder if there is any flow of information from *High* to *Low*.

In [4, 5, 6, 7] a family of information flow security properties called *Non Deducibility on Compositions* (*NDC*, for short) was proposed. Intuitively, a system is *NDC* if, by *interacting* with every possible high level user, it always *appears* the same to low level users, so that no information at all can be deduced by low level users. The above idea can be instantiated in a lot of ways, by choosing a particular way of interacting between systems and various criteria of equivalence. First we consider trace equivalence, thus *NDC* is described in terms of CCS process algebra [24]as follows:

$$E \text{ is } NDC \text{ iff } \forall\, \Pi \in \ \textit{High} \text{ users },\, (E\|\Pi)\backslash H \approx_T E\backslash H$$

where $H$ is a set of high actions. It is possible to give the definition of *NDC* by using contexts.

**Definition 2.10** *Let $Sort(\Pi)$ be the set of actions that occurs in $\Pi$ and let $H$ be the set of high actions. Let $C^1_{0,H} = \{\Pi \mid Sort(\Pi) \subseteq H \cup \{\tau\}\}$ be the set of High users. $E \in C^1_0$ is* NDC *if and only if* $\forall\, \Pi \in C^1_{0,H} \backslash_H(\|(E \times \Pi)) \approx_T \backslash_H(E)$

We can obtain a bisimulation based *NDC* by simply replacing $\approx_T$ with $\approx$. In particular, in [6, 7], the authors argue that *BNDC* is the right choice. Also in this case we give the definition of *BNDC* by exploiting the semantics of contexts as follows.

**Definition 2.11** *Let $Sort(\Pi)$ be the set of actions that occurs in $\Pi$ and let $H$ be the set of high actions. Let $C^1_{0,H} = \{\Pi \mid Sort(\Pi) \subseteq H \cup \{\tau\}\}$ be the set of High users. $E \in C^1_0$ is* BNDC *if and only if* $\forall \Pi \in C^1_{0,H}$ *we have* $\backslash_H(\|(E\times\Pi)) \approx \backslash_H(E)$.

**Proposition 2.1 ([6, 7])** *Let $C \in C^1_0$ be a context.*

$$C \in BNDC \implies C \in NDC$$

The viceversa does not hold, as the following example shows.

**Example 2.7** *([6, 7]) Let $E = +((\tau^* \circ l^* \circ Nil) \times (\tau^* \circ h^* \circ l^* \circ Nil))$ be a context in $C^1_0$. According to [6, 7], $E \in NDC$. However $E \notin BNDC$; indeed if we consider $\Pi = \bar{h}^* \circ Nil$, then $\backslash_H(\|(E\times\Pi))$ behaves as $+((\tau^* \circ l^* \circ Nil) \times (\tau^* \circ \tau^* \circ l^* \circ Nil))$ while $\backslash_H(E)$ behaves as $+((\tau^* \circ l^* \circ Nil) \times (\tau^* \circ Nil))$.* ■

## 3 Specification of Information flow property for distributed systems

In this section we want to extend the definitions of *NDC* and *BNDC* given for processes to the case of partially specified distributed systems described here as context.

**Example 3.1** *Let us consider a system:*

$$C = ((h_0^* \circ l_0^*)^\dagger \| X_0)\backslash_{\{h_0,\bar{h_0}\}} \| ((h_1^* \circ l_1^*)^\dagger \| X_1)\backslash_{\{h_1,\bar{h_1}\}}$$

*where, in order to make this example more readable, we use the infix notation. Let us interpret $l_0$ and $l_1$ as bit 0 and 1 respectively. According to how we choose $X_0$ and $X_1$ it is possible to generate sequences of 0 and 1 obtaining (possible infinite) traces that represent information that (indirectly) flows from high to low. As a matter of fact, the high level user is able to generate a string of 0s and 1s that represents a message that a low level user receives.* ■

### 3.1 Specification of *NDC* for distributed systems

According to the definition of *NDC*, there is a universal quantification on all possible high users. Hence it is possible to specify the *NDC* property as a context $S = \backslash_H(\|(E \times \_))$, where there is a hole in which we have to consider a high user.

Here we analyze a partially specified system $C$ in which there are more than one high user. Indeed we consider the unspecified components of the context $C$ as high users, *i.e.*, if $C \in C_n^1$ there are $n$ high processes in $C_0^1$.

There is however some flexibility in the way the unspecified components can behave. We consider two different approaches: The *centralized* approach, $CNDC$, and the *decentralized* one, *DNDC*.

First we give the following notational definition.

**Definition 3.1** *Let $H \subseteq Act$ be the set of high actions and let $H^n \subseteq Act^n$ be the set of n-tuples of high actions. The context $\backslash_{H^n} \in C_n^n$ is defined by the following rule:*

$$\backslash_{H^n} \xrightarrow[\tilde{a}]{\tilde{a}} \backslash_{H^n} \quad \tilde{a} \notin H^n$$

*where $\tilde{a} \notin H^n$ means that, being $\tilde{a} = (a_1, \ldots, a_n)$ there does not exist any $a_i \in H$ for $i = 1, \ldots, n$.*

Let us start with the centralized one.

**Definition 3.2** *Let $C \in C_n^m$ be a generic context. $C \in CNDC$ iff*

$$\forall X \in C_{0,H}^n \quad \backslash_{H^m} C(X) \approx_T \backslash_{H^m} C(\tilde{Nil})$$

*where $\tilde{Nil}$ is the n-ary context that does not perform any action.*

As before we have a universal quantification in the specification that is difficult to manage. Referring to the theory developed in [9], we want to give a static characterization of *CNDC* by using a particular context $Top^n \in C_0^n$ that is a a feed-back context on all possible $n$-tuples of high actions. Its semantics definition in pseudo-CCS[1] is $Top^n = \sum_{\tilde{a} \in H^n} \tilde{a}^* \circ Top^n$, where $H^n$ is a set of typles of actions in $H$, and semantically defined as follows:

$$Top^n \xrightarrow{\tilde{a}} Top^n$$

for any $\tilde{a} \in H^n$. This context allows for all possible $n$-tuples of high actions.

The following result holds, whose proof is postponed to the Appendix.

**Proposition 3.1** *Let $C \in C_n^m$ be a generic context.*

$$C \in CNDC \Leftrightarrow \backslash_{H^m} C(Top^n) \approx_T \backslash_{H^m} C(\tilde{Nil})$$

This means that the *NDC* property is statically characterized by a single context, namely the $Top^n$ context. In this way the universal quantification on all possible high users is embedded into the context $Top^n$.

Now, let us consider the decentralized approach.

**Definition 3.3** *Let $C \in C_n^m$ be a generic context. $C \in$ DNDC iff*

$$\forall X_1, \ldots, X_n \in C_{0,H}^1 \quad \backslash_{H^m} C(X_1, \ldots, X_n) \approx_T \backslash_{H^m} C(Nil, \ldots, Nil)$$

Also in this case, by exploiting the context $Top^1$ we prove (see the Appendix) the following result.

**Proposition 3.2** *Let $C \in C_n^m$ be a generic context.*

$$C \in DNDC \Leftrightarrow \backslash_{H^m} C(Top^1, \ldots, Top^1) \approx_T \backslash_{H^m} C(Nil, \ldots, Nil)$$

Hence, also in the decentralized case, it is possible to statically characterize *DNDC*.

In order to study the relation that exists between *CNDC* and *DNDC* we give the following proposition (proof in the Appendix).

**Proposition 3.3** *Let $C \in C_n^m$ be a generic context.*

$$C \in CNDC \Leftrightarrow C \in DNDC$$

Hence there is no difference between the two approaches if we consider trace equivalence as the behavioral relation between contexts in the analysis of the information flow properties. This means that, in this setting, there is no difference if the system is attacked by $n$ independent malicious agents with no knowledge of each other or by $n$ attackers that manage to violate a system in a collaborative way.

In the next subsection, we will show that this is no longer the case if bisimulation is used in place of trace semantics. Hence, according to what Focardi and Gorrieri have already concluded about the analysis of systems with only one high user ([6, 7]), also in presence of several high components, the *BNDC* property turns out to be more interesting and appropriate than *NDC*.

[1]We define it in CCS because the context notations makes this definition too cumbersome.

## 3.2 Specification of *BNDC* for distributed systems

According to [8, 20, 21, 22, 25], the *BNDC* property can be analyzed by using the *open system* paradigm (see [20, 21, 22, 25]).

As we have already done in the previous section for the *NDC* property, we extend the specification of *BNDC* property to distributed systems by considering that more than one high level user interacts with the system. Also in this case, we consider these high level users as unspecified components of a distributed system modelled by a context, *e.g.*, if $C \in C^1_n$ there are $n$ high processes in $C^1_0$.

We distinguish between a centralized notion of *BNDC* (*CBNDC*) and a decentralized one (*DBNDC*).

**Definition 3.4** *Let $C \in \mathcal{C}^m_n$ be a context and let $\mathcal{C}^n_{0,H}$ be the set of $n$-ary high contexts. C is CBNDC iff:*

$$\forall X \in \mathcal{C}^n_{0,H} \setminus_{H^m}(C(X)) \approx \setminus_{H^m}(C(\tilde{Nil}))$$

*where $\tilde{Nil}$ is the $n$-ary context that does not perform any action.*

**Definition 3.5** *Let $C \in \mathcal{C}^m_n$ be a context and let $\mathcal{C}^1_{0,H}$ be the set of unary high contexts. C is DBNDC if and only if:*

$$\forall X_1, \dots, X_n \in \mathcal{C}^1_{0,H} \setminus_{H^m}(C \circ (X_1 \times \dots \times X_n)) \approx \setminus_{H^m}(C \circ (Nil \times \dots \times Nil))$$

*where, according to the rule of the product operation, the product $X_1 \times \dots \times X_n$ is a context in $\mathcal{C}^n_{0,H}$*

We show that *CBNDC* is a property strictly finer than *DBNDC*. First, the next proposition shows the implication *CBNDC* $\Rightarrow$ *DBNDC*. Then, the following example shows that the reverse implication does not hold.

**Proposition 3.4** *Let C be a context in $\mathcal{C}^m_n$. If C is CBNDC, then C is also DBNDC.*

*Proof*: It is enough to observe that for any $D \in \mathcal{C}^1_{0,H} \times ... \times \mathcal{C}^1_{0,H}$ (for $n$ times), there exists $D' \in \mathcal{C}^n_{0,H}$ such that $D$ and $D'$ are strongly bisimilar. $\square$

However the vice-versa of the Proposition 3.4 does not hold as the following example shows.

**Example 3.2** *Let us consider now a context $D \in C^1_2$ such that $D = \tau^* \circ l^*_1 \circ Nil + \tau^* \circ l^*_2 \circ Nil + (\tau^* \circ l^*_1 \circ Nil \| \tau^* \circ l^*_2 \circ Nil) + C$ where C is the same context of Example 2.5.*

*In order to make this example readable we use the infix notation that is more suitable than the prefix one in order to point out why the two approaches differ.*

*As in Example 2.5, first we combine D with another context $P = (\bar{h_1}, 0)^* \circ (\bar{h_1}, 0)^* \circ (0, \bar{h_2}) \in C^2_0$. Here we show that D is not* CBNDC. *As a matter of fact by calculating $D \circ \tilde{Nil}$ we obtain $\tau^* \circ l^*_1 \circ Nil + \tau^* \circ l^*_2 \circ Nil + (\tau^* \circ l^*_1 \circ Nil \| \tau^* \circ l^*_2 \circ Nil)$.*

*On the contrary, by considering* $P = (\bar{h_1}, 0)^* \circ (\bar{h_1}, 0)^* \circ (0, \bar{h_2})$*, we obtain that* $D \circ P$ *is* $\tau^* \circ l_1^* \circ Nil + \tau^* \circ l_2^* \circ Nil + (\tau^* \circ l_1^* \circ Nil \| \tau^* \circ l_2^* \circ Nil) + \tau^* \circ l_1^* \circ \tau^* \circ \tau^* \circ l_2^*$ *that it is not weakly bisimilar to* $D \circ \tilde{Nil}$*.*

*Now, we want to prove that* $D$ *is* DBNDC*. It is not difficult to note that the possible contexts that can interact with* $D$ *and make an attack are* $P_1 = \bar{h_1}^* \circ \bar{h_1}^* \circ Nil$ *and* $P_1' = \bar{h_1}^* \circ Nil$*, for the first component, and* $P_2 = \bar{h_2}^* \circ Nil$ *for the second one. It is possible to prove that both* $\backslash_H(D(P_1, P_2)) \approx \backslash_H(D(Nil, Nil))$ *and* $\backslash_H(D(P_1', P_2)) \approx \backslash_H(D(Nil, Nil))$*. Hence* $C$ *is* DBNDC[2]*.*

*First of all we calculate:*

$$\backslash_H(D(Nil, Nil)) = \tau^* \circ l_1^* \circ Nil + \tau^* \circ l_2^* \circ Nil + (\tau^* \circ l_1^* \circ Nil \| \tau^* \circ l_2^* \circ Nil)$$

*In the first case, by calculating* $D \circ (P_1 \times P_2)$ *we obtain:*

$$\tau^* \circ l_1^* \circ Nil + \tau^* \circ l_2^* \circ Nil + (\tau^* \circ l_1^* \circ Nil \| \tau^* \circ l_2^* \circ Nil) + (\tau^* \circ l_1^* \circ \tau^* \circ Nil \| \tau^* \circ l_2^* \circ Nil)$$

*that it is weakly bisimilar to* $\tau^* \circ l_1^* \circ Nil + \tau^* \circ l_2^* \circ Nil + (\tau^* \circ l_1^* \circ Nil \| \tau^* \circ l_2^* \circ Nil)$

*In the second case, by calculating* $D \circ (P_1' \times P_2)$ *we obtain:*

$$\tau^* \circ l_1^* \circ Nil + \tau^* \circ l_2^* \circ Nil + (\tau^* \circ l_1^* \circ Nil \| \tau^* \circ l_2^* \circ Nil) + (\tau^* \circ l_1^* \circ Nil \| \tau^* \circ l_2^* \circ Nil)$$

*that it is weakly bisimilar to* $\tau^* \circ l_1^* \circ Nil + \tau^* \circ l_2^* \circ Nil + (\tau^* \circ l_1^* \circ Nil \| \tau^* \circ l_2^* \circ Nil)$*.*

*It is interesting to note that the context* $P \in C_0^2$ *is an example of a process that can not be written as the product of two unary contexts,* i.e.*, there do not exist* $P_1, P_2 \in C_0^1$ *such that* $P_1 \times P_2$ *is weakly bisimilar to* $P$*. Hence the main difference between the* CBNDC *and* DBNDC *is that, in the centralized case, the universal quantification is made on contexts in* $C_0^n$ *semantically strictly larger than* $\underbrace{C_0^1 \times \ldots \times C_0^1}_{n}$*.* ■

To sum up, if we consider the trace equivalence as behavioral equivalence among contexts, we do not have any difference between the centralized and decentralized approaches for information flow specification, *i.e.*, there are no differences between *CNDC* and *DNDC*. On the other hand, if we consider bisimulation equivalence, the two approaches are different. This depends on the fact that it is not possible to give a static characterization of *BNDC* in terms of a finite context (see [9]). Since the *BNDC* property is a particular case of both *CBNDC* and *DBNDC*, we can conclude that there is no finite context playing the same role as $Top^n$ for the trace case for statically characterizing neither *CBNDC* nor *DBNDC*.

## 4 Analysis of Information Flow for distributed systems

In this section we focus our attention on the analysis of *BNDC* in distributed systems. In particular, we analyze *CBNDC* and *DBNDC* by following a logical

[2]To be complete we have also to prove that $\backslash_H(D(Nil, P_2)) \approx \backslash_H(D(Nil, Nil))$, $\backslash_H(D(P_1, Nil)) \approx \backslash_H(D(Nil, Nil))$ and $\backslash_H(D(P_1', Nil)) \approx \backslash_H(D(Nil, Nil))$. These follow obviously, so we decide to show the two more difficult cases.

approach based on characteristic formulas.

It is important to note that the analysis of $CNDC$ and $DNDC$ consists of proving if two closed systems are trace equivalent or not (Proposition 3.1 and Proposition 3.2). In the literature there are a lot of mechanisms that performs this kind of check (see *e.g.*, [18]). So it is not the matter of this paper.

## 4.1 Analysis of $CBNDC$ property

For the analysis of the *CBNDC* property we propose a logical approach exploiting characteristic formulas.

Since the context $\backslash_{H^m}(C(\tilde{Nil}))$ is a closed context in $C_0^m$ we are able to find its characteristic formula $\tilde{\phi}$. Hence, it is possible to give a logical specification of the $CBNDC$ property equivalent to the one given in the Definition 3.4 as follows.

$$C \in C_n^m \in CBNDC \text{ iff } \forall X \in C_{0,H}^n \quad \backslash_{H^m}(C(X)) \models \tilde{\phi} \tag{6}$$

In order to solve this problem, we exploit the property transformer function (see Section 2.2.2) in such a way that we reduce this problem to a validity one. For simplicity of notation, let $D = \backslash_{H^m} \circ C$. Hence we have the following reduction:

$$\forall X \in C_{0,H}^n \quad X \models \mathcal{W}(D, \tilde{\phi}) \tag{7}$$

In this case, since $X$ can be seen as a process that performs a $n$-tuple of actions instead of a single one, it is possible to apply an already existing verification method for modal logic (*e.g.*, see [3]).

**Example 4.1** *Coming back to Example 3.1, let us consider again the following context $C \in \mathcal{C}_2^1$.*

$$((h_0^* \circ l_0^*)^\dagger \| X_0)\backslash_{\{h_0,\bar{h_0}\}} \| ((h_1^* \circ l_1^*)^\dagger \| X_1)\backslash_{\{h_1,\bar{h_1}\}}$$

*where $X_0$ and $X_1$ are two variable in $\mathcal{C}_0^1$. In order to verify if this context is* CB-NDC*, we consider the characteristic formula of $\backslash_{\{h_0,\bar{h_0},h_1,\bar{h_1}\}}(C(\tilde{Nil}))$ that is $\mathbf{F}$. In this case we can easily conclude that $C$ is not* CBNDC*.*

## 4.2 Analysis of $DBNDC$ property

Also in the decentralized case, a universal quantification occurs and also in this case we appeal to the notion of characteristic formulas. As a matter of fact, we consider the characteristic formula $\tilde{\varphi}$ of the context $\backslash_{H^m}(C(Nil \times \ldots \times Nil)) \in C_0^m$. As before we apply the property transformer function for evaluating the behavior context $D = \backslash_{H^m} \circ C$. Hence we have to solve the following problem:

$$\forall \{X_1, \ldots, X_n\} \subseteq C_{0,H}^1 \quad X_1 \times \ldots \times X_n \models \mathcal{W}(D, \tilde{\varphi}) \tag{8}$$

It is interesting to note that Statement (8) is a particular instance of the problem in Statement (7) and so it is solvable as before.

However, since in the decentralized case we have $n$ independent components that interact with the system, we would like to solve our problem by considering

$n$ unary validity problem instead of a $n$-ary one. To do this we should find a decomposition of the formula $\tilde{\varphi'} = \mathcal{W}(D, \tilde{\varphi})$ into unary formulas $\phi'_1, \dots, \phi'_n$ such that:

$$\models_\times \phi'_1 \times \dots \times \phi'_n \Leftrightarrow \tilde{\varphi'}$$

where each $\phi'_1, \dots, \phi'_n$ has to be satisfied independently by the components $X_1, \dots, X_n$.

To do this, we find which property each component has to satisfy in order to guarantee that the whole system is $DBNDC$, *i.e.*, it satisfies $\mathcal{W}(D, \tilde{\varphi}) = \tilde{\varphi}'$.

We exploit the procedure of decomposition of an $n$-ary formula $\phi$ described in [17].

**Definition 4.1** *Given an $n$-ary formula $\phi$, we look for a* product formula, *$\phi_1 \times \dots \times \phi_n$ where each $\phi_i$, $i = 1, \dots, n$, is a unary formula in such a way that we have:*

$$Q \in [\![\phi_1 \times \dots \times \phi_n]\!] \Leftrightarrow \exists P_1 \in [\![\phi_1]\!] \dots \exists P_n \in [\![\phi_n]\!] \text{ s. t. } Q \sim P_1 \times \dots \times P_n$$

*where $Q$ is a $n$-ary contexts and $P_1, \dots, P_n$ are unary contexts.*

For a product formula $\phi$, we write $\models \phi$ if $\phi$ is satisfied by all $n$-ary processes of any context system. In this case, we say that $\phi$ is *valid.* Moreover $\models_\times \phi$ if $\phi$ is satisfied by all $n$-product processes $P_1 \times \dots \times P_n$ of any context system. In this case $\phi$ is *weakly valid.* According to [17], note that, for a product formula $\phi$, the notions of validity and weak validity coincide.

The quest of decomposition can be formally stated as the search for a single product formula $\phi_1 \times \dots \times \phi_n$ such that:

$$\models_\times \phi_1 \times \dots \times \phi_n \Leftrightarrow \phi$$

Usually there does not exist a single product formula $\phi_1 \times \dots \times \phi_n$ such that $\models_\times \phi_1 \times \dots \times \phi_n \Leftrightarrow \mathcal{W}(C, \phi)$. Whenever $\phi$ is a *finite* formula, *i.e.*, it is neither LET MAX nor LET MIN, there exists a finite decomposition, *i.e.*, a finite collection of product formulas $\langle \phi^i_1 \times \dots \times \phi^i_n \rangle_{i \in I}$, where $I$ is and index set, such that

$$\models_\times \bigvee_{i \in I} \phi^i_1 \times \dots \times \phi^i_n \Leftrightarrow \mathcal{W}(C, \phi)$$

**Definition 4.2** *A disjunctive product formula, $\bigvee_{i \in I} \phi^i_1 \times \dots \times \phi^i_n$ is said to be* saturated, *provided that for any product formula $\varphi_1 \times \dots \times \varphi_n$*

$$\models \varphi_1 \times \dots \times \varphi_n \Rightarrow \bigvee_{i \in I} \phi^i_1 \times \dots \times \phi^i_n$$

*is equivalent to*

$$\models \varphi_1 \Rightarrow \phi^i_1 \ \dots \ \models \varphi_n \Rightarrow \phi^i_n \text{ for some } i \in I$$

| | | |
|---|---|---|
| $(i) \models_\times \mathbf{T}$ | $\Leftrightarrow$ | $\mathbf{T} \times \mathbf{T}$ |
| $(ii) \models_\times \mathbf{F}$ | $\Leftrightarrow$ | $\mathbf{T} \times \mathbf{F} \vee \mathbf{F} \times \mathbf{T}$ |
| $(iii) \models_\times \phi_1 \times \psi_1 \wedge \phi_2 \times \psi_2$ | $\Leftrightarrow$ | $(\phi_1 \wedge \phi_2) \times (\psi_1 \wedge \psi_2)$ |
| $(iv) \models_\times \langle (a,b) \rangle (\phi \times \psi)$ | $\Leftrightarrow$ | $(\langle a \rangle \phi) \times (\langle b \rangle \psi)$ |
| $(v) \models_\times [(a,b)](\bigvee_i \phi^i \times \psi^i)$ | $\Leftrightarrow$ | $\bigvee_i ([a]\phi^i) \times ([b]\psi^i)$ |

where in $(v)$ $\bigvee_i \phi^i \times \psi^i$ is assumed to be saturated.

Table 4: Weak equivalences for decomposition of properties (see [17]).

It is important to note that every general disjunctive product formula can be saturated [17]. Moreover, there exist weak equivalences that provide the means for decomposing properties. Table 4 reports the equivalences that, according to [17], are all weakly valid.

The following theorem holds. The interested reader can find all the theory in [17].

**Theorem 4.1** *Let $\phi$ be a $n$-ary finite formula. Then there exists a disjunctive product formula $\bigvee_{i \in I} \varphi_1^i \times \varphi_n^i$ weakly equivalent to $\phi$ such that*

$$\models_\times \phi \Leftrightarrow \bigvee_{i \in I} \varphi_1^i \times \ldots \times \varphi_n^i$$

Hence, whenever $\tilde{\varphi}' = \mathcal{W}(D, \tilde{\varphi})$ is a finite formula, by applying Theorem 4.1 we have that $\models_\times \tilde{\varphi}' \Leftrightarrow \bigvee_{i \in I} \varphi_1^{'i} \times \ldots \times \varphi_n^{'i}$. Moreover, by considering that every disjunctive product formula can be saturated, it is possible to solve the problem in Statement (8) by solving the following one:

$$\forall \{X_1, \ldots, X_n\} \subseteq C^1_{0,H} \quad X_1 \times \ldots \times X_n \models_\times \phi'_1 \times \ldots \times \phi'_n$$

where $\phi'_1 \times \ldots \times \phi'_n$ implies $\bigvee_{i \in I} \varphi_1^{'i} \times \ldots \times \varphi_n^{'i}$. Hence, according to Definition 4.1, the verification problem in Statement (8) can be reduced to $n$ problems of the form:

$$\forall \ X \in C^1_0 \quad X \models \phi'_i \ i = 1, \ldots, n$$

This is a classical validity problem that, according to [3, 27], for finite state processes is decidable. Here we follow a similar reasoning made in [20, 21, 22, 25] for analyzing *BNDC*. Then, by finding a solution for each of these $n$ problems we obtain the solution for the general problem.

In order to explain how this theory works, we show the following example.

**Example 4.2** *Let us consider the context $D \in C^1_2$ of Example 3.2 that is not* DBNDC. *To prove that, firstly we calculate the characteristic formula $\varphi$ of the context $D(Nil, Nil)$.*

$$\varphi = \text{LET MAX } X_D \text{ IN } \phi$$

*where*

$$\phi = \langle\langle l_1\rangle\rangle\mathbf{T} \wedge \langle\langle l_2\rangle\rangle\mathbf{T} \wedge \langle\langle l_1\rangle\rangle\langle\langle l_2\rangle\rangle\mathbf{T} \wedge \langle\langle l_2\rangle\rangle\langle\langle l_1\rangle\rangle\mathbf{T}$$

*According to the analysis theory we have presented above, $D \in DBNDC$ when*

$$\forall\{X_1, X_2\} \subseteq C^1_{0,H} \quad X_1 \times X_2 \models \mathcal{W}(D, \varphi)$$

*where in this case*

$$\tilde{\varphi}' = \mathcal{W}(D, \varphi) = \langle h_1, 0\rangle\mathbf{T} \wedge \langle 0, h_2\rangle\mathbf{T} \wedge \langle h_1, h_2\rangle\mathbf{T}$$

*Since $\phi$ does not depend by $X_D$, we can consider $\varphi$ a finite formula. According to the weak equivalences in Table 4, a decomposition of $\tilde{\varphi}'$ is the following:*

$$\langle h_1\rangle\mathbf{T} \times \mathbf{T} \wedge \mathbf{T} \times \langle h_2\rangle \wedge \langle h_1\rangle\mathbf{T} \times \langle h_2\rangle\mathbf{T} \;\Leftrightarrow\; \langle h_1\rangle\mathbf{T} \times \langle h_2\rangle\mathbf{T}$$

*Hence solving the following statement*

$$\forall\{X_1, X_2\} \subseteq C^1_{0,H} \quad X_1 \times X_2 \models \langle h_1\rangle\mathbf{T} \times \langle h_2\rangle\mathbf{T}$$

*is equivalent to solve the following two ones:*

$$\begin{array}{ll} \forall X_1 & X_1 \models \langle h_1\rangle\mathbf{T} \\ \forall X_2 & X_2 \models \langle h_2\rangle\mathbf{T} \end{array}$$

*Since neither of these statements hold for all implementation of the unknown components $X_1$ and $X_2$, we conclude that $D$ is not* DBNDC.

**Example 4.3** *Coming back to Example 3.1, let us consider again the following context $C \in \mathcal{C}^1_2$.*

$$((h_0^* \circ l_0^*)^\dagger \| X_0)\backslash_{\{h_0, \bar{h_0}\}} \| ((h_1^* \circ l_1^*)^\dagger \| X_1)\backslash_{\{h_1, \bar{h_1}\}}$$

*where $X_0$ and $X_1$ are two variables in $\mathcal{C}^1_0$. In order to verify if this context is* DBNDC*, we consider the characteristic formula of $\backslash_{\{h_0, \bar{h_0}, h_1, \bar{h_1}\}}(C(Nil \times Nil))$ that is $\mathbf{F}$. In this case we can easily conclude that $C$ is not* DBNDC. ■

Now, let us consider the case when $\tilde{\varphi}'$ is not a finite formula. According to [17], a *safe* (or sufficient) decomposition may be established. This means that is possible to find a product formula $\phi'_1 \times \ldots \times \phi'_n$ such that

$$\models_\times \phi'_1 \times \ldots \times \phi'_n \Rightarrow \tilde{\varphi}'$$

In this case, $\phi'_1 \times \ldots \times \phi'_n$ is said to be the safe decomposition of $\phi$.

When we are considering safe decompositions the concept of weak validity and validity coincide. We prove the following result (see Appendix).

**Theorem 4.2** *Let $\tilde{\varphi}$ be a generic $n$-ary formula in the considered modal logic. Then there exists a safe decomposition of $\tilde{\varphi}$,* i.e.*, there exists a product formula $\phi_1 \times \ldots \times \phi_n$ such that*

$$\models \phi_1 \times \ldots \times \phi_n \Rightarrow \tilde{\varphi}$$

Thus, also when $\tilde{\varphi}' = \mathcal{W}(D, \tilde{\varphi})$ is not a finite formula we are able to decompose it in such a way that the following holds:

$$\forall \{X_1, \ldots, X_n\} \subseteq C^1_{0,H} \quad X_1 \times \ldots \times X_n \models \phi'_1 \times \ldots \times \phi'_n$$

Hence, also in this case, the verification problem in Statement (8) can be reduced to $n$ problems of the form:

$$\forall\, X \in C^1_0 \quad X \models \phi'_i \; i = 1, \ldots, n$$

Finding a solution for each of these $n$ problems we have the solution of the general problem. As a matter of fact, once we prove that each formula $\phi'_i$ is valid we are able to conclude that the product $\phi'_1 \times \ldots \times \phi'_n$ is valid too, remembering that concepts of validity and weak validity coincide on product formulas. Hence, since $\phi'_1 \times \ldots \times \phi'_n$ is a safe decomposition for $\tilde{\varphi}'$, we have found a safe condition for verifying *DBNDC*. This means that if the safe decomposition is valid then the system is *DBNDC*.

## 5 Related work

A lot of work has been done in order to specify and analyze information flow properties such as *NDC* and *BNDC*, *e.g.*, see [20, 21, 22, 25]. Most of this work, however, is done by considering only one possible high user that interacts with the system.

In [2] the authors have studied which could be the effect of the environment on the information flow security in a multilevel system. They introduce the notion of *secure contexts* for a class of processes as a parametric notion with respect to both the observation equivalence and the operation to characterize the low level views of a process. In particular they also show that *BNDC* and *NDC*, are just special instances of the general notion. Our work can be considered an extension of this one in which we consider more than one high user.

In [10] the authors have studied the information flow properties in the setting of mobile agents. They have considered a distributed system with several unspecified locations and have studied when the system is secure with respect to information flow considering that an agent performs a different action for each different location. They refer to this property as mobile $BNDC$, $M\_BNDC$ for short. Moreover they study the relation that exists between *Persistent* $BNDC$ and $M\_BNDC$. However they do not present any verification method in order to analyze $M\_BNDC$. Moreover in our approach, there are several high users that can also interact. This is not the case of $M\_BNDC$ in which a single process passes from a location to another by performing a set of actions in each location, sequentially.

In [13, 14, 15], the author aims to formalize a notion of "network timing attacks". [13] introduces for the first time a particular process algebra, called *Network Security Process Algebra (nSPA)* for reasoning about security properties and, in particular, information flow properties in a timed setting. In particular, it investigates different locations of a process in systems with interconnection networks of heterogeneous structure where some connections might influence the system security. [14] presents a formal model for description of passive and active timing attacks, that in [15] is extended by introducing the concept of probability in the definition and analysis of such attacks. Hence, the author formulates information flow in terms of probabilities. In particular, [13] it is studies the case in which an attacker exploits several high level processes that can cooperate. Our work exploits context theory, in place of *nSPA*, to specify the presence of several high level processes that interact with the given system. In particular, we contribute on this line of research by specifying and analyzing information flow properties also when there are several high level processes but they work independently on each other.

In [28], Tini pointed out that in general $BNDC$ is not compositional with respect to all the process algebras operators. This results is an extension of the one given in [22], in which there is the proof that $BNDC$ is not compositional with respect to parallel operators. Moreover, in [28], he has defined several properties stronger than $BNDC$ in order to have properties that are compositional with respect to process algebras operators. In this work we have proposed a different specification of the problem that takes into account the environment of the system.

## 6 Conclusion and future work

In this paper we present a possible extension of the approach, based on the open system paradigm used to specify and analyze information flow properties with one high level user, to deal with such a systems in which more than one high level user is active.

In particular, in this research, we focus our attention on $BNDC$ and $NDC$ properties by showing methods to specify and verify both *centralized* and *decentralized* $NDC$ and $BNDC$.

We aim to extend this work by considering that some unspecified components of the analyzed partially specified system can perform low actions. In particular we could consider that the unspecified components of the system are not only possible malicious agents but could also be generic processes that can perform high actions as well as low ones.

Moreover we intend to deal with enforcement mechanisms for monitoring information flow properties in distributed system. In particular, we would like to extend the work in [23] to define and synthesize controller operators also for the properties defined in this work.

## References

[1] Andersen, H. R.: Partial Model Checking, *LICS '95: Proceedings of the 10th Annual IEEE Symposium on Logic in Computer Science*, IEEE Com-

puter Society, 1995, ISBN 0-8186-7050-6.

[2] Bossi, A., Macedonio, D., Piazza, C., Rossi, S.: Information flow in secure contexts, *J. Comput. Secur.*, **13**(3), 2005, 391–422, ISSN 0926-227X.

[3] Bradfield, J., Stirling, C.: *Handbook of Process Algebra*, chapter Modal logics and mu-calculi: an introduction, Elsevier, 2001, 293–330.

[4] Focardi, R.: *Analysis and Automatic Detection of Information Flows in Systems and Networks*, Ph.D. Thesis, Department of Computer Science, University of Bologna, 1998.

[5] Focardi, R., Gorrieri, R.: A Classification of Security Properties for Process Algebras, *Journal of Computer Security*, **3**(1), 1994/1995, 5–33.

[6] Focardi, R., Gorrieri, R.: The Compositional Security Checker: A Tool for the Verification of Information Flow Security Properties, *IEEE Transactions on Software Engineering*, **27**, 1997, 550–571.

[7] Focardi, R., Gorrieri, R.: Classification of Security Properties (Part I: Information Flow), *FOSAD '00: In International School on Foundations of Security Analysis and Design*, 2171, Springer-Verlag, 2001, ISBN 3-540-42896-8.

[8] Focardi, R., Gorrieri, R., Martinelli, F.: Classification of Security Properties - Part II: Network Security, *FOSAD II*, 2002.

[9] Focardi, R., Martinelli, F.: A Uniform Approach for the Definition of Security Properties, *FM '99: Proceedings of the Wold Congress on Formal Methods in the Development of Computing Systems-Volume I*, Springer-Verlag, London, UK, 1999, ISBN 3-540-66587-0.

[10] Focardi, R., Rossi, S.: Information Flow Security in Dynamic Contexts, *CSFW '02: Proceedings of the 15th IEEE workshop on Computer Security Foundations*, IEEE Computer Society, Washington, DC, USA, 2002, ISBN 0-7695-1689-0.

[11] Goguen, J. A., Meseguer, J.: Security Policy and Security Models, *Proc. of the 1982 Symposium on Security and Privacy*, IEEE Press, 1982.

[12] Gorrieri, R., Martinelli, F., Matteucci, I.: Towards information flow properties for distributed systems, *In Proceedings of the 3rd VODCA Views On Designing Complex Architectures*, 2008, To appear.

[13] Gruska, D. P.: Network Information Flow, *Fundam. Inform.*, **72**(1-3), 2006, 167–180.

[14] Gruska, D. P.: Observation Based System Security, *Fundam. Inform.*, **79**(3-4), 2007, 335–346.

[15] Gruska, D. P.: Probabilistic Information Flow Security, *Fundam. Inform.*, **85**(1-4), 2008, 173–187.

[16] Hennessy, M., Milner, R.: Algebraic laws for nondeterminism and concurrency, *J. ACM*, **32**(1), 1985, 137–161, ISSN 0004-5411.

[17] Larsen, K. G., Xinxin, L.: Compositionality Through an Operational Semantics of Contexts, *Journal of Logic and Computation*, **1**(6), December 1991, 761–795.

[18] Levy, P. B.: Infinite Trace Equivalence, *21st Annual Conference on Mathematical Foundations of Programming Semantics (MFPS XXI)*, 155, Elsevier Science B.V., May, 2006.

[19] Lowe, G.: Semantic models for information flow, *Theor. Comput. Sci.*, **315**(1), 2004, 209–256, ISSN 0304-3975.

[20] Martinelli, F.: *Formal Methods for the Analysis of Open Systems with Applications to Security Properties*, Ph.D. Thesis, University of Siena, Dec. 1998.

[21] Martinelli, F.: Partial Model Checking and Theorem Proving for Ensuring Security Properties, *CSFW '98: Proceedings of the 11th IEEE workshop on Computer Security Foundations*, IEEE Computer Society, Washington, DC, USA, 1998, ISBN 0-8186-8488-7.

[22] Martinelli, F.: Analysis of security protocols as open systems, *Theoretical Computer Science*, **290**(1), 2003, 1057–1106.

[23] Martinelli, F., Matteucci, I.: Synthesis of Local Controller Programs for Enforcing Global Security Properties, *ARES*, IEEE Computer Society, 2008.

[24] Milner, R.: *Communication and Concurrency*, Prentice Hall, London, 1989.

[25] R.Focardi, R.Gorrieri, F.Martinelli: Real-time Information Flow Analysis, *IEEE JSAC*, **21**(1), Jan 2003, 20–35.

[26] Ryan, P. Y. A., Schneider, S. A.: Process Algebra and Non-interference, *Proceedings of the 1999 IEEE Computer Security Foundations Workshop*, IEEE Computer Society, 1999, ISBN 0-7695-0201-6.

[27] Street, R. S., Emerson, E. A.: The propoitional $\mu$-calculus is elementary, *Eleventh International Colloquim in Automata, Languages and Programming*, **172 of LNCS**, 1984, 465–472.

[28] Tini, S.: Rule formats for compositional non-interference properties, *J. Log. Algebr. Program.*, **60-61**, 2004, 353–400.

[29] Winskel, G.: On the compositional checking of validity (extended abstract), *CONCUR '90: Proceedings on Theories of concurrency : unification and extension*, LNCS 458, Springer-Verlag, Amsterdam, The Netherlands, 1990, ISBN 0-387-53048-7.

## A Technical proofs

Before the proofs of Proposition 3.1 and 3.2 we give the following lemmata.

**Lemma A.1** $\forall X \in C^n_{0,H}\ X \prec Top^n$

*Proof:* Let $\mathcal{S}$ be the binary relation defined as follows:

$$\mathcal{S} = \{(C, Top^n) | C \in C^n_{0,H}\}$$

The thesis follows immediately if relation $\mathcal{S}$ is a strong simulation. But this derives trivially from the definition of $Top^n$, as it can perform any vector of high actions. Indeed, if $C \xrightarrow{\tilde{a}} C'$, then $Top^n \xrightarrow{\tilde{a}} Top^n$ with $(C', Top^n) \in \mathcal{S}$. Hence we have the thesis.

$\square$

**Lemma A.2** *Let $C, D \in C^n_0$ be two contexts. The following hold:*

- $C \prec D \Rightarrow C \leq_T D$
- $C \sim D \Rightarrow C \approx_T D$

*Proof:*

- If $C \prec D$ then, if $C \xrightarrow{\tilde{a}} C'$, there exists $D'$ such that $D \xrightarrow{\tilde{a}} D'$ with $C' \prec D'$. This means that computations of length one performed by $C$ are matched by corresponding computations of length one performed by $D$. Inductively, one can prove that if $C \xrightarrow{\tilde{a_1}} C_1 \xrightarrow{\tilde{a_2}} C_2 \ldots \xrightarrow{\tilde{a_n}} C_n$, then $D \xrightarrow{\tilde{a_1}} D_1 \xrightarrow{\tilde{a_2}} D_2 \ldots \xrightarrow{\tilde{a_n}} D_n$ with $C_n \prec D_n$. This ensures that any trace of $C$, that can be obtained by abstracting on the sequence $\tilde{a_1}\tilde{a_2}\ldots\tilde{a_n}$, is also a trace of $D$.
- For the definition of bisimulation, $C \sim D$ implies that $C \prec D$ and $D \prec C$. Hence, for the previous point of this lemma, we have that $Tr(C) \subseteq Tr(D)$ and $Tr(D) \subseteq Tr(C)$. This implies that $C \approx_T D$.

$\square$

**Proposition 3.1** Let $C \in C^m_n$ a generic context.

$$C \in CNDC \Leftrightarrow \backslash_{H^m} C(Top^n) \approx_T \backslash_{H^m} C(\tilde{Nil})$$

*Proof:* Since

$$C \in CNDC \text{ iff } \forall X \in C^n_{0,H} \quad \backslash_{H^m} C(X) \approx_T \backslash_{H^m} C(\tilde{Nil})$$

the proof is divided into two parts:

$\Rightarrow$ Since $\forall X \in C^n_{0,H}$ $\backslash_{H^m} C(X) \approx_T \backslash_{H^m} C(\tilde{Nil})$, it holds obviously also for $Top^n$. Hence $\backslash_{H^m} C(Top^n) \approx_T \backslash_{H^m} C(\tilde{Nil})$.

$\Leftarrow$ We prove that $\backslash_{H^m} C(\tilde{Nil}) \leq_T \backslash_{H^m} C(X) \leq_T \backslash_{H^m} C(Top^n)$.

For Lemma A.1, $X \prec Top^n$. Since $\prec$ is a pre-congruence for context, *i.e.*, it is a congruence and a pre-order, we have that $\backslash_{H^m} C(X) \prec \backslash_{H^m} C(Top^n)$. Hence, for Lemma A.2, $\backslash_{H^m} C(X) \leq_T \backslash_{H^m} C(Top^n)$.

A similar reasoning is done by noticing that $\tilde{Nil} \prec X$. Hence, by applying the previous lemmata, we have that $\backslash_{H^m} C(\tilde{Nil}) \leq_T \backslash_{H^m} C(X)$.

To sum up $\backslash_{H^m} C(\tilde{Nil}) \leq_T \backslash_{H^m} C(X) \leq_T \backslash_{H^m} C(Top^n) \approx_T \backslash_{H^m} C(\tilde{Nil})$. Hence $\backslash_{H^m} C(X) \approx_T \backslash_{H^m} C(\tilde{Nil})$.

□

**Proposition 3.2** Let $C \in C^m_n$ be a generic context.

$$C \in DNDC \Leftrightarrow \backslash_{H^m} C(Top^1, \ldots Top^1) \approx_T \backslash_{H^m} C(Nil, \ldots Nil)$$

*Proof:* Since

$$C \in DNDC \text{ iff } (\forall X_1, \ldots, X_n \in C^1_{0,H} \quad \backslash_{H^m} C(X_1, \ldots, X_n) \approx_T \backslash_{H^m} C(Nil, \ldots, Nil))$$

the proof is divided into two parts:

$\Downarrow$ Since $\forall X_1, \ldots, X_n \in C^1_{0,H}$ $\backslash_{H^m} C(X_1, \ldots, X_n) \approx_T \backslash_{H^m} C(Nil, \ldots, Nil)$, it holds obviously also for $Top^1$. Hence $\backslash_{H^m} C(Top^1, \ldots, Top^1) \approx_T \backslash_{H^m} C(Nil, \ldots, Nil)$.

$\Uparrow$ We prove that $\backslash_{H^m} C(Nil, \ldots, Nil) \leq_T \backslash_{H^m} C(X_1, \ldots, X_n) \leq_T \backslash_{H^m} C(Top^1, \ldots, Top^1)$.

For Lemma A.1, each $X_i$ for $i = 1, \ldots, n$ $X_i \prec Top^1$. Since $\prec$ is a pre-congruence for the operation of context, *i.e.*, it is a congruence and a pre-order, we have that

$$\backslash_{H^m} C(X_1, \ldots, X_n) \prec \backslash_{H^m} C(Top^1, \ldots, Top^1).$$

Hence, for Lemma A.2, $\backslash_{H^m} C(X_1, \ldots, X_n) \leq_T \backslash_{H^m} C(Top^1, \ldots, Top^1)$.

A similar reasoning is done by noticing that $Nil \prec X_i$ for all $i = 1, \ldots, n$. Hence, applying the previous lemmas, we have that $\backslash_{H^m} C(Nil, \ldots, Nil) \leq_T \backslash_{H^m} C(X_1, \ldots, X_n)$.

To sum up $\backslash_{H^m} C(Nil, \ldots, Nil) \leq_T \backslash_{H^m} C(X_1, \ldots, X_n) \leq_T \backslash_{H^m} C(Top^1, \ldots, Top^1) \approx_T \backslash_{H^m} C(Nil, \ldots, Nil)$. Hence $\backslash_{H^m} C(X_1, \ldots, X_n) \approx_T \backslash_{H^m} C(Nil, \ldots, Nil)$.

□

**Proposition 3.3** Let $C \in C^m_n$ a generic context.

$$C \in CNDC \Leftrightarrow C \in DNDC$$

*Proof:* According to Definition 3.2, $C \in \mathit{CNDC}$ iff

$$\forall X \in C^n_{0,H} \quad \backslash_{H^m} C(X) \approx_T \backslash_{H^m} C(\tilde{Nil})$$

Then, according to Definition 3.3, $C \in \mathit{DNDC}$ iff

$$\forall X_1, \ldots, X_n \in C^1_{0,H} \quad \backslash_{H^m} C(X_1, \ldots, X_n) \approx_T \backslash_{H^m} C(Nil, \ldots, Nil)$$

Hence, to prove Proposition 3.3, we have to prove the following double implication:

$$\backslash_{H^m} C(Top^n) \approx_T \backslash_{H^m} C(\tilde{Nil}) \Leftrightarrow \backslash_{H^m} C(Top^1, \ldots Top^1) \approx_T \backslash_{H^m} C(Nil, \ldots Nil)$$

In particular, we prove that the two contexts $Top^n$ and $\underbrace{Top^1 \times \ldots \times Top^1}_{n}$ are strong bisimilar. Then for Lemma A.2, we conclude that they are also trace equivalent. Hence the two static characterizations coincide.

Hence, let $\mathcal{R}$ be a binary relation defined as follows:

$$\mathcal{R} = \{(Top^n, \underbrace{Top^1 \times \ldots \times Top^1}_{n})\}$$

that we want to prove to be a bisimulation. Initially, $(Top^n, \underbrace{Top^1 \times \ldots \times Top^1}_{n}) \in \mathcal{R}$ obviously.

If $Top^n \xrightarrow{\tilde{a}} Top^n$, where $\xrightarrow{\tilde{a}} = \xrightarrow{(a_1, \ldots, a_n)}$, we have to prove that there exists $C$ such that

$$\underbrace{Top^1 \times \ldots \times Top^1}_{n} \xrightarrow{\tilde{a}} C$$

and $(Top^n, C) \in \mathcal{R}$.

For the semantics definition of $Top^1$, we consider that the first component of the product performs $a_1$, the second one $a_2$, and so on. Hence, from the semantic definition of $Top^1$, we have that $Top^1 \xrightarrow{a_i} Top^1$ for $i = 1, .., n$. Hence, the context $C$ we are looking for is $\underbrace{Top^1 \times \ldots \times Top^1}_{n}$. Hence, the condition $(Top^n, C) \in \mathcal{R}$ holds trivially.

If $\underbrace{Top^1 \times \ldots \times Top^1}_{n} \xrightarrow{\tilde{a}} D$ we want to prove that there exists $Top'^n$ such that $Top^n \xrightarrow{\tilde{a}} Top'^n$ and $(D, Top'^n) \in \mathcal{R}$. Since $D$, for the same reasoning made before, is $\underbrace{Top^1 \times \ldots \times Top^1}_{n}$ and $Top^n$ performs all the possible $n$-tuples of high action and goes into itself, the thesis follows directly from the definition of $Top^n$.

Hence, being $Top^n$ and $\underbrace{Top^1 \times \ldots \times Top^1}_{n}$ bisimilar, since the bisimulation is a congruence for contexts (see Theorem 2.1), we have that

$$\backslash_{H^m} C(Top^n) \sim \backslash_{H^m} (C(Top^1, \ldots Top^1))$$

For Lemma A.2, $\backslash_{H^m}(C(Top^n)) \approx_T \backslash_{H^m}(C(Top^1, \ldots Top^1))$.

Since $\backslash_{H^m}(C(\tilde{Nil})) \approx_T \backslash_{H^m} C(Nil, \ldots Nil)$ trivially, we have the thesis.

□

**Theorem 4.2** Let $\tilde{\varphi}$ be a generic $n$-ary formula in the considered modal logic. Then there exists a safe decomposition of $\tilde{\varphi}$, *i.e.*, there exists a product formula $\phi_1 \times \ldots \times \phi_n$ such that

$$\models \phi_1 \times \ldots \times \phi_n \Rightarrow \tilde{\varphi}$$

First of all, we give the following result used in the proof of the theorem.

**Proposition A.1** *Let $\phi(X)$ and $\psi(X)$ two open modal $\mu$-calculus formulas. If $\phi(X) \Rightarrow \psi(X)$,* i.e.*, $[\![\phi]\!] \subseteq [\![\psi]\!]$, then* $\textsc{let max}\ X = \phi\ \textsc{in}\ X \ \Rightarrow \textsc{let max}\ X = \psi\ \textsc{in}\ X$.

*Proof*: The implication $\phi \Rightarrow \psi$ means that if $S$ is a model of $\phi$, *i.e.*, $S \subseteq [\![\phi]\!]$ then $S$ is also a model for $\psi$. According to the semantics definition of the maximum fix point given in [3]:

$$[\![\textsc{let max}\ X = \phi\ \textsc{in}\ X]\!] = \bigcup\{S \| S \subseteq [\![\phi]\!]\} \subseteq \bigcup\{S \| S \subseteq [\![\psi]\!]\} = [\![\textsc{let max}\ X = \psi\ \textsc{in}\ X]\!]$$

Hence $\textsc{let max}\ X = \phi\ \textsc{in}\ X \ \Rightarrow \textsc{let max}\ X = \psi\ \textsc{in}\ X$.

□

*Proof*: According to [17], without loss of generality, we can consider dyadic closed formulas (contexts) and we can consider $\tilde{\varphi}$ as a formula of modal $\mu$-calculus, since the simultaneous recursion of our language is only a syntactic facilitation. For that reason, in order to find a safe decomposition of $\tilde{\varphi}$, we can refer to [29] in which a particular fragment of the modal $\mu$-calculus, without the $\vee$ operator and the minimum fixpoint operator and in which the box modality $[a,b]\phi$ is replaced by $(a,b)\phi = [a,b]\phi \wedge \langle a,b\rangle \mathbf{T}$, is fully decomposed as product of two unary formulas as follows:

| | | |
|---|---|---|
| $\models \mathbf{T}$ | $\Leftrightarrow$ | $\mathbf{T} \times \mathbf{T}$ |
| $\models \langle a,b\rangle\phi$ | $\Leftrightarrow$ | $\langle a\rangle\phi_1 \times \langle b\rangle\phi_2$ |
| $\models (a,b)\phi$ | $\Leftrightarrow$ | $(a)\phi_1 \times (b)\phi_2$ |
| $\models \phi \wedge \psi$ | $\Leftrightarrow$ | $\phi_1 \wedge \psi_1 \times \phi_2 \wedge \psi_2$ |
| $\models \textsc{let max}\ X = \phi\ \textsc{in}\ X$ | $\Leftrightarrow$ | $\textsc{let max}\ Y_1 = \phi_1 \textsc{in}\ Y_1 \times \textsc{let max}\ Y_2 = \phi_2 \textsc{in}\ Y_2$ |

where $\models \phi_1 \times \phi_2 \Leftrightarrow \phi$, $\models \psi_1 \times \psi_2 \Leftrightarrow \psi$ and $Y_1 \times Y_2$ is the change of variable related to $X$.

Our goal is finding a safe decomposition that is weaker than a fully one. Hence, we want to prove that, that at least a safe decomposition exists for the box modality, the $\vee$ operator and the minimun fixpoint operator.

Referring to Theorem 4.1, we know that finite formulas with box modality and/or $\vee$ operator have a decomposition. So the problem is when these operators are combined in a formula with recursion. Hence we proceed by analyzing each case separately.

$\phi = \textbf{let min}\ X = \psi\ \textbf{in}\ X$ : According to the definition of minimum fixpoint given in [3], we have that $(\textsc{let min}\ X = \psi\ \textsc{in}\ X) = \bigvee_{\alpha<k} \psi^\alpha(\mathbf{F})$ where $\alpha$ is an ordinal, $k$ is at worst the number of state of the system and $\psi^\alpha = \psi(\psi^{\alpha-1}(\mathbf{F}))$ where $\psi^0 = \mathbf{F}$. Hence, $(\textsc{let min}\ X = \psi^0\ \textsc{in}\ X) = \mathbf{F}$, $\textsc{let min}\ X = \psi^\alpha\ \textsc{in}\ X = \psi(\textsc{let min}\ X = \psi^{\alpha-1}\ \textsc{in}\ X) = \psi^\alpha(\mathbf{F})$. We can notice that if $\psi^1 = \mathbf{F}$ the minimum fixpoint is already found and it if $\mathbf{F}$. On the other hand, if $\psi^1 = \psi(\mathbf{F}) \neq \mathbf{F}$, we analyze how safely decompose $\psi(\mathbf{F})$ since $\psi(\mathbf{F}) \Rightarrow \bigvee_{\alpha<k} \psi^\alpha(\mathbf{F}) = \textsc{let min}\ X = \psi\ \textsc{in}\ X$. If it is a finite formula, we are able to decompose it by exploiting weak equivalences in Table 4 and then we conclude that:

$$\models \phi_1 \times \phi_2 \Leftrightarrow \psi(\mathbf{F}) \Rightarrow \bigvee_{\alpha<k} \psi^\alpha(\mathbf{F}) = \textsc{let min}\ X = \psi\ \textsc{in}\ X$$

If $\psi(\mathbf{F})$ is not a finite formula, it could be a minimum fixpoint so we proceed as above and, since the formula is well defined, the procedure ends. If $\psi(\mathbf{F})$ is a maximum fixpoint, we proceed as described below.

$\phi = \textbf{let max}\ X = \psi\ \textbf{in}\ X$ : We proceed by induction on the structure of the formula $\psi$.

- In the base case, $\psi = \mathbf{T}$, we have, according to [29]:

$$\models \textsc{let max}\ X = \mathbf{T}\ \textsc{in}\ X \Leftrightarrow \textsc{let max}\ Y_1 = \mathbf{T}\textsc{in}\ Y_1 \times \textsc{let max}\ Y_2 = \mathbf{T}\textsc{in}\ Y_2$$

- Let $\psi = \psi_1 \wedge \psi_2$ be the considered formula. For inductive hypothesis we have that $\psi_1$ and $\psi_2$ can be safely decomposed as $\models \psi_1' \times \psi_1'' \Rightarrow \psi_1$ and $\models \psi_2' \times \psi_2'' \Rightarrow \psi_2$. Hence, according to weak equivalences in Table 4:

$$\models (\psi_1' \wedge \psi_2') \times (\psi_1'' \wedge \psi_2'') \Leftrightarrow (\psi_1' \times \psi_1'') \wedge (\psi_2' \times \psi_2'') \Rightarrow \psi_1 \wedge \psi_2 = \psi$$

  and according to [29], having found a safe decomposition of $\psi$ we have:

$$\models \textsc{let max}\ Y_1 = \psi_1' \wedge \psi_2' \textsc{in}\ Y_1 \times \textsc{let max}\ Y_2 = \psi_1'' \wedge \psi_2'' \textsc{in}\ Y_2 \Rightarrow \textsc{let max}\ X = \psi\ \textsc{in}\ X$$

- Let $\psi = \psi_1 \vee \psi_2$ be the considered formula. Since we want to find a safe decomposition of $\psi$, we chose the first formulas of the disjunction, in this case $\psi_1$ and then, since $\psi_1 \Rightarrow \psi_1 \vee \psi_2$ and for inductive hypothesis we know that $\models \psi_1' \times \psi_1'' \Rightarrow \psi_1$, by exploiting the Proposition A.1 and referring to [29], we conclude that

$$\models \textsc{let max}\ X = \psi_1' \times \textsc{let max}\ X = \psi_1''\ \textsc{in}\ X \Rightarrow \textsc{let max}\ X = \psi\ \textsc{in}\ X$$

$$\Rightarrow \textsc{let max}\ X = \psi_1 \vee \psi_2\ \textsc{in}\ X$$

- Let $\psi = [a,b]\varphi$ be the considered formula. In order to find a safe decomposition, we refer to [29], by considering a strong version of box modality that implies the considered one. Indeed $(a,b)\varphi = [a,b]\varphi \wedge \langle a,b\rangle\varphi \Rightarrow [a,b]\varphi$. According [29] and since for inductive hypothesis, there exists a safe decomposition for $\varphi$ s.t. $\models \varphi_1 \times \varphi_2 \Rightarrow \varphi$, we have that $\models [a]\varphi_1 \wedge \langle a\rangle\mathbf{T} \times [b]\varphi_2 \wedge \langle b\rangle\mathbf{T} = (a)\varphi_1 \times (b)\varphi_2 \Rightarrow (a,b)\varphi \Rightarrow [a,b]\varphi$. Hence, by exploiting Proposition A.1:

$$\models \text{LET MAX } X_1 = [a]\varphi_1 \wedge \langle a\rangle\mathbf{T} \text{ IN } X_1 \times \text{LET MAX } X_2 = [b]\varphi_2 \wedge \langle b\rangle\mathbf{T} \text{ IN } X_2 \Rightarrow \\ \text{LET MAX } X = (a,b)\varphi \text{ IN } X \Rightarrow \text{LET MAX } X = [a,b]\varphi \text{ IN } X$$

  Hence

$$\models \text{LET MAX } X_1 = [a]\varphi_1 \wedge \langle a\rangle\mathbf{T} \text{ IN } X_1 \times \text{LET MAX } X_2 = [b]\varphi_2 \wedge \langle b\rangle\mathbf{T} \text{ IN } X_2$$

$$\Rightarrow \text{LET MAX } X = \psi \text{ IN } X$$

- Let $\psi = \langle a,b\rangle\varphi$ be the considered formula. For inductive hypothesis, there exists a safe decomposition for $\varphi$ s.t. $\models \varphi_1 \times \varphi_2 \Rightarrow \varphi$, hence, referring to [29]:

$$\models \langle a\rangle\varphi_1 \times \langle b\rangle\varphi_2 \Rightarrow \langle a,b\rangle\varphi = \psi$$

  By exploiting Proposition A.1, we conclude that:

$$\models \text{LET MAX } X_1 = \langle a\rangle\varphi_1 \text{ IN } X_1 \times \text{LET MAX } X_2 = \langle b\rangle\varphi_2 \text{ IN } X_2 \Rightarrow \text{LET MAX } X = \psi \text{ IN } X$$

- Let $\psi = \text{LET MIN } X = \varphi \text{ IN } X$ be the considered formula. According to the definition of minimum fixpoint given in [3], we have that $\text{LET MIN } X = \varphi \text{ IN } X = \bigvee_{\alpha<k} \varphi^\alpha(\mathbf{F})$ where $\alpha$ is an ordinal, $k$ is at worst the number of state of the system and $\varphi^\alpha = \varphi(\varphi^{\alpha-1}(\mathbf{F}))$ and $\varphi^0 = \mathbf{F}$. Hence, $\text{LET MIN } X = \varphi^0 \text{ IN } X = \mathbf{F}$, $\text{LET MIN } X = \varphi^\alpha \text{ IN } X = \phi(\text{LET MIN } X = \varphi^{\alpha-1} \text{ IN } X)$. Hence, there exists $\alpha'$ s.t. $\varphi^{\alpha'} \neq \mathbf{F}$. In particular, if $\varphi^1 = \mathbf{F}$ the minimum fixpoint is already found and it if $\mathbf{F}$. On the other hand, if $\varphi^1 = \varphi(\mathbf{F}) \neq \mathbf{F}$, it is a formula that, for inductive hypothesis we are able to safely decompose. Let $\phi_1$ and $\phi_2$ s.t. $\models \phi_1 \times \phi_2 \Rightarrow \varphi(\mathbf{F})$ then

$$\models \phi_1 \times \phi_2 \Rightarrow \varphi(\mathbf{F}) \Rightarrow \bigvee_{\alpha<k} \varphi^\alpha(\mathbf{F}) = \text{LET MIN } X = \varphi \text{ IN } X$$

□