UNIVERSITE D'ORLEANS
Faculté des Sciences
LIFO

Laboratoire d'Informatique Fondamentale d'Orléans
4, rue Léonard de Vinci, BP 6759
F-45067 Orléans Cedex 2
FRANCE

Rapport de Recherche
www : http://www.univ-orleans.fr/SCIENCES/LIFO/

A Synchronous Bisimulation-Based Approach for Information Flow Analysis

S. ANANTHARAMAN, G. HAINS
Université d'Orléans, LIFO

Rapport N^o **2003-01**

A Synchronous Bisimulation Based Approach for Information Flow Analysis

Siva Anantharaman and Gaétan Hains *

Abstract

A process algebra is defined where parallel composition is structured around *synchronous* communication. Its essential difference with CCS is the hypothesis that internal actions must be observable for the clock; consequently, in our formalism (strong) bisimulation will be the basis for information flow analysis, instead of equivalences based on trace or weak bisimulation. Bisimulation reduces in our formalism to equality modulo an ACUID equational theory extended with prefixes. We show that information flow analysis based on our formalism is finer than analyses based on trace or weak bisimulation.

1 Introduction

In the context of information flow analysis, an important issue is that of ensuring the security of distributed systems. The notions of non-interference and of admissible interference have been proposed with such an objective. In many of the known works (e.g. [11], [12], [3]) they are verified with the help of type systems on the process variables; however, such type systems can be too restrictive since they tend to categorize too many programs as unsafe. Other approaches based on process algebras have also been proposed (e.g. [6], [7]), where the equivalence with respect to weak trace or weak bisimulation is the key notion above which non-interference is defined. In these approaches, the so-called internal CCS action τ has kept its 'magic' property of being unobservable even for the clock. Undoubtedly this is unsatisfactory since the 'divergence' of a process (in the terminology of [6]) where an internal action τ may loop for too long will then go unobservable by definition.

In this work we propose an approach which is different: our ultimate aim is an exact verification algorithm for non-interference (and subsequently also for admissible interference as defined in [10]). We design the process algebra around an ACUID equational theory extended with prefixes which symbolize actions, and by making parallel synchronous composition distributive over non-deterministic choice; such a synchronous composition will be commutative and non-associative. Bisimulation between processes

*LIFO, CNRS - Université d'Orléans, France, {siva,ghains}lifo.univ-orleans.fr

is then interpretable as congruence over this equational theory. We then establish that information flow analysis is strictly finer when it is based on such a bisimulation, than when it is based on trace or weak bisimulation equivalence.

This paper is organized as follows. In section 2 we define our process algebra in a way very much similar to CCS, and without considerations of secrecy levels on the action symbols. There are two essential differences with a CCS-like approach: i) in defining the parallel composition of actions, we focus explicitly on the synchronous side; and it is shown how the notion of asynchronous parallel composition – although not formally part of the setup – can be defined in complete conformity with CCS, in our formalism; ii) we replace the internal action τ of CCS – meant as unobservable (including *even as clock*) in all standard works on process analysis – by an action denoted θ, which will be 'internal' *except* for the clock. One of our objectives in doing so is to eliminate the observational problems caused by the divergence of a process, where for instance an internal action τ à la CCS may loop for too long.

In section 3, we define the usual notion of bisimulation between processes, in our formalism, and derive its algebraic properties. We also establish an equational characterization of bisimulation as a congruence defined by an equational theory. In section 4, the action symbols are labeled with levels of secrecy, and a notion of non-interference between processes called RNI ('robust non interference') is developed. RNI is a generalization of the notions 'non-deterministic non interference' based on weak trace and weak bisimulation, respectively called $SNNI$ and $BNNI$ in [6]. In section 5, we propose a notion of *information tightness* for any process P with respect to any *intruder* process I, based on strong bisimulation. This notion covers more than the confidentiality of cryptographic protocols; it is also capable of expressing the parasite behaviors caused by the presence of an intruder. Several examples are given to illustrate the finer power of information flow analysis based on our formalism.

2 A Synchronous Process Algebra

In this section we construct a restricted form of process algebra over a given set $\mathtt{Act}$ of action symbols; attributing levels of secrecy to action symbols will be our concern in a later section. Let

$$\mathtt{Act} = \{\theta\} \biguplus \{a, \bar{a}, b, \bar{b}, \ldots\} \tag{1}$$

where θ is a special 'internalizing' action symbol, similar to the τ in CCS; but its role in information flow will be different: it will 'count' a unit of time, meant to be observable; 'bars' serve in pairing out synchronizing (or communicating) actions, in particular $\bar{\bar{x}} = x$; we refer to $\bar{x}$ as the conjugate of x; the special action θ is assumed conjugate to itself. We will use variables $x, y, \ldots \in \mathtt{Act}$, $\alpha, \beta, \ldots \in \mathtt{Act}^*$ and $P, Q, R, \ldots \in \mathtt{Proc}$.

The grammar of ground process terms is

$$\mathtt{Proc} ::= 0 \mid \mathtt{Act.Proc} \mid \mathtt{Proc} + \mathtt{Proc} \mid \mathtt{Proc} * \mathtt{Proc} \mid P \setminus x; \tag{2}$$

its operations are respectively called null process, prefix, (non-deterministic) choice, (parallel) synchronous composition, and restriction. Process $x.0$ will be abbreviated to x and $x.(y.0)$ to xy where no confusion is likely between processes and their traces.

The synchronization of actions is defined as a partial binary operation denoted $*$ in analogy with synchronous process composition:

$$x * y = \begin{cases} \theta & \text{if } x = \bar{y} \\ x & \text{if } y = \theta \\ y & \text{if } x = \theta \\ undef & \text{otherwise} \end{cases}$$

The difference between CCS's τ and our θ must be already perceptible: CCS's τ would not have synchronized with any non-internal action, nor with itself. The operational semantics of our processes is a labeled transition system, corresponding to a relation on $\texttt{Proc} \times \texttt{Act} \times \texttt{Proc}$, that we shall denote as $P \xrightarrow{x} Q$ and define by the following rules:

(`Prefix`) $$\frac{}{x.P \xrightarrow{x} P}$$

(`Sum`) $$\frac{P \xrightarrow{x} P'}{P + Q \xrightarrow{x} P'} \qquad \frac{Q \xrightarrow{x} Q'}{P + Q \xrightarrow{x} Q'}$$

(`Sync`) $$\frac{P \xrightarrow{x} P' \quad Q \xrightarrow{y} Q' \quad x * y = z}{P * Q \xrightarrow{z} P' * Q'}$$

(`Restr`) $$\frac{P \xrightarrow{x} P' \quad x \neq y}{P \setminus y \xrightarrow{x} P' \setminus y}$$

For the purposes of the current work we shall assume that if a process is defined recursivley, then the recursion is bounded. The reason is that our aim is to propose techniques for verification, where every action - including our 'internal' θ - consumes a visible unit of time, and we want the verification done in bounded time. Our processes will therefore be of bounded size. [1]

Definition 1 *For any process term P the set of possible actions for P is defined as the set:*

$$\texttt{Act}(P) = \{x \in \texttt{Act} \mid \exists P'.\ P \xrightarrow{x} P'\}$$

and the set of its direct successor processes as:

$$\texttt{Next}(P) = \{P' \in \texttt{Proc} \mid \exists x.\ P \xrightarrow{x} P'\}.$$

[1]However this assumption does not preclude an extension of our formalism to context-free process grammars, for which decidability results are known [8].

Given any finite process term P, it is easily checked that $\texttt{Act}(P)$ and $\texttt{Next}(P)$ are finite sets of process terms.

Remarks 1. 1) Rule ($\texttt{Sync}$) is symmetric in $x, \bar{x}$ because $\bar{\bar{x}} = x$.

2) The semantics behind rule ($\texttt{Sync}$) is that θ is meant as a passive clock step to be absorbed by non-empty events in a synchronous composition. A process emitting θ is "counting" non-θ events in its parallel sibling; in particular $\theta * \theta = \theta$, hence two 'counting' processes may synchronize.

3) Transition labels can be generalized to non-empty strings of actions by a transitive rule so that $P \xrightarrow{ab} Q$ means $P \xrightarrow{a} R$ and $R \xrightarrow{b} Q$ etc:

$$(\texttt{Trans}) \qquad \frac{P \xrightarrow{x} P' \qquad P' \xrightarrow{\alpha} P''}{P \xrightarrow{x\alpha} P''}$$

4) We do not have (as yet) a rule à la CCS for asynchronous parallel composition in our algebra; thus, we cannot say for instance $a.P * b.Q \xrightarrow{a} P * b.Q$. This is why the above process algebra is referred to as *synchronous.* However, it *is* possible to introduce formally – up to bisimulation - such a notion of asynchronous parallel composition in our approach; see Remark 3 below.

The hypothesis of bounded recursion in process terms makes our operational semantics noetherian :

Lemma 1 *If $P \xrightarrow{x} P'$ then process term P' is strictly smaller, i.e. has fewer action symbols, than P.*

Proof: By induction on the derivation of $P \xrightarrow{x} P'$. The base for this induction is the ($\texttt{Prefix}$) axiom for the operational semantics: if $P = x.Q \xrightarrow{x} Q = P'$ then P' has fewer action symbols than P. The induction hypothesis is that the lemma's statement is true for all transitions appearing as hypotheses in ($\texttt{Sum}$),($\texttt{Sync}$) or ($\texttt{Restr}$). There is one induction step for each of the above transition rules:
($\texttt{Sum}$): P' is smaller than P by hypothesis and hence smaller than $P + Q$.
($\texttt{Sync}$): P' is smaller than P and Q' is smaller than Q by hypothesis, hence $P' + Q'$ is smaller than $P + Q$.
($\texttt{Restr}$): P' is smaller than P by hypothesis, hence $P' \setminus y$ is smaller than $P \setminus y$. □

3 Bisimulation

Definition 2 *Process P is said to be* simulated *by Q or $P \leq Q$ when $P \xrightarrow{x} P'$ implies the existence of a process Q' such that $Q \xrightarrow{x} Q'$ and $P' \leq Q'$. Processes P and Q are* bisimilar*, or $P \approx Q$ if $P \leq Q$ and $Q \leq P$.*

Lemma 2 *Simulation is a preorder relation.*

Proof: Reflexivity is trivial. Transitivity is verified as follows. Assume $P \leq Q$, $Q \leq R$ and $P \xrightarrow{x} P'$. The first simulation implies the existence of Q' such that $Q \xrightarrow{x} Q'$ and $P' \leq Q'$. The second simulation thus implies the existence of R' such that $R \xrightarrow{x} R'$ and $Q' \leq R'$. The lemma implies that P', Q', R' are respectively smaller than P, Q, R. By induction hypothesis on the sizes of terms we may therefore assume that transitivity holds for P', Q', R' and conclude $P' \leq R'$ from $P' \leq Q'$ and $Q' \leq R'$. The definition of $P \leq R$ is thus satisfied. □

Since simulation is a preorder, bisimulation is an equivalence relation by construction: moreover it is a congruence for process terms:

Proposition 1 *Bisimulation is a congruence for the process operators, in the following sense: if $X \approx X'$ and $Y \approx Y'$,and x is any action symbol, then:*

$$0 \approx 0, \quad x.X \approx x.X', \quad X + Y \approx X' + Y', \quad X * Y \approx X' * Y', \quad \textit{and} \quad X \setminus x \approx X' \setminus x.$$

Proof:

1. The only transitions from $x.X$ and $x.X'$ are respectively $\xrightarrow{x} X$, and $x.X' \xrightarrow{x} X'$; by hypothesis $X \leq X'$, hence $x.X \leq x.X'$. The inverse simulation is proved symmetrically.

2. The only transition from $X + Y$ is (by symmetry) of the form $X + Y \xrightarrow{x} Z$ when $X \xrightarrow{x} Z$. But then the hypothesis $X \leq X'$ implies the existence of a transition $X' \xrightarrow{x} Z'$ such that $Z \leq Z'$. Rule (`Sum`) therefore implies that $X' + Y \xrightarrow{x} Z'$ and the definition of $X + Y \leq X' + Y$ is satisfied. We can then prove $X' + Y \leq X' + Y'$ similarly and conclude by transitivity of simulation that $X' + Y'$ simulates $X + Y$. We conclude similarly that $X + Y$ simulates $X' + Y'$, hence $X + Y \approx X' + Y'$.

3. Transitions possible from $X * Y$ are of the form $X * Y \xrightarrow{a} Z * T$ when $X \xrightarrow{x} Z$, $Y \xrightarrow{y} T$ and $x * y = a$. Simulation $X \leq X'$ implies that $X' \xrightarrow{x} Z'$ with $Z \leq Z'$; simulation $Y \leq Y'$ implies that $Y' \xrightarrow{y} T'$ with $T \leq T'$. The (`Sync`) rule then implies that $X' * Y' \xrightarrow{a} Z' * T'$, and since Z, T, Z', T' are respectively smaller than X, Y, X', Y' (by Lemma 1), we may apply an induction hypothesis on the size of terms and conclude from $Z \leq Z'$, $T \leq T'$ that $Z * T \leq Z' * T'$. We have thus satisfied the definition of $X * Y \leq X' * Y'$. A symmetric proof would conclude that $X * Y \geq X' * Y'$ and hence $X * Y \approx X' * Y'$.

4. Transitions from $X \setminus x$ must be of the form $X \setminus x \xrightarrow{y} Z \setminus x$ with $X \xrightarrow{y} Z$ and $x \neq y$. Any such transition would imply by hypothesis $X \leq X'$ that $X' \xrightarrow{y} Z'$, $Z \leq Z'$ and $x \neq y$. Rule (`Restr`) then yields $X' \setminus x \xrightarrow{y} Z' \setminus x$, while $Z \leq Z'$ and $x \neq y$. We can then apply induction on the size of terms (Z is smaller than X and Z' is smaller than X') to conclude that $Z \leq Z'$ implies $Z \setminus x \leq Z' \setminus x$. We have then inferred that whenever $X \setminus x \xrightarrow{y} Z \setminus x$, we have $X' \setminus x \xrightarrow{y} Z' \setminus x$ and $Z \setminus x \leq Z' \setminus x$, i.e. to say $X \setminus x \leq X' \setminus x$. The inverse simulation is proved symmetrically. □

3.1 Bisimulation as an Equational Congruence

Let $\equiv$ be the binary relation on `Proc` obtained by making $+$ an associative, commutative and idempotent operator, and adding an operator '$*$' 2-sided distributive over '$+$', and an element 0 neutral for '$+$' and absorbent for '$*$' (we thus get an ACUID theory in the rewriting terminology). We also assume $*$ to be commutative, and that it annihilates synchronous compositions for which the actions do not synchronize; and finally that restriction is commutative and idempotent on its right (action) argument. In other words, $P \equiv Q$ iff $P = Q$ or P can be obtained from Q by applying one or more of the following equational axioms. (We shall see later, cf. Remarks 7.1 and 7.2 in the Appendix, that not all of these axioms are really needed for an equational characterization of bisimulation.)

(A) $X + (Y + Z) \doteq (X + Y) + Z$	(C) $X + Y \doteq Y + X$
(I) $X + X \doteq X$	(D) $X * (Y + Z) \doteq X * Y + X * Z$
(U) $X + 0 \doteq X$	(C*) $X * Y \doteq Y * X$
(Z) $X * 0 \doteq 0.$	(P) $a.X * b.Y \doteq (a * b).(X * Y)$; is 0 if $(a * b)$ undef
(C$\,\backslash\,$) $P \setminus x \setminus y \doteq P \setminus y \setminus x$	(I$\,\backslash\,$) $P \setminus x \setminus x \doteq P \setminus x$
(D$\,\backslash\,$) $(P + Q) \setminus x \doteq P \setminus x + Q \setminus x.$	(P$\,\backslash\,$) $(a.P) \setminus x \doteq a.(P \setminus x)$ if $x \neq a; 0$ if $x = a$.
(Z$\,\backslash\,$) $0 \setminus x \doteq 0$	

We propose to prove that the congruence $\equiv$ defined by the above equational theory on process terms is exactly the same as that defined by bisimulation.

We begin by showing that bisimulation satisfies the equational axioms for $\equiv$; i.e. to say, the above equational axioms are sound for axioms for $\equiv$; i.e. to say, the above equational axioms are sound for bisimulation. We shall be needing some of the properties listed below (where $P, Q, R, ..$ denote arbitrary processes, and $ab, b, , ., x, y, ..$ are arbitrary actions). They are all easy consequences of our definitions; proofs are given in the Appendix.

(1) The choice operator '$+$' is associative, commutative, and idempotent, up to bisimulation, and admits a unit element: For any process terms P, Q, R, we have
$P + (Q + R) \approx (P + Q) + R, \quad P + Q \approx Q + P, \quad P + P \approx P$
$P + 0 \approx P, \quad P * 0 \approx 0, \quad P * Q \approx Q * P.$

(2) Prefix and (parallel) synchronous composition commute: $a.P*b.Q \approx (a*b).(P*Q)$

(3) Synchronous composition '$*$' distributes over the choice operator '$+$' up to bisimulation: $P * (Q + R) \approx P * Q + P * R.$

(4) Double restriction is commutative and idempotent on its action arguments, up to bisimulation: $P \setminus x \setminus y \approx P \setminus y \setminus x, \quad P \setminus x \setminus x \approx P \setminus x.$

(5) Restriction distributes over choice, up to bisimulation: $(P+Q) \setminus x \approx P \setminus x + Q \setminus x.$

(6) $0 \setminus a \approx 0$. Restriction distributes conditionally over prefix, up to bisimulation:

$$(a.P) \setminus x \approx \begin{cases} a.(P \setminus x) & \text{if} \quad x \neq a \\ 0 & \text{if} \quad x = a \end{cases} .$$

Remark 2. (Parallel) Synchronous composition is *not associative*: in general we do not have $P * (Q * R) \not\lesssim (P * Q) * R$. Here is an example: Let $P = a$, $Q = \bar{a}$ and $R = b$, with $b \notin \{a, \theta\}$. Then $P*(Q*R) = a*(\bar{a}*b)$ has no possible transition because $Q*R$ has none. But on the other hand $(P * Q) * R = (a * \bar{a}) * b \xrightarrow{\theta * b} 0 * 0$, because $(P * Q) = (a * \bar{a}) \xrightarrow{\theta} 0$ and $R = b \xrightarrow{b} 0$. □

We show now that the equivalence $\equiv$ is correct w.r.t bisimulation. We will denote by $P[R]$ a process term P having R as subterm; in this context $P[R']$ represents term P where (one occurrence of) R has been replaced by the process term R'.

Proposition 2 *$P \equiv Q$ implies $P \approx Q$*

Proof: Suppose $R \equiv R'$ be an instance of any one of the axioms. Then, one of the above propositions shows that $R \approx R'$ and since bisimulation is a congruence for process terms, this equivalence can be propagated to $P[R] \approx P[R']$. Any equational proof of $P \equiv Q$ is a finite sequence of such transformations. As all of them preserve bisimulation, transitivity of bisimulation implies $P \approx Q$. □

To prove the reverse implication, we need a couple of preliminary results. We begin by deducing an *additive (normal) form* for process terms, which is uniquely determined up to bisimulation.

Proposition 3 *Any process term P is bisimilar to a (uniquely determined) process term of the form: $a_0.P_0 + \ldots + a_{n-1}.P_{n-1}$ where $\{a_0, \ldots, a_{n-1}\} = \mathtt{Act}(P)$.*

Proof: The proof is by induction on the size of P. The base case is $P = 0$ where the result is trivial since $\mathtt{Act}(0) = \emptyset$. The steps in the general case are as follows:

- If $P = a.Q$ then $\mathtt{Act}(P) = \{a\}$ and the form $a.Q$ satisfies the statement of the Proposition.
- Assume $P = Q + R$ and by induction hypothesis $Q \approx \sum q_i.Q_i$, $R \approx \sum r_j.R_j$. Then by associativity and commutativity we have $P \approx \sum_{i,j}(q_i.Q_i + r_j.R_j)$. Moreover rule ($\mathtt{Sum}$) implies $\mathtt{Act}(P) = \mathtt{Act}(Q) \cup \mathtt{Act}(R)$ which, by the hypotheses on $\mathtt{Act}(Q)$ and $\mathtt{Act}(R)$ means $\mathtt{Act}(P) = \{q_i, r_j\}_{i,j}$.
- $P = Q * R$ and by induction hypothesis $Q \approx \sum q_i.Q_i$, $R \approx \sum r_i.R_i$. Repeated application of distributivity yields $P \approx \sum_{i,j}(q_i.Q_i) * (r_j.R_j)$ which, by rules ($\mathtt{Sum}$) and ($\mathtt{Sync}$) is bisimilar to $P \approx \sum_{z=q_i*r_j} z.(Q_i * R_j)$: those terms for which $q_i * r_j$ is undefined are bisimilar to 0 and thus absorbed into the sum. Comparing this set of prefixes with the form $Q * R$ for P we conclude by rule ($\mathtt{Sync}$) that $\mathtt{Act}(P) = \{z = q_i * r_j \mid q_i \in \mathtt{Act}(Q), r_j \in \mathtt{Act}(R)\}$.
- $P = Q \setminus x$ and by induction hypothesis $Q \approx \sum q_i.Q_i$. From rule ($\mathtt{Restr}$) we know that $\mathtt{Act}(P) = \mathtt{Act}(Q) - \{x\} = \{q_i \mid q_i \neq x\}$. We also have $(\sum q_i.Q_i) \setminus x \approx \sum(q_i.Q_i) \setminus x \approx \sum_{q_i \neq x} q_i.(Q_i \setminus x)$; this proves the statement of the proposition. □

Lemma 3 *$P \approx 0$ implies $P \equiv 0$.*

Proof: Since P is bisimilar to 0, it follows that $\mathtt{Act}(P) = \emptyset$. We will reason by induction on the size of terms, by cases on the syntax of P and use several times the lemma showing that any term is bisimilar to its additive normal form.

If $P = 0$ there is nothing to prove. Note that P may not be of the form $a.P'$ because $\mathtt{Act}(P) = \emptyset$.

If $P = P' + P''$ then we must have $\mathtt{Act}(P') = \mathtt{Act}(P'') = \emptyset$, otherwise there would be a transition for P, and since P', P'' are smaller terms than P we may use the induction hypothesis to conclude that $P' \equiv 0$, $P'' \equiv 0$ and hence $P = P' + P'' \equiv 0$.

If $P = P' * P''$, let the additive normal form for P' and P'' be respectively $\sum_i a_i.P'_i$, $\sum_j b_j.P''_j$. By induction hypothesis, P' and P'' are also equationally equivalent to those sums and distributivity implies $P = P' * P'' \equiv \sum_{i,j}(a_i.P'_i) * (b_j.P''_j)$. Moreover we know that every product $a_i * b_j$ is undefined because otherwise the rule for transitions of a product would have implied a transition for $P = P' * P'$. Hence the axioms imply that $\forall i, j.(a_i.P'_i) * (b_j.P''_j) \equiv 0$ and thus $P = P' * P'' \equiv \sum_{i,j} 0 \equiv 0$.

If $P = P' \setminus a$ then either $P' \approx 0$ or not. In the first case, by induction we have $P' \equiv 0$ so $P \equiv 0 \setminus a \equiv 0$. Otherwise there exists a $b \in \mathtt{Act}(P')$ and, since $P' \setminus a$ has no possible action, the only such b must be a and hence $P' \approx \sum_i a.P'_i$. Since P' is smaller than P we may apply the induction hypothesis to yield $P' \equiv \sum_i a.P'_i$. From this it follows that $P = P' \setminus a \equiv (\sum_i a.P'_i) \setminus a \equiv \sum_i (a.P'_i \setminus a) \equiv \sum_i 0 \equiv 0$. □

Proposition 4 *$P \approx Q$ implies $P \equiv Q$.*

Proof: If $P \approx 0$ then $P \equiv 0$ by the previous lemma and similarly for $Q \equiv 0 \equiv P$. So let us assume that $\mathtt{Act}(P)$ and $\mathtt{Act}(Q)$ are non-empty.

It is easy to verify that $\mathtt{Act}(P) = \mathtt{Act}(Q)$; let A denote this common set of actions. Then we know that $P \approx \sum_{a \in A} P_a$ and $Q \approx \sum_{a \in A} Q_a$. Bisimulation implies that for every $a \in A$ we have $P_a \approx Q_a$ and by induction hypothesis (the terms P_a, Q_a are smaller than P, Q respectively) $P_a \equiv Q_a$. One deduces then, by commutativity and associativity of $+$, that we have $P \equiv Q$. □

Remarks 3. Though parallel asynchronous composition (denoted usually '|' in CCS) has not been formally defined in our formalism, it can be 'simulated' with the help of the additive normal forms, as follows: Let P, Q be any process terms and $P = \sum_{a_i} a_i.P_i$ and $Q = \sum_{b_j} b_j.Q_j$ be their additive normal forms. Then their asynchronous (parallel) composition is the process term (recursively) defined uniquely up to bisimulation, as: $P \mid Q = \displaystyle\sum_{a_i} a_i.(P_i \mid Q) \; + \; \sum_{b_j} b_j.(P \mid Q_j) \; + \sum_{a_i = \overline{b_j}} \theta.(P_i \mid Q_j)$. We set as the base case: $0 \mid P = P = P \mid 0$, for this definition of '|' to be effective.

We can define along similar lines the parallel asynchronous composition of any finite family of processes $P_1, \ldots, P_r$, in such a way that it is (uniquely determined up to bisimulation, and) is associative-commutative on $P_1, \ldots, P_r$; we shall employ the standard CCS notation $P_1 \mid \ldots \mid P_r$ to denote this parallel asynchronous composition.

4 Secrecy Levels for Actions and Non-Interference

Our objective in this section is to attribute levels of secrecy to action symbols. Although it is possible to present our ideas with a semi-complete lattice of levels of secrecy, for readability we shall limit the discussion to two levels: denoted Hi and Lo (and referred to as *high* and *low*) respectively.

We assume that the set `Act` of action symbols is partitioned into two subsets denoted by Hi and Lo, representing secret and public events respectively; θ is assumed to be in Lo ('clock is observable by anyone'). Both Hi and Lo are assumed stable under conjugacy, i.e. $Hi = \bar{Hi}, Lo = \bar{Lo}$. An arbitrary action symbol in Hi (resp. Lo) will be denoted hi (resp. lo). When no confusion is likely, these sets will be referred to simply as H and L.

We define then - following [2] - an *observation criterion* for low level users $\mathcal{O}_L$ as follows: For any process Q, its observable image $Q/\mathcal{O}_L$ is the process constructed inductively from Q under the following inference rules:

(`No_Mask`)
$$\frac{P \xrightarrow{x} P' \qquad x \in Lo}{P/\mathcal{O}_L \xrightarrow{x} P'/\mathcal{O}_L}$$

(`Mask`)
$$\frac{P \xrightarrow{x} P' \qquad x \in Hi}{P/\mathcal{O}_L \xrightarrow{\theta} P'/\mathcal{O}_L}$$

The semantics behind these inference rules is that for any high level action x, the part observable by a low level observer reduces to the time unit that x consumes.

Remarks 4. 1) It is not true that $P/\mathcal{O}_L$ is bisimilar to the process obtained from P by replacing any occurrence of any high level action symbol hi in P by an occurrence of θ. For instance for any P, any $lo \neq \theta$ and any hi, we have by our definitions that: $lo * hi.P \approx 0$, but $(lo * \theta.P) \not\approx 0$.

2) The following assertion can however be deduced easily, via induction, from the above definition: For any process $P \in \mathcal{P}$, the process $P/\mathcal{O}_L$ is (bisimilar to) the process term obtained by replacing any occurrence of any high level action symbol hi by an occurrence of θ, *in the additive normal form* of P.

3) $\mathcal{O}_L$ is compatible with bisimulation (see the Appendix for a proof). Also note that if we replace the θ by the τ à la CCS, then this $P/\mathcal{O}_L$ is the same as what is generally denoted by P/H in the literature.

Definition 3 *A process P is said to satisfy* non-interference *iff the following holds:*
(RNI) $$P/\mathcal{O}_L \ \leq \ P$$

This notion will be referred to as RNI ('*Robust Non-Interference*'), in order to avoid confusion with the notions and notation prevailing in the literature. The adjective '*Robust*' is meant to reflect the fact that it is based on exact simulation, as well as exact (and not weak) bisimulation, as is shown in the proposition below. Following usual convention, for any process Q the process $Q \backslash H$ is that obtained from Q by eliminating the branches of Q where the action prefixes are in H: if $H = \{h_1, \ldots, h_n\}$, then $Q \backslash H = Q \backslash h_1 \backslash \ldots \backslash h_n$.

Theorem 1 *A process P satisfies RNI if and only if:* $P \backslash H \approx P/\mathcal{O}_L$.

Proof: We first prove the 'only if' assertion, inductively. Assume that P satisfies RNI, i.e. to say $P/\mathcal{O}_L \leq P$, and let $P \backslash H \xrightarrow{x} P' \backslash H$ be a transition on the process $P \backslash H$ to some process $P' \backslash H$. Then x must be a lo: so we also have $P/\mathcal{O}_L \xrightarrow{x} P'/\mathcal{O}_L$, and by inductive hypothesis we have that $P' \backslash H \approx P'/\mathcal{O}_L$. we deduce therefore that $P \backslash H \leq P/\mathcal{O}_L$. On the other hand, if z is any action such that we have a transition $P/\mathcal{O}_L \xrightarrow{z} Q$ to some Q, then by definition of $\mathcal{O}_L$, the action z must be lo, so cannot be a hi; so we actually get a transition of the form $P \backslash H \xrightarrow{z} Q$; we thus deduce that $P \backslash H \geq P/\mathcal{O}_L$. In other words: $P \backslash H \approx P/\mathcal{O}_L$.

The proof of the 'if' assertion is even more straightforward: consider any transition on $P/\mathcal{O}_L$ labeled by an action symbol a; then a must be a lo, so there is a transition on $P \backslash H$ labeled by a; which actually must be a transition on P with the same label. So $P/\mathcal{O}_L$ is simulated by P, that is to say P satisfies RNI. □

Remarks 5. It must be pointed out that the notion RNI of non-interference as defined above is a strict *generalization* of the notion of 'observation based non-deterministic non-interference' developed in the literature, e.g. the $BNNI$ of [6]. Indeed it suffices to replace our action symbol θ by the τ à la CCS, and replace the exact bisimulation of our definition by weak bisimulation, to 'deduce' $BNNI$ from RNI. Algebraically this could be done by modifying the product of actions, and adding to the equational theory the additional axiom $\tau.P \doteq P$. But a process satisfying $BNNI$ does *not* in general satisfy RNI; this is so, because in $BNNI$ high level actions are labeled τ for a low level user and τ is assumed there to be unobservable *and* 'without incidence on time'; cf. the example given below where h is assumed hi and a is lo, and R is the null process.

P = ha.R + a.R is BNNI (not RNI)   P = ha.R + θa.R is RNI (and BNNI)

Figure 1: RNI is a *strict generalization of* BNNI

The point here is that while a low level user can only get information a in the process P to the left, the time he spends to get this information from its two different branches is *not* the same. Whereas in the process P to the right, the low level user spends the same amount of time to retrieve the same information, whatever be the branch he observes.

5 Analysis of Information Flow

5.1 Value-Passing Process Algebras and Cryptographic Protocols

It is often more convenient to express processes in a so-called value-passing process algebra. Following Milner's treatment of CCS such terms are reduced to the pure process algebra by a non-deterministic enumeration of possible values for the value-variables. We recall here this convenient extended syntax.

An action in the value-passing syntax consists of either θ or combined names like $c(x)$ or $\bar{c}(v)$ where c is a channel name, v a value from an externally defined domain (e.g. integers) and x is a variable, not to be confused with the upper-case process variable X. The product of actions is defined so that any $\bar{c}(v)$ is the conjugate of $c(x)$: $c(x) * \bar{c}(v) = \theta$ (the *channel names must be equal*), $c(x) * \theta = \theta * c(x) = c(x)$, $\bar{c}(v) * \theta = \theta * \bar{c}(v) = \bar{c}(v)$, $\theta * \theta = \theta$; *no other product* is defined. Our (`Sync`) transition rule will get divided here into two cases:

(`Sync-comm`) $$\frac{P \xrightarrow{\bar{c}(v)} P' \quad Q \xrightarrow{c(x)} Q'}{P * Q \xrightarrow{\theta} P' * Q'[x/v]}$$

and a symmetrical rule where P is exchanged with Q.

(`Sync-absorb`) $$\frac{P \xrightarrow{\alpha} P' \quad Q \xrightarrow{\beta} Q' \quad \alpha * \beta = \gamma}{P * Q \xrightarrow{\gamma} P' * Q'}$$

where it is assumed that $(\alpha, \beta) \notin \{(c(x), \bar{c}(v)), (\bar{c}(v), c(x))\}$

It will also be useful to have terms of the form " `let` $x = e$ `in` P" which are defined as synonyms of $P[x/e]$; the issue of defining expression e's syntax and evaluating it to a value is assumed settled.

Value-passing processes are easily compiled into standard processes, while preserving their semantics, as follows. Assume without loss of generality that all values come from a given domain D; and for our current purpose of finite processes, that D is finite: $D = \{v_1, \ldots, v_n\}$. Then any value-passing process $\bar{c}(v).P$ can be compiled into a standard process term of the form $\bar{c}_v.P$, and similarly any value-passing process $c(x).Q$ can be compiled into $c_{v_1}.Q[x/v_1] + \ldots + c_{v_n}.Q[x/v_n]$; such a compilation uses in the general case the distributivity of '$*$' w.r.t. the choice operator '$+$. In the sequel we shall therefore refer to value-passing processes just as processes. The (standard) terminology of '*event*' will denote action symbols in our processes obtained by such a compilation.

For describing cryptographic protocols we need some special value terms of the form (m_1, m_2), $\pi_1(m)$, $\pi_2(m)$, $\mathtt{enc}(m, k)$, $\mathtt{dec}(m, k')$, representing respectively, a pair-message, the first component of a message, the second component of a message, the message m encrypted with key (integer) k and the decryption of m by k'. The value terms in these protocols are then assumed to satisfy the following identities:

$$\begin{array}{ll} (1) & \pi_1\ (x, y) = x \\ (2) & \pi_2\ (x, y) = y \\ (3) & \texttt{dec}(\ \texttt{enc}(x, k), k^{-1}\ ) = x \end{array}$$

which can be used – once oriented from left to right – as a convergent rewrite system on value terms. (We shall in general omit the parentheses around the arguments of π_1, π_2.)

Let m be a certain message appearing in protocol P and to be made inaccessible to the outside world. Let also C be the set of channels appearing in P and $\texttt{learnt} \notin C$ a channel on which an intruder process I may reveal m to the outside world (if he can manage to get hold of m as materialized by a binding-input of the form $c(m)$ or let $m = \ldots$ whose value of m is equal to the free variable m). An event such as $\texttt{learnt}\,(m)$ is violation of m's confidentiality and represents a failure of the cryptographic protocol. We will characterize failure properties of this kind in terms of properties based on bisimulation.

5.2 Information Tight Processes

We are concerned here with the problem of detecting if there is a flow of information from a given process P to another I. Typical situations are when P represents a cryptographic protocol as the parallel asynchronous composition of all its legitimate participants (called *principals*). Process I will represent an *intruder* who is not one of the principals; I will be assumed executed concurrently, in asynchronous composition with P. The reason for the asynchronous composition is that the protocol's expected behavior *is* a branch of the composite process; it also models the possibility of message interception.

The following idea is adapted from the so-called NDC property (“*non-deducibility on composition*”) developed in [7]. We propose to model ‘information leaks’ as special events that I may use to reveal information about P to the outside world. Following usual convention, we assume that: i) at least one of the channels used by to I is not used by the principals acting in P; ii) all the channels used by the principals are assumed (‘public’, meaning) accessible to all, in particular to the intruder. This last assumption is legitimate since messages exchanged between the principals are all supposed encrypted appropriately.

Definition 4 *Let P, I be processes, C the set of all channels in P, and assume that C is a strict subset of the channels in I. Then process P is said to be* `information tight` *for process I iff $(P \mid I) \setminus C \approx P \setminus C$.*

By construction the only possible action for $P \setminus C$ is θ; but we use strong bisimulation for which $\theta^2.0 \not\approx \theta.0 \not\approx 0$. In this context, the behavior of $P \setminus C$ reveals the existence of all synchronizations internal to P and nothing else. Information tightness of P for I thus means that:

1. if I interferes with P's internal synchronizations the resulting protocol behavior is unchanged, i.e. I induces *no parasite behavior of* P;

2. and I is unable to produce any *information leak* on a channel unknown to P.

Thus, the absence of information tightness of P for I corresponds to either or both of two segments of observability: *parasite behavior* in P, *information leak* outside P.

Example 1. This simple example justifies the use θ instead of τ and (strong) bisimulation in our above definition of information tightness. Let P be a trivial "ping-pong protocol" where two principals exchange a value back and forth: $P = \bar{a}(1).a(y) \mid a(x).\bar{a}(x)$. Its channel set is $C = \{a\}$, and $P \setminus C = P \setminus a \approx \theta.\theta.0$. Consider an intruder I(having at his disposal possibly many channels, but) using here only a: $I = a(z)$. Let us check if P is information tight for I; we get:

$$(P \mid I) \setminus a \approx \theta.0 + \theta.(\theta.0 + \theta.0) \approx \theta.0 + \theta.\theta.0$$

We deduce then that $(P \mid I) \setminus a \not\approx P \setminus a$: there is a parasite behavoir in P, and our ping-pong protocol is *not* information tight for the intruder I, who is only using here the same channel as the principals in P.

In contrast, it must be noted that in a (weak) trace or weak bisimulation based analysis, $(P \mid I) \setminus a$ and $P \setminus a$ would both be considered 0, so equivalent. Suppose we opt for a trace based analysis that we reinforce with a non-idempotent θ ('strong trace' so to say if θ is seen as τ). We would then get, in the above example, that both $(P \mid I) \setminus a$ and $P \setminus a$ have the same language, namely $\{\theta, \theta.\theta\}$, so are equivalent.

Our analysis based on strong bisimulation is thus finer: it is capable of observing a parasite behaviors; this is unobservable in other theories. □

Example 2. Consider the trivial 'protocol' $P = \bar{a}(1).0 \mid a(x).0$ where two principals are simply exchanging message-value 1. Here the channel set is $C = \{a\}$ and the "internal" behavior of P, i.e. to say $P \setminus a$, is simply $\theta.0$.

Let us consider two possible intruders $I_0 = a(y).0$ and $I_1 = a(z).\bar{c}(z).0$. For I_0, we get $(P \mid I_0) \setminus a \approx \theta.0 + \theta.0 \approx \theta.0$. So P is information tight for I_0.

As for I_1, we get: $(P \mid I_1) \setminus a = (\ \bar{a}(1).0 \mid a(x).0 \mid a(z).\bar{c}(z).0\ ) \setminus a = \theta.0 + \theta.\bar{c}(1).0$; which shows that P is not information tight for I: indeed the value message-value 1 exchanged between the principals in P has been 'intercepted' by I who is using his private channel c to transmit it to the outside world. (Obviously in this example, a trace based analysis would have detected the information leak too.) □

5.3 Absence of Information Leak seen as RNI

In all the approaches that we know of, information flow between a protocol and an intruder – both seen as value-passing processes – is based on either (weak) trace equivalence (cf. Focardi et al), or weak bisimulation (cf. Mullins). We saw that under these analysis tools a parasite behavior of the protocol will remain unobservable. On the other hand, suppose we restrict our attention to only the absence of information leaks from P to I; this is what is generally called *confidentiality preservation* of P w.r.t. I in the literature. This has also been characterized as the property SNNI – 'strong non-deterministic non interference' – in [6]; and as BNAI – '(weak) bisimulation based non-deterministic admissible interference' – in [10]. It is therefore natural that in our turn we formulate a similar characterization in our setup, for the absence of information leaks from P to I.

For this purpose we need to slightly generalize the notion of RNI as follows, in the setup of value-passing CCS. Let X be any given set of channels. We can then define an observation criterion w.r.t. X, that we shall denote by $\mathcal{O}_X$ on processes along the same lines as we did earlier: all communication units along channels *not in* X are seen as θ. We may then define a process P to be RNI_X w.r.t. the set of channels X if and only if:

$$(\mathbf{RNI_X}) \qquad P/\mathcal{O}_X \approx P \setminus X^c$$

where X^c is the set of all channels not in X (possibly including θ). The observation criterion $\mathcal{O}_L$ we defined earlier appears then as the special case where X is taken as the set of all low level channels.

Definition 5 *Let P, I be processes, C the set of all channels in P, and assume that C is a strict subset of the channels in I. Let X be the set of action channels of I not shared with P, and $\mathcal{O}_X$ the corresponding observation criterion. Denote by P_I the process $(P \mid I) \setminus C$, where the intruder I is in asynchronous composition with P, in forced synchronization on the shared channels C.*

Then P is said to be `confidentiality preserving` *w.r.t. I, iff P_I satisfies the property RNI_X. We shall say that P has an* `information leak` *for I iff P is* not *confidentiality preserving w.r.t. I.*

For instance, the "trivial" protocol of Example 2 has an information leak for I_1.

5.4 The Wide Mouthed Frog Protocol

In this cryptographic protocol, two principals denoted A, B exchange a (symmetric) session key denoted k_{AB} intended to serve for encrypting and decrypting messages they will transmit to each other, during the given session. The key exchange between A and B is not done directly, but under the control of a key server S (in whom all the principals have confidence, and) with whom each principal X shares a secret key denoted k_{XS} used for encrypting/decrypting the messages he exchanges with S; when the server S receives a message from principal X, S recovers the value of k_{XS} with the help of a function `secret–key`() applied to X.

A simplified version of the protocol for which bugs are easy to detect (in particular via formal information flow analysis techniques), goes as follows:

$$\begin{aligned} &1)\ A \longrightarrow S : A, B,\ \texttt{enc}\ (k_{AB}, k_{AS}) \\ &2)\ S \longrightarrow B :\ \texttt{enc}\ (\{A, k_{AB}\}, k_{BS}) \\ &3)\ A \longrightarrow B :\ \texttt{enc}\ (m, k_{AB}) \end{aligned}$$

In other words: A *says to* S : I'm A, want to talk to B using the key k_{AB}; S *says to* B: A wants to talk to you using the key k_{AB}; subsequently A *sends to* B *message* m encrypted with the key k_{AB}.

The bug in this simplified version of the protocol can be depicted as the following attack by an intruder I :

$$\begin{array}{l} 1)\ \ A \longrightarrow I(S) : A, B,\ \texttt{enc}\ (k_{AB}, k_{AS}) \\ 1)\text{'}\ I(A) \longrightarrow S : A, I,\ \texttt{enc}\ (k_{AB}, k_{AS}) \\ 2)\ \ S \longrightarrow I :\ \texttt{enc}\ (\{A, k_{AB}\}, k_{IS}) \\ 3)\ A \longrightarrow I :\ \texttt{enc}\ (m, k_{AB}) \end{array}$$

In other words: intruder I intercepts message 1) addressed to S by A, and sends it to S as message 1)' after having swapped B's identity with his own, and claiming to be A; then S sends the session key intended by A to I – and not to B – so encrypted with k_{IS}; this gives I access to the session key k_{AB}; I can then intercept all further messages that A sends to B during the session (and decrypt them at ease).

The protocol can be seen as the following value-passing CCS (cf. e.g. [7]): Denote by c_1 (resp. c_2) the channel of communication between A (resp. B) and S; and by c_3 the channel between A and B; all transmissions being encrypted, these channels are assumed public. We get the following process terms:

$$A = \bar{c}_1(A, B, \texttt{enc}(k_{AB}, k_{AS}))\,.\,\bar{c}_3(\texttt{enc}(m, k_{AB})).$$

$$B = c_2(y)\,.\ \texttt{let}\ k = \pi_2(\texttt{dec}(y, k_{BS}))\ \texttt{in}\ c_3(z)\,.\ \texttt{let}\ m' = \texttt{dec}(z, k)\ \texttt{in}\ 0.$$

$$\begin{array}{l} S = c_1(x)\,.\ \texttt{let}\ A' = \pi_1(\pi_1 x)\ \texttt{in} \\ \qquad\qquad \texttt{let}\ B' = \pi_2(\pi_1 x)\ \texttt{in} \\ \qquad\quad \texttt{let}\ k = \texttt{dec}(\pi_2 x, \texttt{secret–key}(A'))\ \texttt{in}\ \bar{c}_2(enc(\{A', k\}, \texttt{secret–key}(B')))\,.\,S. \end{array}$$

(Here the S at the end has the semantics that S is again available as a key server.)

The process term representing the protocol is therefore $P = A \mid B \mid S$. The intruder I depicted above for whom P is not information tight, is the following process where c_4 is 'some' channel of communication:

$$I = c_1(x)\,.\,\bar{c}_1((A, I), \pi_2 x)\,.\,c_2(z)\,.\ \texttt{let}\ k = \pi_2(\texttt{dec}(z, k_{IS}))\ \texttt{in}\ c_3(u)\,.\,\bar{c}_4(\texttt{dec}(u, k)).0.$$

In the approaches where information flow analysis is based on traces or weak bisimulation (like $SNNI, BNNI$), the attack depicted above is detected by *assuming* that the channel c_4 is private for the intruder, and inaccessible to A, B, S; in other words, the message m from A to B that I has intercepted, is getting transmitted 'outside'. If we set $C = \{c_1, c_2, c_3\}$, it is therefore easily deduced that: $(P \mid I) \setminus C$ is not trace equivalent to $P \setminus C$.

The analysis based on our approach *is finer*: we do **not** need to assume that the intruder I uses any private channel to show that P is not information tight for I: Indeed assume that all the channels are public, that is $C = \{c_1, c_2, c_3, c_4\}$; if we compute the branches of $(P \mid I) \setminus C$ and $P \setminus C$, it is easily checked that these two processes are *not* bisimilar: In the process $P \setminus C$, the server S gets the message sent by A *preceded by* θ ('one clock count'); while in $(P \mid I) \setminus C$, the server gets in addition the same message from (someone pretending to be) A but this time *preceded by* θ^2 ('two clock counts').

This example shows that strong bisimulation is a much better basis for information flow analysis. It also explains (in part) why in the non-simplified version of the Wide Mouthed Frog protocol, timestamps are included in the encrypted part of the messages sent to S.

6 Conclusion

Our principal objective in this paper has been to build a CCS like formalism, but containing no internal actions with a magical behavior like the τ of CCS. A first and natural step in this direction is obviously to "make τ pay its dues" at least to the clock; τ then becomes observable; and its internal character reduces to synchronizing with every action symbol, when we assume that all the action symbols have the same incidence on the clock – which is legitimate as a first approximation. That gave us our action θ.

The other inference rules for the construction of our process terms then came rather naturally. The assumption that our processes are bounded in height arose from the fact we are interested in applying the resulting formalism to practical situations where such a property holds almost always. The equational characterization of bisimulation that we obtained has two motivations: i) first, it gave us a theoretical setup for deducing in formal terms the notion of parallel asynchronous composition between processes without having to introduce that at the basis; we saw that such a composition is needed for analyzing processes which model protocols; ii) the $ACUID$-equational theory is known to be useful in program specification based on set constraints; extending it with prefixes is therefore only natural, if we want to analyze programs specified that way.

We showed that a bisimulation based notion of non-interference in our setup provides an appropriate tool for confidentiality analysis of protocols, but also for discovering parasite behaviors which may get created by the presence of intruders; and we saw that parasite behaviors cannot be tracked by the other approaches based on $SNNI, BNNI$ or $BNAI$. This is a novel contribution, to our knowledge.

Several tasks remain: for instance we still have to define in formal terms the notion of authentication in protocol analysis. This is part of future work, as well as the realization of a tool implementing the ideas presented in this paper. We also hope to be able to exploit some of the equational ideas developed here, for discovering intruders as yet unknown, for some among the huge list of cryptographic protocols (cf. `http://www.lsv.ens-cachan.fr/spore/table.html`).

References

[1] M. Abadi, A.D. Gordon, *Reasoning about Cryptographic Protocols in the Spi Caclulus*, In. Proc. Int. Conf. CONCUR'97, Concurrency Theory, Springer-Verlag, LNCS 1243,

pp. 59–73, 1997

[2] G. Boudol, *Notes on Algebraic Cacluli of Processes*, In Logic and Models of Concurrent Systems, NATO ASI Series F-13, pp. 261–303? Springer, 1985.

[3] G. Boudol and I. Castellani. *Non–Interference for Concurrent Programs and Thread Systems*. Technical Report 4254, Institut National de Recherche en Informatique et en Automatique (INRIA), Sept. 2001. Accepted for publication in Theoretical Computer Science.

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

[5] R. Focardi, R. Gorrieri, *The Compositional Security Checker: a Tool for the Verification of Information Flow Security Properties*, IEEE Trans. on Software Engineering, 23(9):550-571, 1997.

[6] R. Focardi, *Analysis and Automatic Detection of Information Flow in Systems and Networks*. Ph.D Thesis - Tech. Report UBLCS-99-16, Dept. of Comp. Sc., University of Bologna, July 1999.

[7] R. Focardi, R. Gorrieri, F. Martinelli, *Secrecy in Security Protocols as Non Interference*, In Elect. Notes in Theo. Computer Sc., Vol 32, Elsevier, 2000.

[8] Y. Hirshfeld and F. Moller *Decidability Results in Automata and Process Theory*, In F. Moller and G. Birtwistle eds., Logics for Concurrency, structure versus Automata, Springer LNCS no. 1043, 1996.

[9] S. Lafrance and J. Mullins. *Bisimulation-based Non-deterministic Admissible Interference and its Applications to the Analysis of Cryptographic Protocols*. In Electronic Notes in Theor. Comp. Sc. (ENTCS), Elsevier, vol 61, 2001.

[10] J. Mullins, *Non Deterministic Admissible Interference*, In Journal of Universal Computer Science, 6(11):1054–1070, November 2000.

[11] G. Smith, D. M. Volpano, and C. Irvine. *A Sound Type System for Secure Flow Analysis*. *Journal of Computer Security*, 4(3):167 – 187, 1996.

[12] G. Smith and D. M. Volpano. *Secure Information Flow in a Multi-Threaded Imperative Language*. In Proc. of the 25^{th} ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL '98), San Diego, Jan. 1998, pages 355 – 364.

Appendix

Proposition 5 *The choice operator '+' is associative, commutative, and idempotent, up to bisimulation. That is to say, for any process terms P, Q, R, we have:*

$$P + (Q + R) \approx (P + Q) + R, \qquad P + Q \approx Q + P, \qquad P + P \approx P$$

Proof: *Associativity*: Any transition of $P + (Q + R)$ must come either from P or from $(Q + R)$ whose transitions in turn must come from Q or R. Hence $(P + Q) + R$ must have exactly the same transition and we thus conclude that $P + (Q + R) \leq (P + Q) + R$; the other simulation is proved similarly.

Commutativity: If $P + Q$ has a transition on P then $Q + P$ has exactly the same transition since the rule (`Sum`) is symmetric. As a result $P + Q \leq Q + P$; the other simulation follows by symmetry.

Idempotence: If $P + P$ has a transition, it must come from rule (`Sum`) and hence from P; so $P + P \leq P$. We also have $P \leq P + P$ because any transition on P is also a transition on $P + P$. □

Proposition 6 *Let P, Q be any process terms. Then we have:*

$$i)\ P + 0 \approx P, \qquad ii)\ P * 0 \approx 0, \qquad iii)\ P * Q \approx Q * P.$$

Proof: i) Any transition of $P + 0$ is either a transition of P or a transition of 0. But the latter has none and so the transitions of $P + 0$ are exactly those of P.

ii) Any transition from $P * 0$ would imply a synchronous transition of P and a transition of 0. But the latter does not exist, hence there are no transitions from $P * 0$ i.e. the (non-existent) transition of $P * 0$ are precisely the transitions of 0.

iii) Follows from the fact that rule (`Sync`) and the product of actions $x * y$, as defined earlier, are both symmetric. □

Proposition 7 *Prefix and (parallel) synchronous composition commute:*

$$a.P * b.Q \approx (a * b).(P * Q)$$

Proof: Any transition from $a.P * b.Q$ implies a transition of $a.P$ and a transition of $b.Q$ and yields $P * Q$ on action $a * b$, assuming the product is defined. This is also a transition on $(a * b).(P * Q)$ yielding the same process $P * Q$, hence the left-hand side of the equation is simulated by the right-hand side. The inverse simulation is also proved by the same correspondence. □

Proposition 8 *Synchronous composition '$*$' distributes over the choice operator '$+$' up to bisimulation:*

$$P * (Q + R) \approx P * Q + P * R.$$

Proof: Let LHS $= P * (Q + R)$ and RHS $= P * Q + P * R$.

LHS $\leq$ RHS: Suppose first that LHS $\xrightarrow{z}$ then rule (`Sync`) is responsible, so that there are transitions $P \xrightarrow{x} P'$ and $(Q + R) \xrightarrow{y} Z$ such that $x * y = z$. But the last transition comes either from $Q \xrightarrow{y} Z$ or $R \xrightarrow{y} Z$. We have therefore ($P \xrightarrow{x} P'$ and ($Q \xrightarrow{y} Z$ or $R \xrightarrow{y} Z$)); which amounts to (($P \xrightarrow{x} P'$ and $Q \xrightarrow{y} Z$) or ($P \xrightarrow{x} P'$ and $R \xrightarrow{y} Z$)). Applying rule (`Sync`) to both alternatives yields ($P * Q \xrightarrow{z} P' * Z$ or $P * R \xrightarrow{z} P' * Z$) which in turn, by rules (`Sum`) means that RHS $\xrightarrow{z} P' * Z$. But by construction, the original transition from LHS must also have been LHS $\xrightarrow{z} P' * Z$ and trivially $P' * Z \leq P' * Z$. We have thus shown that LHS $\leq$ RHS.

$LHS \geq RHS$: Suppose now that RHS $\xrightarrow{z}$ then rule (`Sum`) is responsible, and either $P * Q \xrightarrow{z} P' * Q'$ or $P * R \xrightarrow{z} P' * R'$. Both alternatives can only hold by virtue of ($P \xrightarrow{x} P'$, $Q \xrightarrow{y} Q'$ and $x * y = z$) or ($P \xrightarrow{x} P'$, $R \xrightarrow{y} R'$ and $x * y = z$) which is logically equivalent to ($P \xrightarrow{x} P'$ and $x * y = z$ and ($Q \xrightarrow{y} Q'$ or $R \xrightarrow{y} R'$)). By rule (`Sum`), this is in turn equivalent to ($P \xrightarrow{x} P'$ and $x * y = z$ and ($Q + R \xrightarrow{y} Q'$ or $Q + R \xrightarrow{y} R'$)), which implies by rule (`Sync`) that either $P * (Q + R) \xrightarrow{z} P' * Q'$ or $P * (Q + R) \xrightarrow{z} P' * R'$ depending on the initial transition from RHS. Thus any transition of RHS is also a transition of LHS and one concludes that RHS $\leq$ LHS. $\square$

Remark 6. (Parallel) Synchronous composition is *not associative*: in general we do not have $P * (Q * R) \not\leq (P * Q) * R$. Here is an example: Let $P = a$, $Q = \bar{a}$ and $R = b$, with $b \notin \{a, \theta\}$. Then $P * (Q * R) = a * (\bar{a} * b)$ has no possible transition because $Q * R$ has none. But on the other hand $(P * Q) * R = (a * \bar{a}) * b \xrightarrow{\theta * b} 0 * 0$, because $(P * Q) = (a * \bar{a}) \xrightarrow{\theta} 0$ and $R = b \xrightarrow{b} 0$. $\square$

Proposition 9 *Double restriction is commutative and idempotent on its action arguments, up to bisimulation; i.e. for any $P \in$ `Proc` and action symbols x, y, we have:*

$$P \setminus x \setminus y \approx P \setminus y \setminus x, \qquad P \setminus x \setminus x \approx P \setminus x.$$

Proof: Any transition from $P \setminus x \setminus y$ is, by rule (`Restr`) applied twice, a transition of P over an action which is neither x nor y. But this is also the characteristic of a transition from $P \setminus y \setminus x$. The two processes have exactly the same transitions and must be bisimilar. The second assertion is proved just as easily. $\square$

Proposition 10 *Restriction distributes over choice, up to bisimulation:*

$$(P + Q) \setminus x \approx P \setminus x + Q \setminus x.$$

Proof: Any transition of $(P+Q) \setminus x$ is either a transition of P or a transition of Q which is not on action x. If it has the form $P \xrightarrow{a} P', a \neq x$ then it is also a transition of $P \setminus x$ and similarly for the other alternative. Hence the left-to-right simulation holds. Conversely, any transition of $P \setminus x + Q \setminus x$ is either a transition of P not on x or a transition of Q not on x and such a transition is also one of $(P + Q) \setminus x$. $\square$

Proposition 11 *Restriction distributes conditionally over prefix, up to bisimulation:*

$$(a.P) \setminus x \approx \begin{cases} a.(P \setminus x) & \textit{if} \quad x \neq a \\ 0 & \textit{if} \quad x = a \end{cases}.$$

Proof: If $x = a$ then $(a.P) \setminus x$ has no possible transition and is thus bisimilar to 0. If $x \neq a$ then rule (`Restr`) allows its only possible transition on a and yields $P \setminus x$. This make the process bisimilar to $a.(P \setminus x)$. $\square$

We also get just as easily the fact that: *Restriction of a null process is null:* $0 \setminus a \approx 0$.

Remarks 7. 1). Actually the (`Restr`) rule is not needed in the calculation of the additive normal form for a process term. This means that, once such a normal form has been obtained for any P (without using the (`Restr`) rule), for any action symbol $x \neq \theta$ one can deduce the term $P \setminus x$ from this normal form, uniquely up to bisimulation, 'recursively' as follows: if $P = \sum_{a \in A} a.P_a$, then: $P \setminus x = \sum_{a \in A, a \neq x} a.(P_a \setminus x)$.

2) The implication of the preceding remark is that, the last five equational axioms $(C \setminus), (I \setminus), (D \setminus), (P \setminus), (Z \setminus)$ concerning (`Restr`) are *not* needed for the equational analysis of our process algebra. It can be shown without difficulty, (e.g. with the help of existing equational theorem provers) that the set of remaining equational axioms, transformed into rewrite rules by orienting them left to right, form a convergent rewrite system; we shall refer to this system as a $PACUID$-rewrite system ('$ACUID$ extended with Prefixes').

Proposition 12 *The observation criterion $\mathcal{O}_L$ is compatible with bisimulation: i.e. to say, if $P \approx Q$ then $P/\mathcal{O}_L \approx Q/\mathcal{O}_L$.*

Proof: Assume $P \approx Q$. By symmetry it will be sufficient to prove that $P/\mathcal{O}_L$ is simulated by $Q/\mathcal{O}_L$.
If $P/\mathcal{O}_L \xrightarrow{lo} P'/\mathcal{O}_L$ because $P \xrightarrow{lo} P'$ (in particular if $lo = \theta$) then the simulation of P by Q implies that $Q \xrightarrow{lo} Q'$ and P' is simulated by Q'. Since the observation criterion is oblivious to lo, this implies that that $Q/\mathcal{O}_L \xrightarrow{lo} Q'/\mathcal{O}_L$ and by induction hypothesis (the target processes are smaller) $P'/\mathcal{O}_L$ is simulated by $Q'/\mathcal{O}_L$.
If $P/\mathcal{O}_L \xrightarrow{\theta} P'/\mathcal{O}_L$ because $P \xrightarrow{hi} P'$ then the simulation of P by Q implies that $Q \xrightarrow{hi} Q'$ and P' is simulated by Q'. This implies that that $Q/\mathcal{O}_L \xrightarrow{\theta} Q'/\mathcal{O}_L$ and by induction hypothesis (the target processes are smaller) $P'/\mathcal{O}_L$ is simulated by $Q'/\mathcal{O}_L$.