Towards information flow properties for distributed systems [1]

Roberto Gorrieri [2]

Università di Bologna, Bologna, Italy

Fabio Martinelli[3]

Istituto di Informatica e Telematica - C.N.R., Pisa, Italy

Ilaria Matteucci[4]

Istituto di Informatica e Telematica - C.N.R., Pisa, Italy
CREATE-NET, Trento, Italy

Abstract

In this paper we present a framework for the specification of information flow properties for distributed systems. We consider partially specified distributed systems in which there are several unspecified components located in different places. As a case study, in this paper 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 adapt the NDC property to distributed systems by distinguishing between two different approaches. The first one we call *centralized* NDC, according to which 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. The second one is called *distributed* NDC, according to which there is one unspecified component for each distributed location, and the n unspecified components are completely independent, *i.e.*, they cannot coordinate or cooperate each other. 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: Indeed, we prove that *centralized* $BNDC$ is strictly finer than *distributed* $BNDC$, hence proving the quite expected fact that a system that can resist to coordinate attacks is also able to resist to simpler attacks performed by independent entities.

Keywords: Information flow properties, Nondeducibility, distributed systems, bisimulation, 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

[1] Work partially supported by CNR project “Trusted e-services for dynamic coalitions” and by EU-funded project “Software Engineering for Service-Oriented Overlay Computers”(SENSORIA) and by EU-funded project “CONSEQUENCE”.

[2] Email: gorrieri@cs.unibo.it

[3] Email: fabio.Martinelli@iit.cnr.it

[4] Email: ilaria.matteucci@iit.cnr.it

This paper is electronically published in
Electronic Notes in Theoretical Computer Science
URL: www.elsevier.nl/locate/entcs

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 [10] on deterministic machines: 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 γ, let γ' be its subsequence obtained by deleting all the actions of a user in G. G is *non interfering with* G' if and only if for every input sequence γ, the user in G' obtains the same outputs after the execution of γ and γ'.

This basic notion has been also adapted and generalized to the richer setting of nondeterministic and concurrent machines. Among the many definitions derived from NI, we consider the notion of *Bisimulation Non Deducibility on Compositions*, $BNDC$ for short, proposed in [4,6]: a system E is $BNDC$ if $E \setminus H$ (*i.e.*, E where all high level actions are prevented) is bisimulation equivalent to E in parallel with any high level process Π 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 Π, 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).

The goal of this work is to extend the idea of $BNDC$ also to distributed systems, where the possible intruders can be more than one and may also coordinate their effort. A distributed system is modelled as a *transducer* [11], *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 $C(X_1, \dots, X_n)$ can be seen as a *distributed partially specified system* [13,14,15,18], *i.e.*, a system C with *holes*, where the components $X_1 \dots X_n$ 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 $BNDC$, whatever the possible intruders are.

We proceed by following two different approaches to the problem of extending $BNDC$ to distributed partially specified systems. Given the context $C(X_1, \dots, X_n)$, where each X_i denotes a hole in the system, we may define the *distributed* $BNDC$ as the $BNDC$ property where the n intruders act independently, without communicating or coordinating their activities. A system satisfying distributed $BNDC$ should resist to distributed attacks conducted by n independent intruders. Then, we introduce the concept of *centralized* $BNDC$ in which the n intruders are centrally controlled and thus considered as a unique context which performs a vector of n actions, $\tilde{a} = (a_1, \dots, a_n)$. A system satisfying centralized $BNDC$ 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).

As expected, centralized $BNDC$ is proved to be finer than distributed $BNDC$. 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 distributed $BNDC$, but not centralized $BNDC$.

Interestingly enough, when bisimulation semantics is replaced by the less discriminating trace semantics, the results above are completely different. In particular, we prove the quite surprising result that centralized NDC (i.e., *trace-based Non-Deducibility on Composition*) is as discriminating as decentralized NDC. This reinforces the need of bisimulation semantics in the study of information flow properties.

This paper is organized as follows. Section 2 recalls some notions about context theory and introduces some idea behind the use of context for modelling distributed systems. Section 3 presents our framework for the specification of information flow properties in a distributed system. Section 4 shows a comparison with related work. Section 5 concludes the paper.

2 Background

In this section we recall some preliminary notions about contexts theory by referring to [11].

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 $\tilde{b}$ and change 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 2.2 CCS process algebra (see [17]) 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 $+, \| \in C_2^1$; *inactive* $\tilde{N}il \in C_n^m$ for any n and m. 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 CCS context is in Table 1.

■

Inaction:

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

Prefix:

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

Restriction:

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

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:

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

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

Contexts can be composed.

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 *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:

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

where $\tilde{a} = (a_1, \ldots, a_n)$ is a vector of actions.

We usually write $C(P)$ to denote a composed context; the combined process $C(P)$ is nothing but the composition $C \circ P$ as follows form the *context composition*.

Example 2.4 In order to explain how the composition between contexts works, we recall an example given in [11].

Let $a.b.Nil$ be a standard CCS term. It can be composed from the constructs as $a^* \circ$

$b^* \circ Nil$. Using the inference rule for composition, we can infer the following transitions:

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

and

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

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

■

Product of contexts

In order to represent a system with n holes, we use an 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 $m = \sum_i m_i$. This motivates the following construct of (independent) products of contexts.

Definition 2.5 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, s.t. 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 $\tilde{a}\tilde{c} = (a_1, \ldots, a_n, c_1, \ldots, c_r)$.

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 2.6 Also in this case, since in the rest of the paper composition and product operation are the most useful for our purposes, we recall an example already presented in [11].

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 rule for composition and product, we can infer the following transitions:

$$C\,\dfrac{\dfrac{-}{+\overset{a}{\underset{(a,0)}{\longrightarrow}}\Pi^1_2} \qquad \dfrac{C\,\dfrac{\dfrac{-}{a^*\overset{a}{\underset{0}{\to}}I_1} \quad \dfrac{-}{Nil\overset{0}{\longrightarrow}Nil}}{a^*\circ Nil\overset{a}{\longrightarrow}I_1\circ Nil} \qquad \dfrac{-}{Nil\overset{0}{\longrightarrow}Nil}}{a^*\circ Nil\times b^*\circ Nil\overset{(a,0)}{\longrightarrow}I_1\circ Nil\times b^*\circ Nil}\,P}{+\circ(a^*\circ Nil\times b^*\circ Nil)\overset{a}{\longrightarrow}\Pi^1_2\circ(I_1\circ Nil\times b^*\circ Nil)}$$

where the behavior of $\Pi^1_2 \circ (I_1 \circ Nil \times b^* \circ Nil)$ is behavioral equivalent to Nil since, in accordance with the standard CCS transition relation, $a.Nil + b.Nil \overset{a}{\longrightarrow} Nil$.

■

Feed-back on context

In order to deal with *recursion*, a construction of *feed-back* on contexts is defined, s.t. whenever $C \in C^n_n$ then $C^\dagger \in C^n_0$ and $C^\dagger$ is equivalent to $C \circ C^\dagger$. Formally, we have the following definition.

Definition 2.7 Let $\mathcal{C} = (\langle C^m_n \rangle_{n,m}, Act, \langle \rightarrow_{n,m} \rangle_{n,m})$ be a context system. A *feed-back* on $\mathcal{C}$ is a unary operation †, on contexts of $\mathcal{C}$ s.t., whenever $C \in C^{n,n}$ then $C' \in C^n_0$. Furthermore, the transduction for a context $C^\dagger$ with $C \in C^n_n$ is fully characterized by the rule:

$$\dfrac{C \overset{\bar{b}}{\underset{\bar{a}}{\to}} C' \qquad C^\dagger \overset{\bar{a}}{\longrightarrow} D}{C^\dagger \overset{\bar{b}}{\longrightarrow} C' \circ D}$$

In order to understand the meaning of this operator we recall the following example (see [11]).

Example 2.8 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:

$$F\,\dfrac{\dfrac{-}{a^*\overset{a}{\underset{0}{\to}}I_1} \qquad \dfrac{-}{(a^*)^\dagger\overset{0}{\longrightarrow}(a^*)^\dagger}}{(a^*)^\dagger \overset{a}{\longrightarrow} I_1 \circ (a^*)^\dagger}$$

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

■

2.1.2 *Open systems as transducers*

According to [11], 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, one yields 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, whether 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 2.9 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:

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

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

Informally, this context allows for the synchronization on actions h_1, h_2. This permits us, as we will see after, 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 context could be a binary contexts 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 2.3, the context $C \circ P$ behaves according to the following rule:

$$\frac{C \xrightarrow[(\bar{h_1},0)]{\tau} C' \quad P \xrightarrow{(\bar{h_1},0)} P'}{C(P) \xrightarrow{\tau} C'(P')} \tag{1}$$

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 the behavior of $C(P)$ has the same behavior of $\tau^* \circ l_1^* \circ \tau^* \circ \tau^* \circ l_2^* \circ Nil$. This is the only 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 $\backslash_{\{h_1,h_2,\bar{h_1},\bar{h_2}\}}$. Hence the first $\|$ takes the first action from the first component producing the transition showed in Equation 1. A similar reasoning is made for the second and third steps of the computation, so the context performs $l_1^* \circ \tau$. At this point the first component is reduced to Nil so we have a synchronization on the second component and then the action l_2 is unblocked.

Let us now consider the distributed case. Referring to how the contexts are built, we have the following first transition:

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

It is possible to note that the first step is similar. There is a difference between the behavior of $C(P)$ and $C(P_1, P_2)$ at the following steps of transduction. As a matter of fact, in the first case, we have to follow the behavior of P' and C'_1, on the contrary, in the second case the second step could be performed by P_2 and C_2. This crucial difference will be used in the following to prove a central result of this paper.

■

2.1.3 Behavioral equivalences

In order to have a method to compare behaviors of contexts, we recall some definition 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 $\tilde{\tau}$ is a tuple of actions in which there are no actions different from τ or 0. Let $\tilde{a} = (a_1, \dots, a_n)$ be a vector of actions in Act_0^n, then $\check{\tilde{a}} = \tilde{a}$ when $a_i \neq \tau$ for all $i = 1, \dots, n$, or $\check{\tilde{a}} = \tilde{a}[0 \backslash \tau]$ where all the occurrences of the τ 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 τ or 0 actions, the substitution is not performed.

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

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

Let $\check{\gamma} = \check{\tilde{a_1}} \dots \check{\tilde{a_n}}$ be a sequence of vectors of actions in which all the τ actions in each vector has been replaced by 0. $C \stackrel{\check{\gamma}}{\Longrightarrow} C'$ if $C \stackrel{\check{\tilde{a_1}}}{\Longrightarrow} \dots \stackrel{\check{\tilde{a_n}}}{\Longrightarrow} C'$

Let $C \in \mathcal{C}_0^n$ be a context, then the set of traces of C is $Tr(C) = \{\check{\gamma} | \exists C'\ C \stackrel{\check{\gamma}}{\Longrightarrow} C'\}$.

Definition 2.10 Let $C, D \in \mathcal{C}_0^n$ be two contexts. We define the relation of *trace inclusion*, denoted by $\leq_T$ as follows:

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

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* equivalence (see [17]) by distinguishing between a *strong* and a *weak* version.

Definition 2.11 Let $\mathcal{C} = (\langle \mathcal{C}_n^m \rangle_{n,m}, Act, \langle \rightarrow_{n,m} \rangle_{n,m})$ be a context system. Then an n-to-m *strong simulation* $\mathcal{R}$ is a binary relation on $\mathcal{C}_n^m$ s.t., whenever $(C, D) \in \mathcal{R}$ and $\tilde{a} \in Act_0^n$, $\tilde{b} \in Act_0^m$, then the following holds:

$$\text{if } C \xrightarrow[\tilde{a}]{\tilde{b}} C',\ \text{then } D \xrightarrow[\tilde{a}]{\tilde{b}} D' \text{ for some } D' \text{ with } (C', D') \in \mathcal{R}$$

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 $\mathcal{R}$ s.t. both $\mathcal{R}$ and $\mathcal{R}^{-1}$ are 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 w.r.t. contexts in $\mathcal{C}_0^n$. Hence we consider a definition that is an extension of the definition of weak bisimulation given by Milner in [17] for processes, that, as we have already said, can be seen as contexts in C_0^1.

Definition 2.12 Let $\mathcal{C} = (\langle C_0^n \rangle_{0,n}, Act, \langle \rightarrow_{0,n} \rangle_{0,n})$ be a context system. Then an 0-to-n *weak simulation* $\mathcal{R}$ is a binary relation on C_0^n s.t., whenever $(C, D) \in \mathcal{R}$ and $\tilde{a} \in Act_0^n$, then the following holds:

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

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 $\mathcal{R}$ s.t. both $\mathcal{R}$ and $\mathcal{R}^{-1}$ are simulations. We represent with $\approx$ the union of all the bisimulations.

Theorem 2.13 ([11]) *$\sim$ is preserved by composition, product and feed-back of contexts.*

2.2 The 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 is not trivial to find an access control policy which guarantees that no information leak is possible.

In the literature, there are many different security definition reminiscent of the information flow idea, each based on some system model (see [4,6,12,19]). The central property is the *Non Deducibility on composition* (*NDC*, see [4]): the low level users cannot infer the behavior of the high level user from the system because for the the low level users the system is always the same.

Exploiting the relation of bisimulation, we obtain the notion of *Bisimulation Non Deducibility on Compositions*, $BNDC$ for short, proposed in [4,6] as generalization of the classical idea of *Non-Interference* (see [10]) to nondeterministic systems: Low level users cannot infer the behavior of high level users from the system because for low level users the system is always the same.

In the following we recall the definition of non interference as NDC and $BNDC$ property proposed by Focardi and Gorrieri [3,4,5,6].

2.2.1 NDC and $BNDC$ property

To describe information flow property, 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 [3,4,5,6] Focardi and Gorrieri propose a family of information flow security properties called *Non Deducibility on Compositions* (NDC, for short). Intuitively, a system is NDC if, by *interacting* with every possible high level users, it always *appears* the same to low level users. 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. Firstly we consider the trace equivalence, thus the NDC is described in terms of contexts as follows:

$$E \text{ is } NDC \text{ iff } \forall\, \Pi \in \textit{ High} \text{ users }, \backslash_H(\|(E \times \Pi)) \approx_T \backslash_H(E) \text{ w.r.t. } \textit{Low} \text{ users}$$

where H is a set of high actions. NDC requires that high level processes Π are not able to change the low level behavior of the system represented by $\backslash_H(E)$. If it is equivalent to $\backslash_H(\|(E \times \Pi))$ this clearly means that Π is not able to modify in any way the execution of E. Since the operational semantics of contexts respects the operational semantics of process algebras, it is possible to give the definition of NDC by using contexts.

We can obtain a bisimulation based NDC by simply substituting $\approx_T$ with $\approx$. In particular Focardi and Gorrieri argue that $BNDC$ is the right choice (see [5,6]). Hence they give a formulation of $BNDC$ in terms of CCS parallel operator (see [17]). Also in this case we give the definition of $BNDC$ by exploiting the semantics of contexts as follows.

Definition 2.14 Let $Sort(\Pi)$ be the set of actions that occurs in Π 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 $\backslash_H(\|(E \times \Pi)) \approx \backslash_H(E)$.

3 Specification of Information flow property in a distributed system

In this paper we want to extend the definition of NDC and $BNDC$ properties given for processes in order to deal also with distributed systems. In particular, we consider a partially specified system in which several components are unspecified. We describe it as a context. Then we wonder if the context is NDC and/or $BNDC$.

To focus on which kind of scenarios we are going to investigate we present the following example.

Example 3.1 In order to make this example readable we use the infix notation that is more suitable than the prefix one of the context theory.

Let us consider a system specified as follows:

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

Let us interpret l_0 and l_1 as bit 0 and 1 respectively. According to how we chose X_0 and X_1 it is possible to generate sequences of 0 and 1 obtaining (possible infinite) sequence of numbers that represent information that (indirectly) flows from high to low since it depends on the behavior of high agents and can be detected by low ones.

3.1 Specification of NDC in distributed systems

According to the definition of NDC (see [3,4,5,6]), there is an universal quantification on all possible high users. Hence it is possible to specify the NDC property as an open system $S = \backslash_H(\|(E \times \_))$, where there is a hole in which we have to consider a high user. The system S has to satisfy the NDC property whatever the behavior of a high user is.

The scenario we want to analyze here consists in a system in which more than one high user acts on the system. A natural extension of this reasoning to a system in which there are more than one unspecified components, is considering these components as high users, *i.e.*, if $C \in C^1_n$ there are n high processes.

In this section we present how the specification of the NDC property can be extended to consider systems with more than one high level user. We consider two different approaches: The *centralized* approach, $CNDC$, and the *decentralized* one, $DNDC$.

Firstly we give the following notational definition.

Definition 3.2 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:

$$\backslash_{H^n} \overset{\tilde{a}}{\underset{\tilde{a}}{\rightarrow}} \backslash_{H^n} \quad \tilde{a} \notin H^n$$

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

Let us start with the centralized one.

Definition 3.3 Let $C \in C_n^m$ a generic context. $C \in CNDC$ iff

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

As before we have a universal quantification into the specification that results difficult to manage. Referring to the theory developed in [8], we want to give a static characterization of $CNDC$ by using a particular context $Top^n \in C_0^n$ 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-tuple of high actions.

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

Proposition 3.4 *Let $C \in C_n^m$ be a generic context.*

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

This means that the NDC property is statically characterized by 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 3.5 Let $C \in C_n^m$ 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 exploit the context Top^1 we prove (see the Appendix) the following result.

Proposition 3.6 *Let $C \in C_n^m$ a generic context.*

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

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 3.7 *Let $C \in C_n^m$ a generic context.*

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

Hence there is no difference between the two approaches if we consider the trace equivalence as behavioral relation between contexts in the analysis of the information flow properties.

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 open system with only one high user ([5,6]), also in presence of several high components, the $BNDC$ property turns out to be more interesting than NDC.

3.2 Specification of $BNDC$ in distributed systems

According to [7,13,14,15,18], the $BNDC$ property can be analyzed by using the *open system* paradigm (see [13,14,15,18]).

As we have already done in the previous section for the NDC property, we study a system in which more than one high user acts on the system by considering these components as high users, *i.e.*, if $C \in C_n^1$ there are n high processes.

Also in this case, as before, we distinguish between a centralized notion of $BNDC$ ($CBNDC$) and a decentralized one ($DBNDC$).

Firstly we specify the $CBNDC$, in which we consider the unspecified part of the system as a context that can perform a vector of actions.

Definition 3.8 Let $C \in \mathcal{C}_n^m$ be a context and let $\mathcal{C}_{0,H}^n$ be the set of n-ary High context. C is $CBNDC$ if and only if:

$$\forall X \in \mathcal{C}_{0,H}^n \; \backslash_{H^m}(C(X)) \approx \backslash_{H^m}(C(\tilde{Nil}))$$

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

Secondly, we take in account, a decentralized notion of $BNDC$, called $DBNDC$, in which we consider the unspecified part of the system as several independent contexts composed by product operation.

Definition 3.9 Let $C \in \mathcal{C}_n^m$ be a context and let $\mathcal{C}_{0,H}^1$ be the set of unary High context. C is $DBNDC$ if and only if:

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

where, according to the rule of the product operation, the product $X_1 \times \ldots \times X_n$ in a context in $\mathcal{C}_{0,H}^n$

In order to compare Definition 3.8 and Definition 3.9 we give the following result.

Proposition 3.10 *Let C be a context in $\mathcal{C}_n^m$. If C is $CBNDC$, then C is also $DBNDC$.*

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

□

However the vice-versa of the Proposition 3.10 does not hold as the following example shows.

Example 3.11 Let us consider now a context $D \in C_2^1$ s.t. $D = \tau^* \circ l_1^* \circ Nil + \tau^* \circ l_2^* \circ Nil + (\tau^* \circ l_1^* \circ Nil \| \tau^* \circ l_2^* \circ Nil) + C$ where C is the same context of the Example 2.9.

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 the Example 2.9, firstly we consider to combine D with another context $P = (\bar{h_1}, 0)^* \circ (\bar{h_1}, 0)^* \circ (0, \bar{h_2}) \in C_0^2$. 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 $\backslash_H(D(P_1', X_2)) \approx \backslash_H(D(Nil, Nil))$. Hence C is $DBNDC$ [5].

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 product of two unary context, *i.e.*, there are not any $P_1, P_2 \in C_0^1$ 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 $C_0^n \supset \underbrace{C_0^1 \times \ldots \times C_0^1}_{n}$.

To sum up, if we consider the trace equivalence as behavioral equivalence among contexts, we do not have any differences 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

[5] 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.

characterization of $BNDC$ in terms of a finite context (see [8]). Since the $BNDC$ property is a particular case both $CBNDC$ and $DBNDC$, we can conclude that there is no finite contexts playing the same role as Top^n for the trace case for statically characterize neither $CBNDC$ or $DBNDC$.

4 Related work

In [9] 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 w.r.t. 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. Also in our work we show a method to define the information flow property for distributed system. In our specification there are several high user that can also interact one to each other. This is not the case of M_BNDC in which a single process passes from a location to another by performing a set of action in each location, sequentially.

In [1] the authors have studied which could be the effect of the environment on the information flow security in a multilevel system by referring to the context theory. Indeed, they introduce the notion of *secure contexts for a class of processes* as a parametric notion w.r.t. both the observation equivalence and the operation to characterize the low level views of a process. Our work can be consider an extension of this one in which we consider more then one high user.

In [20] the authors pointed out that in general $BNDC$ is not compositional w.r.t. all the process algebras operators. This results is an extension of the one given in [15], in which there is the proof that $BNDC$ is not compositional w.r.t. parallel operators. Moreover, in [20], the author has defined several properties stronger of $BNDC$ in order to have properties that are compositional w.r.t. process algebras operators. In this work we have proposed a different specification of the problem that takes in account the environment of the system.

$BNDC$ has been also studied in the distributed model of elementary net systems in [2], where however the idea is to consider one unique intruder, in the same line of centralized $BNDC$.

5 Conclusion and future work

In this paper we presented a possible extension of the specification of information flow properties also to a framework in which more than one high level user is active. Then we have extended the approach, based on the open system paradigm used to specify information flow properties such as $BNDC$ with one high level user, to deal with such a more complex scenario.

We aim to extend this research by considering also the verification problem. We are working on a method to verify both centralized and distributed $BNDC$. Moreover we will also investigate what could happen if 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 [16] to define and synthesize controller operators also for the properties defined in this work.

References

[1] Bossi, A., D. Macedonio, C. Piazza and S. Rossi, *Information flow in secure contexts*, J. Comput. Secur. **13** (2005), pp. 391–422.

[2] Busi, N. and R. Gorrieri, *Positive non-interference in elementary and trace nets*, in: J. Cortadella and W. Reisig, editors, *ICATPN*, Lecture Notes in Computer Science **3099** (2004), pp. 1–16.

[3] Focardi, R., "Analysis and Automatic Detection of Information Flows in Systems and Networks," Ph.D. thesis, Department of Computer Science, University of Bologna (1998).

[4] Focardi, R. and R. Gorrieri, *A classification of security properties for process algebras*, Journal of Computer Security **3** (1994/1995), pp. 5–33.

[5] Focardi, R. and R. Gorrieri, *The compositional security checker: A tool for the verification of information flow security properties*, IEEE Transactions on Software Engineering **27** (1997), pp. 550–571.

[6] Focardi, R. and R. Gorrieri, *Classification of security properties (part I: Information flow)*, in: *FOSAD '00: Revised versions of lectures given during the IFIP WG 1.7 International School on Foundations of Security Analysis and Design on Foundations of Security Analysis and Design* (2001), pp. 331–396.

[7] Focardi, R., R. Gorrieri and F. Martinelli, *Classification of security properties - part ii: Network security*, in: R. Focardi and R. Gorrieri, editors, *FOSAD*, Lecture Notes in Computer Science **2946** (2002), pp. 139–185.

[8] Focardi, R. and F. Martinelli, *A uniform approach for the definition of security properties*, in: *FM '99: Proceedings of the Wold Congress on Formal Methods in the Development of Computing Systems-Volume I* (1999), pp. 794–813.

[9] Focardi, R. and S. Rossi, *Information flow security in dynamic contexts*, in: *CSFW '02: Proceedings of the 15th IEEE workshop on Computer Security Foundations* (2002), p. 307.

[10] Goguen, J. A. and J. Meseguer, *Security policy and security models*, in: *Proc. of the 1982 Symposium on Security and Privacy* (1982), pp. 11–20.

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

[12] Lowe, G., *Semantic models for information flow*, Theor. Comput. Sci. **315** (2004), pp. 209–256.

[13] Martinelli, F., "Formal Methods for the Analysis of Open Systems with Applications to Security Properties," Ph.D. thesis, University of Siena (1998).

[14] Martinelli, F., *Partial model checking and theorem proving for ensuring security properties*, in: *CSFW '98: Proceedings of the 11th IEEE workshop on Computer Security Foundations* (1998), p. 44.

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

[16] Martinelli, F. and I. Matteucci, *Synthesis of local controller programs for enforcing global security properties*, in: *Proc. First International Workshop on Advances in Policy Enforcement (APE'08)* (2008), to appear.

[17] Milner, R., "Communicating and mobile systems: the π-calculus," Cambridge University Press, 1999.

[18] R.Focardi, R.Gorrieri and F.Martinelli, *Real-time Information Flow Analysis*, IEEE JSAC (2003).

[19] Ryan, P. Y. A. and S. A. Schneider, *Process algebra and non-interference*, in: *CSFW '99: Proceedings of the 1999 IEEE Computer Security Foundations Workshop* (1999), p. 214.

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

A Technical proofs

Before the proof of Proposition 3.4 and 3.6 we give the following lemmata.

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

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

$$\mathcal{S} = \{(C, Top^n) | C \in \mathcal{C}^n_0\}$$

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.

□

Lemma A.2 *Let $C, D \in \mathcal{C}^n_0$ 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' s.t. $D \xrightarrow{\tilde{a}} D'$ 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 $C \approx_T D$.

□

Proposition 3.4 Let $C \in \mathcal{C}^m_n$ a generic context.

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

*Proof:*The proof is divided into two parts:

$\Rightarrow$ Since $\forall X \in \mathcal{C}^n_{0,H}\ \backslash_{H^m} C(X) \approx_T \backslash_{H^m} C(\tilde{Nil})$, 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 3.6 Let $C \in C_n^m$ be a generic context.

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

Proof: 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 3.7 Let $C \in C_n^m$ be a generic context.

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

Proof: 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 s.t. $\underbrace{Top^1 \times \ldots \times Top^1}_{n} \xrightarrow{\tilde{a}} C$ 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 $Top^{n\prime}$ s.t. $Top^n \xrightarrow{\tilde{a}} Top^{n\prime}$ and $(D, Top^{n\prime}) \in \mathcal{R}$. 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 2.13), we have that $\backslash_{H^m} C(Top^n) \sim C(Top^1, \ldots Top^1)$. For Lemma A.2, $\backslash_{H^m} C(Top^n) \approx_T C(Top^1, \ldots Top^1)$. Since $C(\tilde{Nil}) \approx_T \backslash_{H^m} C(Nil, \ldots Nil)$ trivially, we have the thesis.

□