Time and Probability based Information Flow Analysis

Ruggero Lanotte, Andrea Maggiolo–Schettini, and Angelo Troina

Abstract—In multilevel systems it is important to avoid unwanted indirect information flow from higher levels to lower levels, namely the so called *covert channels*. Initial studies of information flow analysis were performed by abstracting away from time and probability. It is already known that systems that are proved to be secure in a possibilistic framework may turn out to be insecure when time or probability are considered. Recently, work has been done in order to consider also aspects either of time or of probability, but not both. In this paper we propose a general framework, based on Probabilistic Timed Automata, where both *probabilistic* and *timing covert channels* can be studied. We define a Non-Interference security property and a Non Deducibility on Composition security property, which allow expressing information flow in a timed and probabilistic setting. We then compare these properties with analogous ones defined in contexts where either time or probability or neither of them are taken into account. This permits a classification of the properties depending on their discerning power. As an application, we study a system with covert channels that we are able to discover by applying our techniques.

Index Terms—Probabilistic Timed Automata, Multilevel Security, Information Flow Analysis, Weak Bisimulation.

1 Introduction

In a multilevel system every agent is confined in a bounded security level; information can flow from a certain agent to another agent only if the level of the former is lower than the level of the latter. Access rules can be imposed by the system in order to control *direct* unwanted transmission from higher levels to lower levels; however, it could be possible to transmit information *indirectly* by using system side effects. Usually, this kind of indirect transmissions, called *covert channels*, do not violate the access rules imposed by the system and are difficult to discover.

The existence of covert channels has led to the more general approach of *information flow security*, which aims at controlling the way information may flow among different entities. The idea is to try to directly control the whole flow of information, rather than only the direct communication among agents. In [29] the authors introduce the notion of *Non-Interference*, stating, intuitively, that low level agents should not be able to deduce anything about the activity of high level agents. By imposing some information flow rules, it is possible to control direct and indirect leakages, as both of them give rise to unwanted information flows.

In the literature there are many different definitions of security based on the information flow idea, and each is formulated in some system model (see, e.g., [29], [41], [30], [26], [28], [1], [25], [7], [2], [27]). Most of the considered properties are based on analysis of information flow that does not take into consideration aspects of time or probability in the analysis of the behaviour of the system. Therefore, these models are not able to check the existence of probabilistic or timing covert channels which exploit these quantitative aspects to transmit information. To overcome this, a significant work has been done in order to extend the study by considering either time (see, e.g., [28], [1], [25], [7]) or probability (see, e.g., [30], [2], [22]).

This has required the use of descriptive means for systems which allow expressing time and probability. Timed Automata have been introduced by Alur and Dill [5] as an extension of ω-Automata to describe real-time systems. Timed Automata are equipped with variables measuring time, called *clocks*. Transitions are guarded by *clock constraints*, which compare the value of clocks against constants, and by *reset updates*, which reset clocks to the initial value 0. Extensions with probability have been proposed (e.g. in [4], [8], [34], [35]). In this paper we are interested in a general framework where both probabilistic and timing covert channels can be studied. In particular, we consider a class of Probabilistic Timed Automata (PTAs) where the analysis of information flow security properties is carried out by partitioning the set of action labels into high level actions and low level actions.

The framework of PTAs allows specification of timed systems showing a probabilistic behaviour in an intuitive and succinct way. Therefore, within the framework of PTAs, where time and probabilities are taken into consideration, the modeler can describe, in the same specification, different aspects of a system, and analyze on a single model real-time properties, performance and reliability properties (by using, e.g., classical model checking techniques), and information flow security properties

Preliminary results of this paper have been presented in [37].

- *R. Lanotte is with Università dell'Insubria.*
- *A. Maggiolo–Schettini is with Università di Pisa.*
- *A. Troina is with Università di Torino.*

able to detect both probabilistic and timing covert channels.

On the one hand, there are new covert channels that arise from the interplay of time and probability which are not detected with the techniques used to study only timed information flow or probabilistic information flow. On the other hand, our weak bisimulation relation for PTAs preserves the results deriving from the study of nondeterministic systems or systems where only time or only probabilities are considered. For example, the results proved for the study of information flow with the use of weak bisimulation in a timed framework (see the work by Agat [1]) or in a probabilistic setting (see the work by Aldini et al. [2]) are preserved by our weak bisimulation for PTAs. We actually feel that the examples studied in the above mentioned papers could be encoded, with reasonable abstractions, within the framework of PTAs.

The remainder of this paper is organised as follows. In Section 2 we present our model of Probabilistic Timed Automata and we recall the definitions of Probabilistic Automata, Timed Automata and Nondeterministic Systems. We define bisimulation equivalences and operations for all these models. In Section 3 we define the Non-Interference and the Non Deducibility on Composition security properties in a probabilistic and timed setting described by Probabilistic Timed Automata. The concept of Non-Interference was proposed originally in a purely nondeterministic setting [29], [41], [26]. Non Deducibility on Composition was proposed by Focardi and Gorrieri in order to detect insecure behaviour that the Non-Interference property is not able to detect. We show here that these concepts, together with the analogous concepts for Probabilistic Automata and Timed Automata, can be expressed in a single framework where both probability and time are considered. In Section 4 we study, as an application, some covert channels which may arise in a network shared buffer. In Section 5 we draw our conclusions.

2 THE FORMALISM

We recall the definition of Probabilistic Timed Automata, operations and bisimulation for these automata, we proposed in [37], [44].

This model is inspired by models of Probabilistic Timed Automata in the literature (see, as examples, [8], [34], [4]). Actually, our model is closer to [8] which extends Markov Decision Processes (MDPs) (see [9], [33]) with time. The main difference with the models in [34] and [4] is that we define, as for MDPs, probability distributions over the set of transitions instead of distributions over target states. This choice, while requiring some particular attention in the definition of the semantics (in particular in the normalization of probabilities), allows us to easily get rid of probabilities when reducing to the non probabilistic case, thus allowing for simpler proofs when showing that equivalences are preserved in the less expressive models where time or probabilities are abstracted away. Note, however, that all those frameworks are equivalent, and translations can be given from each of those models to any other. Hence, the results provided in this paper for our model can be immediately applied to the other models of Probabilistic Timed Automata.

Abstracting away from time, Probabilistic Automata are defined as a particular case. We recall also the definitions of Timed Automata ([5]) and of Nondeterministic Systems.

2.1 Probabilistic Timed Automata

A *discrete probability measure* over a countable set Q is a function $\mu : 2^Q \to [0,1]$ such that $\mu(Q) = 1$ and for each countable family $\{Q_i\}$ of pairwise disjoint elements of 2^Q, $\mu(\cup_i Q_i) = \sum_i \mu(Q_i)$. We adopt the convenient abuse of notation $\mu(q)$ for $\mu(\{q\})$.

Let us assume a set X of positive real variables called *clocks*. A *valuation* over X is a mapping $v : X \to \mathbb{R}^{\geq 0}$ assigning real values to clocks. For a valuation v and a time value $t \in \mathbb{R}^{\geq 0}$, let $v + t$ denote the valuation such that $(v + t)(x) = v(x) + t$, for each clock $x \in X$.

The set of *constraints* over X, denoted $\Phi(X)$, is defined by the following grammar:

$$\phi ::= x \sim c \,|\, \phi \wedge \phi \,|\, \neg\phi \,|\, \phi \vee \phi \,|\, true$$

where ϕ ranges over $\Phi(X)$, $x \in X$, $c \in \mathbb{Q}$ and $\sim \in \{<, \leq, =, \neq, >, \geq\}$.

We write $v \models \phi$ when *the valuation v satisfies the constraint ϕ*. Formally, $v \models x \sim c$ iff $v(x) \sim c$, $v \models \phi_1 \wedge \phi_2$ iff $v \models \phi_1$ and $v \models \phi_2$, $v \models \neg\phi$ iff $v \not\models \phi$, $v \models \phi_1 \vee \phi_2$ iff $v \models \phi_1$ or $v \models \phi_2$, and $v \models true$.

Let $B \subseteq X$; with $v[B]$ we denote the valuation resulting after resetting all clocks in B. More precisely, $v[B](x) = 0$ if $x \in B$, $v[B](x) = v(x)$, otherwise. Finally, with $\mathbf{0}$ we denote the valuation with all clocks reset to 0, namely $\mathbf{0}(x) = 0$ for all $x \in X$.

Definition 2.1: A *Probabilistic Timed Automaton* (PTA) is a tuple $A = (\Sigma, X, Q, q_0, Inv, \delta, \Pi)$, where:

- Σ is a finite alphabet of actions;
- X is a finite set of positive real variables (*clocks*);
- Q is a finite set of states and $q_0 \in Q$ is the initial state;
- $Inv : Q \to \Phi(X)$ is a function assigning a constraint $\phi \in \Phi(X)$ (called *state invariant*) to each state in Q.
- δ is a finite set of transitions in $Q \times \Sigma \times \Phi(X) \times 2^X \times Q$.
- $\Pi = \{\pi_1, \ldots, \pi_n\}$ is a finite set of probability distributions as functions $\pi_i : \delta \to [0,1]$, for any $i = 1, \ldots, n$, where $\pi_i(e)$ is the probability of performing transition e according to distribution π_i.

For a state $q \in Q$, we denote with $\delta(q)$ the set of transitions with q as source state, i.e. the set $\delta(q) = \{(q', a, \phi, B, q'') \in \delta \mid q' = q\}$. We require that $\sum_{e \in \delta(q)} \pi_i(e) = 1$ for any i and q. Moreover, we assume that for all e_j there exist some π_i such that $\pi_i(e_j) > 0$.

Transitions from a state q take the form (q, a, ϕ, B, q') for a starting state q, a label $a \in \Sigma \cup \{\tau\}$, an enabling condition $\phi \in \Phi(X)$, a set B of clocks to be reset and a target state q'.

As we shall see in the next subsection, the nondeterministic behaviour of a PTA derives by the fact that there will be, in general, more probability distributions in Π that will enable a move from a given state. The probabilistic behaviour of a PTA is modelled by the fact that a transition (q, a, ϕ, X, q') will be chosen according to some probability distribution $\pi_i \in \Pi$.

2.1.1 Semantics of PTAs

A *configuration* of a PTA A is a pair (q, v), where $q \in Q$ is a state of A, v is a valuation over X and $v \models Inv(q)$. The initial configuration of A is represented by $(q_0, \mathbf{0})$.

There is a discrete *transition step* from a configuration $s_i = (q_i, v_i)$ to a configuration $s_j = (q_j, v_j)$ through action $a \in \Sigma \cup \{\tau\}$, written $s_i \xrightarrow{a} s_j$, if there is a transition $e = (q_i, a, \phi, B, q_j) \in \delta$ such that $v_i \models \phi$, $v_j = v_i[B]$, $v_i \models Inv(q_i)$ and $v_j \models Inv(q_j)$.

The values of all the clocks increase uniformly with the passage of time. Thus, there is a *time step* from a configuration $s_i = (q_i, v_i)$ to a configuration $s_j = (q_j, v_j)$ through time $t \in \mathbb{R}^{>0}$, written $s_i \xrightarrow{t} s_j$, if $q_j = q_i$, $v_j = (v_i + t)$ and $\forall t' \in [0, t]\ v_i + t' \models Inv(q_i)$.

An *execution fragment* of A is a finite sequence of steps $\sigma = s_0 \xrightarrow{\alpha_1} s_1 \xrightarrow{\alpha_2} \dots \xrightarrow{\alpha_k} s_k$, where $s_0, \dots, s_k$ are configurations, and $\alpha_i \in \Sigma \cup \{\tau\} \cup \mathbb{R}^{>0}$. With $ExecFrag_A$ we denote the set of execution fragments of A, and with $ExecFrag_A(s)$ we denote the set of execution fragments of A starting from configuration s. We define $last(\sigma) = s_k$ and $|\sigma| = k$. For any $j \leq |\sigma|$, with σ^j we define the sequence of steps The execution fragment σ is called *maximal* iff there are no configuration s and $\alpha \in \Sigma \cup \{\tau\} \cup \mathbb{R}^{\geq 0}$ such that $\sigma \xrightarrow{\alpha} s$.

With $\mathcal{S}_A$ we denote the set of configurations reachable by A, more precisely, $\mathcal{S}_A = \{last(\sigma) \mid \sigma \in ExecFrag_A(s_0)\}$.

An *execution* of A is either a maximal execution fragment or an infinite sequence of steps $s_0 \xrightarrow{\alpha_1} s_1 \xrightarrow{\alpha_2} \dots$, where $s_0, s_1 \dots \in \mathcal{S}_A$ and $\alpha_1, \alpha_2, \dots \in \Sigma \cup \{\tau\} \cup \mathbb{R}^{\geq 0}$. We denote with $Exec_A$ the set of executions of A, and with $Exec_A(s)$ the set of executions of A starting from s.

Given a configuration $s = (q, v)$, with $Adm(s) = \{(q, a, \phi, B, q') \in \delta \mid v \models \phi \text{ and } v[B] \models Inv(q')\}$ we represent the set of admissible transitions that an automaton could execute from configuration s, and we say that a transition in $Adm(s)$ is *enabled* in s. Given two configurations $s = (q, v)$, $s' = (q', v')$, and given $a \in \Sigma \cup \{\tau\}$, we represent with $Adm(s, a, s') = \{(q, a, \phi, B, q') \in \delta \mid v \models \phi \ \wedge \ v' = v[B] \models Inv(q')\}$ the set of transitions that lead from configuration s to configuration s' through a transition step labeled with a[1].

A configuration $s = (q, v)$ is called *terminal* iff $Adm(s') = \emptyset$ for all $s' = (q, v + t)$ with $t \in \mathbb{R}^{\geq 0}$; we denote with S_T the set of terminal configurations.

The probability of executing a transition step from a configuration s is chosen, among all the transitions enabled in s, according to the values returned by some distribution π, while we set the probability of executing a time step labeled with $t \in \mathbb{R}^{>0}$ to the value 1. Intuitively, a PTA chooses nondeterministically the distribution of a transition step or to let time elapse by performing a time step, and, in this case, also the amount of time passed is chosen nondeterministically.

Executions and execution fragments of a PTA arise by resolving both the nondeterministic and the probabilistic choices [34]. To resolve the non-deterministic choices of a PTA, we introduce now *schedulers* of PTAs.

A *scheduler* of a PTA A is a partial function from $ExecFrag_A$ to $\Pi \cup \mathbb{R}^{>0}$. Given a scheduler F and an execution fragment σ, we assume that F is defined for σ if and only if $\exists s \in \mathcal{S}_{\mathcal{A}}$ and $\alpha \in \Sigma \cup \{\tau\} \cup \mathbb{R}^{>0}$ such that $last(\sigma) \xrightarrow{\alpha} s$.

For a scheduler F of a PTA A we define $ExecFrag_A^F$ (resp. $Exec_A^F$) as the set of execution fragments (resp. the set of executions) $\sigma = s_0 \xrightarrow{\alpha_1} s_1 \xrightarrow{\alpha_2} s_2 \xrightarrow{\alpha_3} \dots$ of A such that, for any i, $\alpha_i \in \mathbb{R}^{>0}$ iff $F(\sigma^{i-1}) = \alpha_i$, and $\alpha_i \in (\Sigma \cup \{\tau\})$ iff $\exists e \in Adm(s_{i-1}, \alpha_i, s_i)$ and $\pi_i(e) > 0$, where $\pi_i = F(\sigma^{i-1})$.

Given a scheduler F and an execution fragment $\sigma = s_0 \xrightarrow{\alpha_1} s_1 \xrightarrow{\alpha_2} s_2 \xrightarrow{\alpha_3} \dots \xrightarrow{\alpha_k} s_k \in ExecFrag_A^F$, if $k = 0$ we put $P_A^F(\sigma) = 1$, else, if $k \geq 1$, we define $P_A^F(\sigma) = P_A^F(\sigma^{k-1}) \cdot p$ where

$$p = \begin{cases} \dfrac{\sum_{e \in Adm(s_{k-1}, \alpha_k, s_k)} (F(\sigma^{k-1}))(e)}{\sum_{e \in Adm(s_{k-1})} (F(\sigma^{k-1}))(e)} & \text{if } \alpha_k \in \Sigma \cup \{\tau\} \\ 1 & \text{if } \alpha_k \in \mathbb{R}^{>0}. \end{cases}$$

Notice that such a measure is consistent, since we are assuming, given the execution fragment σ, that there is a step from s_{k-1} to s_k labeled with α_k. Now, if $\alpha_k \in \mathbb{R}^{>0}$, then $p = 1$, otherwise the probability of going from s_{k-1} to s_k through a discrete transition labeled with α_k is renormalized according to the transitions enabled in s_{k-1}. In this latter case, $Adm(s_{k-1}) \neq \emptyset$, since there exists at least the step $s_{k-1} \xrightarrow{\alpha_k} s_k$.

Given a scheduler F, let $Exec_A^F(s)$ be the set of executions in $Exec_A^F$ starting in s, $ExecFrag_A^F(s)$ be the set of execution fragments in $ExecFrag_A^F$ starting in s, and $\sigma Field_A^F(s)$ be the smallest sigma field on $Exec_A^F(s)$ containing the basic cylinders $\sigma\uparrow$, where $\sigma \in ExecFrag_A^F(s)$. The probability measure $Prob_A^F$ is the unique measure on $\sigma Field_A^F(s)$ such that $Prob_A^F(\sigma\uparrow) = P_A^F(\sigma)$.

Note that, given a PTA A and a scheduler F, the executions of A *driven* by F do not contain any nonde-

1. Since the underlying meaning is the same, we overloaded the function Adm to represent either the admissible transitions exiting a given configuration, or, more specifically, the set of transitions that, starting from a given configuration, lead to a certain configuration executing the specified action.

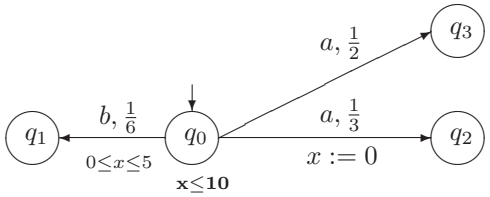


Fig. 1. Example of PTA.

terministic choice. Hence, a PTA A driven by a scheduler F gives rise to a fully probabilistic behaviour.

Example 2.2: In Figure 1 we show an example of PTA with $\Pi = \{\pi\}$. Transitions of the automaton are $e_1 = (q_0, b, 0 \leq x \leq 5, \emptyset, q_1)$, $e_2 = (q_0, a, true, \{x\}, q_2)$ and $e_3 = (q_0, a, true, \emptyset, q_3)$. The automaton has only a probability distribution π, with $\pi(e_1) = \frac{1}{6}$, $\pi(e_2) = \frac{1}{3}$ and $\pi(e_3) = \frac{1}{2}$. As required, $\sum_{e \in \delta(q_0)} \pi(e) = 1$ holds.

Intuitively, from the initial configuration $(q_0, \mathbf{0})$, the PTA may nondeterministically choose whether to perform some time step and update the value of clock x (for a time less or equal than 10, respecting the state invariant) or to perform, probabilistically, transitions e_1, e_2 or e_3 with probabilities $\frac{1}{6}$, $\frac{1}{3}$ and $\frac{1}{2}$, respectively.

If some time step is performed in state q_0, such that the value of clock x becomes greater than 5, then the transition labeled with b cannot be performed anymore, and the probabilities of performing the other transitions should be redistributed. In this case, even if the transition was at some point enabled in state q_0, we might intuitively say that the automaton has consumed too much time resource to be able to perform transition e_1 anymore. Even if the case is quite simple in the depicted automaton, similar situations may arise whenever the automaton returns to a certain state at different times and some of the transitions may not be enabled anymore. Note that re-normalizing probability at run-time allows us to relax the condition of *admissible target states* used in [34].

Examples of executions of the PTA in Figure 1 are $\sigma_1 = (q_0, 0) \xrightarrow{9.7} (q_0, 9.7) \xrightarrow{a} (q_2, 0) \xrightarrow{3} (q_2, 3)$ and $\sigma_2 = (q_0, 0) \xrightarrow{3} (q_0, 3) \xrightarrow{b} (q_1, 3) \xrightarrow{1.2} (q_1, 4.2)$ with $P(\sigma_1) = \frac{2}{5}$ and $P(\sigma_2) = \frac{1}{6}$, where (q, t) represents the configuration composed by the state q and the valuation v such that $v(x) = t$.

In the following, A is a PTA, F is a scheduler of A, $\hat{\alpha}$ stands for α if $\alpha \in \Sigma \cup \mathbb{R}^{>0}$ and for ε (the empty string) if $\alpha = \tau$, $s \in \mathcal{S}_A$ and $\mathcal{C} \subseteq \mathcal{S}_A$.

Let $Exec_A^F(s, \tau^* \hat{\alpha} \tau^*, \mathcal{C})$ be the set of executions that lead to a configuration in $\mathcal{C}$ via a sequence belonging to the set of sequences $\tau^* \hat{\alpha} \tau^*$ starting from the configuration s and crossing configurations equivalent to s. Finally, we define the probability $Prob_A^F(s, \tau^* \hat{\alpha} \tau^*, \mathcal{C}) = Prob_A^F(Exec_A^F(s, \tau^* \hat{\alpha} \tau^*, \mathcal{C}))$.

2.1.2 Regions

We recall the definitions of clock equivalence [5]. Clock equivalence is a finite index equivalence relation permitting to group sets of evaluations and to have decidability results.

Let A be a PTA; with C_A we denote the greatest constant that appears in A.

Let us consider the equivalence relation $\sim$ over clock valuations containing precisely the pairs (v, v') such that:

- for each clock x, either $\lfloor v(x) \rfloor = \lfloor v'(x) \rfloor$, or both $v(x)$ and $v'(x)$ are greater than C_A;
- for each pair of clocks x and y with $v(x) \leq C_A$ and $v(y) \leq C_A$ it holds that $fract(v(x)) \leq fract(v(y))$ iff $fract(v'(x)) \leq fract(v'(y))$ (where $fract(\cdot)$ returns the fractional part of the argument);
- for each clock x with $v(x) \leq C_A$, it holds that $fract(v(x)) = 0$ iff $fract(v'(x)) = 0$.

As proved in [5], $v \sim v'$ implies that, for any $\phi \in \Phi(X)$ with constants less than or equal to C_A, $v \models \phi$ iff $v' \models \phi$.

With $[v]$ we denote the equivalence class $\{v' \,|\, v \sim v'\}$. The set of equivalence classes $V = \{[v] \mid v \text{ is a valuation}\}$ is finite, and with $|V|$ we denote its cardinality.

We also recall the definition of *clock zones* (for more details see [11] and [32]).

The set of clock zones on X (denoted with $\Psi(X)$) is the set of formulae ψ such that

$$\psi ::= true \,|\, false \,|\, x \sim c \,|\, x - y \sim c \,|\, \psi_1 \wedge \psi_2 \,|\, \psi_1 \vee \psi_2 \,|\, \neg\psi$$

where $\sim \{<, \leq, >, \geq, =\}$, $c \in \mathbb{N}$ and $x, y \in X$.

Let A be a PTA with states in Q and clocks in X; a *region* R of A is a pair (q, ψ) where $q \in Q$ and $\psi \in \Psi(X)$.

2.1.3 Weak Bisimulation for PTAs

Bisimilarity is widely accepted as the finest extensional behavioural equivalence one would want to impose on systems, and it may be used to verify a property of a system by assessing the bisimilarity of the considered system, with a system one knows to enjoy the property.

The *bisimulation* of a system by another system is based on the idea of mutual step-by-step simulation. Intuitively, two systems A and A' are bisimilar if whenever one of the two systems executes a certain action and reaches a configuration s, the other system is able to simulate this single step by executing the same action and reaching a configuration s' which is again bisimilar to s. A *weak bisimulation* is a bisimulation that does not take into account τ (internal) moves. Hence, whenever a system simulates an action of the other system, it can also execute some internal τ actions before and after the execution of that action.

In order to abstract away from τ moves, Milner [43] introduces the notion of observable step, which consists of a single *visible* action α preceded and followed by an arbitrary number (including zero) of internal moves. Such moves are described by a *weak* transition relation $\Longrightarrow$, defined as $\stackrel{\alpha}{\Longrightarrow} = (\xrightarrow{\tau})^* \xrightarrow{\alpha} (\xrightarrow{\tau})^*$, where $\longrightarrow$ is the classical strong relation, and $\stackrel{\tau}{\Longrightarrow} = (\xrightarrow{\tau})^*$. It is worth noting that with such a definition a weak internal

transition $\stackrel{\tau}{\Longrightarrow}$ is possible even without performing any internal action.

For the definition of weak bisimulation in the probabilistic setting, Baier and Hermanns [6] replace Milner's weak internal transitions $q \stackrel{\tau}{\Longrightarrow} q'$ by the probability $Prob(q, \tau^*, q')$ of reaching state q' from q via internal moves. Similarly, for visible actions α, Baier and Hermanns define $q \stackrel{a}{\Longrightarrow} q'$ by means of the probability $Prob(q, \tau^*\alpha\tau^*, q')$. Thus, we give a definition of weak bisimulation for PTAs which is inspired by [6].

Definition 2.3: Let $A = (\Sigma, X, Q, q_0, Inv, \delta, \Pi)$ be a PTA. A *weak bisimulation* on A is an equivalence relation $\mathcal{R}$ on $\mathcal{S}_A$ such that, for all schedulers F and $(s, s') \in \mathcal{R}$, there exists a scheduler F' such that for all $\mathcal{C} \in \mathcal{S}_A/\mathcal{R}$ and $\alpha \in \Sigma \cup \{\tau\} \cup \mathbb{R}^{>0}$:

$$Prob_A^F(s, \tau^*\hat{\alpha}\tau^*, \mathcal{C}) = Prob_A^{F'}(s', \tau^*\hat{\alpha}\tau^*, \mathcal{C})$$

and vice versa.
Two configurations s, s' are called *weakly bisimilar* on A (denoted $s \approx_A s'$) iff $(s, s') \in \mathcal{R}$ for some weak bisimulation $\mathcal{R}$.

In order to define weak bisimulation among PTAs, we resort to the notion of a disjoint sum of automata.

Definition 2.4: Let $A = (\Sigma, X, Q, q_0, Inv, \delta, \Pi)$ and $A' = (\Sigma', X', Q', q_0', Inv', \delta', \Pi')$ such that $Q \cap Q' = \emptyset$ and $X \cap X' = \emptyset$. Let $\hat{A} = (\Sigma \cup \Sigma', X \cup X', Q \cup Q' \cup \{\hat{q}\}, \hat{q}, \hat{Inv}, \delta \cup \delta' \cup \{(\hat{q}, \tau, true, \emptyset, q_0), (\hat{q}, \tau, true, \emptyset, q_0')\}, \hat{\Pi})$, where, $\hat{\pi_1}, \hat{\pi_2} \in \hat{\Pi}$ such that $\hat{\pi_1}(e) = 1$ if $e = (\hat{q}, \tau, true, \emptyset, q_0)$, 0 otherwise, and $\hat{\pi_2}(e) = 1$ if $e = (\hat{q}, \tau, true, \emptyset, q_0')$, 0 otherwise. Moreover, for each couple $(\pi, \pi') \in \Pi \times \Pi'$, $\hat{\pi} \in \hat{\Pi}$ such that:

$$\hat{\pi}(e) = \begin{cases} \pi(e) & if\ e \in \delta \\ \pi'(e) & if\ e \in \delta'. \end{cases}$$

The invariant conditions of $\hat{A}$ are given by:

$$\hat{Inv}(q) = \begin{cases} Inv(q) & if\ q \in Q \\ Inv'(q) & if\ q \in Q' \end{cases}$$

We say that A and A' are weakly bisimilar (denoted by $A \approx A'$) if $(q_0, \mathbf{0}) \approx_{\hat{A}} (q_0', \mathbf{0})$, where the valuation $\mathbf{0}$ is defined over all clocks of the set $X \cup X'$.

For deciding our notion of weak bisimulation, we follow the classical approach of refining relations between configurations ([6], [13]). In particular, starting from the initial relation where all configurations of a PTA are equivalent, we stepwise specialize relations until we obtain a weak bisimulation.

At each step we refine the set of classes by deleting the relations between configurations s^i and s^j which do not satisfy the condition that, for all schedulers F, there exists a scheduler F' such that $Prob_A^F(s^i, \tau^*\alpha, \mathcal{C}) = Prob_A^{F'}(s^j, \tau^*\alpha, \mathcal{C})$ and vice versa.

An algorithm for deciding weak bisimulation for PTAs can be found in [38]. The complexity of deciding weak bisimulation grows exponentially [38]. As we are dealing with timed regions, such a complexity bound could not be improved. The automatic analysis of complex systems may thus became difficult, however, reasonable abstractions of realistic systems could still be modelled and analysed in our framework.

Proposition 2.5: It is decidable whether two PTAs are weakly bisimilar.

Example 2.6: Consider the PTAs of Figure 2. Intuitively, they both can perform action a or action b before 5 time units, with equal probability $\frac{1}{2}$. By applying the notion of weak bisimulation introduced above, the two PTAs turn out to be equivalent. Let $\hat{A}$ be the automaton built from the automata A_1 and A_2 by following the procedure described in Definition 2.4.

We call π_1 the only probability distribution of A_1 and π_2 the only probability distribution of A_2.

With $\mathcal{R}$ we denote the equivalence relation on $\mathcal{S}_{\hat{A}}$ such that $((q, v), (r, v')) \in \mathcal{R}$ if one of the following requirements holds:

- $q = q_0$, $r = r_0$ and $0 \leq v(x) = v'(z) \leq 5$.
- $q = q_0$, $r = r_0$ and $v(x) = v'(z) > 5$.
- $q \in \{q_1, q_2\}$ and $r \in \{r_1, r_2\}$ and $v(x) = v'(z)$.

Note that the case $x \neq z$ is not considered since no reachable configuration allows this case.

The set of classes has infinite cardinality. Actually, each value in $[0, 5]$ generates a class. Hence, for any $m \in [0, 5]$, we call $\mathcal{C}_m$ the class composed by the two configurations $\{(q_0, x = m), (r_0, z = m)\}$. As we will see, to solve the problem of the infiniteness of classes, the algorithm we propose groups the set of $\{\mathcal{C}_m\}_{m \in [0,5]}$ into the triple $(q_0, r_0, x \leq 5 \wedge x = z)$.

With $\mathcal{C}$ we denote the set of classes containing each configuration (q_0, v) and (r_0, v) such that $v(x) = v(z) > 5$. This set of classes can be represented by the triple $(q_0, r_0, x = z \wedge x > 5)$. Finally, with $\mathcal{C}'$ we denote the class containing each configuration (q, v) such that $q \in \{q_1, q_2\}$ and (r, v) such that $r \in \{r_1, r_2\}$ and $v(x) = v(z)$.

In the following we assume $\alpha \in \{a, b, \tau\} \cup \mathbb{R}^{>0}$, where α is chosen according to a scheduler F and a distribution π_i.

We consider the case in which the configuration is in a state of A_2, the other case is easier since from q_0 there is no τ transition.

Given $s = (r_0, v) \in \mathcal{C}_m$ and $s' = (q_0, v') \in \mathcal{C}_m$ and a scheduler F, we have the following cases:

- if $Prob^F(s, \tau^*, \mathcal{C}_m) = 1$, then $Prob^{F'}(s', \tau^*, \mathcal{C}_m) = 1$, for any scheduler F'. Namely, $\epsilon \in \tau^*$.
- if $Prob^F(s, \tau^*\alpha\tau^*, \mathcal{C}_{m+\alpha}) = 1$ where $\alpha \in \mathbb{R}^{>0}$ and $\alpha \leq 5 - m$, then $Prob^{F'}(s', \tau^*\alpha\tau^*, \mathcal{C}_{m+\alpha}) = 1$ where $F'(s') = \alpha$.
- if $Prob^F(s, \tau^*\alpha\tau^*, \mathcal{C}) = 1$ where $\alpha \in \mathbb{R}^{>0}$ and $\alpha > 5 - x$, then $Prob^{F'}(s', \tau^*\alpha\tau^*, \mathcal{C}) = 1$ where $F'(s') = \alpha$.
- if $Prob^F(s, \tau^*\alpha\tau^*, \mathcal{C}') = \frac{1}{2}$ where $\alpha \in \{a, b\}$, then $Prob^{F'}(s', \tau^*\alpha\tau^*, \mathcal{C}') = \frac{1}{2}$ where $F'(s') = \pi_1$.

Moreover, for each $s \in \mathcal{C}$ we have that if $Prob^F(s, \tau^*\alpha\tau^*, \mathcal{C}) = 1$ where $\alpha \in \mathbb{R}^{>0}$, then $Prob^{F'}(s', \tau^*\alpha\tau^*, \mathcal{C}) = 1$ where $F'(s') = \alpha$. Similarly for each $s \in \mathcal{C}'$.

The probability of any other case we have not considered here, is 0 for any scheduler F. In this case, in order

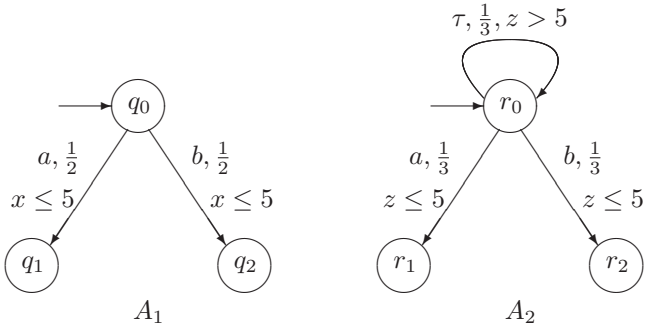


Fig. 2. $A_1 \approx A_2$.

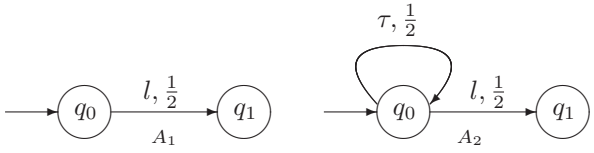


Fig. 3. $A_1 \not\approx A_2$.

to show the weak bisimulation, it has been sufficient to consider $F' = F$.

Now, $\mathcal{R}$ is a weak bisimulation on $\hat{A}$ and, since $(q_0, \mathbf{0})$ and $(r_0, \mathbf{0})$ are in the same class, A_1 and A_2 are weakly bisimilar.

The next example shows a subtle feature of weak bisimulation for PTAs. Namely, internal actions, even if not visible, may alter the probability of observing the passage of time.

Example 2.7: Consider the PTAs of Figure 3. Intuitively, they both, eventually, perform action l with probability 1 and then reach a terminal configuration. At a first glance the two automata appear to be bisimilar, however, the internal move in A_2 allows a scheduler to make different the probability of observing passage of time with respect to A_1. Namely, A_1 and A_2 are not bisimilar because there exists no scheduler F of A_1 able to simulate the behaviour induced by the following scheduler F':

$$
\begin{array}{ll}
F'(\sigma_0) = \pi & \sigma_0 = (r_0, 0) \\
F'(\sigma_1) = 1 & \sigma_1 = (r_0, 0) \xrightarrow{\tau} (r_0, 0) \\
F'(\sigma_2) = \pi & \sigma_2 = (r_0, 0) \xrightarrow{\tau} (r_0, 0) \xrightarrow{1} (r_0, 1) \\
\ldots &
\end{array}
$$

where π is the only probability distribution of A_2 depicted in Figure 3.

If $\mathcal{C}$ is the class containing the configuration $(r_0, 1)$ we have that $Prob^{F'}_{A_2}((r_0, 0), \tau^* 1 \tau^*, \mathcal{C}) = \frac{1}{2}$, and no scheduler exists for A_1 with the same property.

We would like to stress the fact that this problem arises since the internal action in A_2 is probabilistically in competition with the observable action l. If we replace the probabilistic distribution π in A_2 with two distributions π_1 and π_2 assigning, respectively, probability 1 to each of the two transitions, the two PTAs turn out to be weakly bisimilar.

2.1.4 Auxiliary Operators for PTAs

We define operations of *restriction*, *hiding* and *parallel composition* on PTAs.

We assume a PTA $A = (\Sigma, X, Q, q_0, \delta, Inv, \Pi)$ and a set $L \subseteq \Sigma$ of actions.

Definition 2.8: The *restriction* of a PTA A with respect to the set of actions L is $A \setminus L = (\Sigma, X, Q, q_0, \delta', Inv, \Pi')$, where:

- $\delta' = \{(q, a, \phi, B, q') \in \delta \mid a \notin L\}$.
- $\pi' \in \Pi'$ iff $\pi \in \Pi$ where, for all $e = (q, a, \phi, B, q') \in \delta$, $\pi'(e) = \frac{\pi(e)}{\sum_{e' \in \delta' \cap start(q)} \pi(e')}$.

The second condition is assumed in order to normalize the probability of each transition according to the ones remaining after the restriction. Thanks to this rule, the condition $\sum_{e \in start(q)} \pi'(e) \in \{0, 1\}$ continues to be true for each state q of $A \setminus L$.

Definition 2.9: The *hiding* of a transition $e = (q, a, \phi, B, q')$ with respect to the set of actions L (written e/L) is defined as:

$$
e/L = \begin{cases} e & if\ a \notin L \\ (q, \tau, \phi, B, q') & if\ a \in L \end{cases}
$$

The hiding of a PTA A with respect to the set of actions L is given by $A/L = (\Sigma, X, Q, q_0, \delta', Inv, \Pi')$, where $\delta' = \{e/L \mid e \in \delta\}$, and $\Pi' = \{\pi' | \exists \pi \in \Pi. \forall e' \in \delta'\ \pi'(e') = \sum_{e \in \delta : e/L = e'} \pi(e)\}$.

Proposition 2.10: Given a PTA A, $A \setminus L$ and A/L are PTAs for all $L \subseteq \Sigma$.

We assume two PTAs $A_1 = (\Sigma, X_1, Q_1, r_0, \delta_1, Inv_1, \Pi_1)$ and $A_2 = (\Sigma, X_2, Q_2, u_0, \delta_2, Inv_2, \Pi_2)$ with disjoint sets of states and clocks ($Q_1 \cap Q_2 = \emptyset$, $X_1 \cap X_2 = \emptyset$). We also assume a set $L \subseteq \Sigma$ of synchronization actions. Finally, for $i \in \{1, 2\}$, given a transition $e = (q, a, \phi, B, q') \in \delta_i$, and a probability distribution $\pi_i \in \Pi_i$ with $\pi_{i_a}(e)$ we denote the normalized probability of executing transition e with respect to all other transitions starting from q and labelled with a, i.e. $\pi_{i_a}(e) = \frac{\pi_i(e)}{\sum_{e' \in start^a_i(q)} \pi_i(e')}$, where $start^a_i(q)$ denotes the set of transitions in δ_i with q as source state and a as labelling action, i.e. the set $\{(q_1, a', \phi, B, q_2) \in \delta_i \mid q_1 = q \wedge a' = a\}$.

Definition 2.11: The *parallel composition* of two PTAs A_1 and A_2, with respect to the synchronization set L, is defined as $A_1 ||_L A_2 = (\Sigma, X, Q, (r_0, u_0), \delta, Inv, \Pi)$. The set $Q = Q_1 \times Q_2$ of states of $A_1 ||_L A_2$ is given by the cartesian product of the states of the two automata A_1 and A_2, while the set of clocks $X = X_1 \cup X_2$ is given by the union of X_1 and X_2. State invariants are defined as $Inv : Q_1 \times Q_2 \to \Phi(X_1 \cup X_2)$, where, for any $r \in Q_1$ and $u \in Q_2$ such that $Inv_1(r) = \phi_1$ and $Inv_2(u) = \phi_2$, we have that $Inv(r, u) = \phi_1 \wedge \phi_2$. Given a state (r, u) of $A_1 ||_L A_2$ there is a probability distribution $\pi \in \Pi$ for any two probability distributions $\pi_1 \in \Pi_1$ and $\pi_2 \in \Pi_2$. In particular, $\delta = S_1 \cup S_2 \cup \bigcup_{a \in L} S^a_3$ where $S_1 = \{((r, u), b, \phi, B, (r', u)) \mid (r, b, \phi, B, r') \in \delta_1, u \in Q_2, b \notin L\}$, $S_2 = \{((r, u), b, \phi, B, (r, u')) \mid (u, b, \phi, B, u') \in$

$\delta_2, r \in Q_1, b \notin L\}$ and, for any $a \in L$, $S_3^a = \{e = ((r,u), \tau, \phi_1 \wedge \phi_2, B_1 \cup B_2, (r',u')) \mid e_1 = (r, a, \phi_1, B_1, r') \in \delta_1, e_2 = (u, a, \phi_2, B_2, u') \in \delta_2\}$. Moreover, for any pair $\pi_1 \in \Pi_1$, $\pi_2 \in \Pi_2$, there exists $\pi \in \Pi$ such that, for all $e = (q, a, \phi, B, q') \in \delta$, it holds that $\pi(e) = \frac{f(e)}{\sum_{e' \in \delta \cap start(q)} f(e')}$ where $f(e) = \pi_1(e)$ if $e \in S_1$, $f(e) = \pi_2(e)$ if $e \in S_2$ and $f(e) = \pi_1(e_1) \cdot \pi_{2_a}(e_2) + \pi_2(e_2) \cdot \pi_{1_a}(e_1)$ for $e \in \bigcup_{a \in L} S_3^a$ obtained from e_1 and e_2.

Note that, given such a definition of parallel composition, A_1 and A_2 are forced to synchronize in order to perform transitions labelled with actions in L, moreover, whenever they synchronize, they give rise to an internal action τ. Also note that, once chosen a transition e_1 (e_2) with label $a \in L$ of automaton A_1 (A_2), the transition e_2 (e_1) of A_2 (A_1) that synchronizes with e_1 (e_2) is chosen according to the probability $\pi_{2_a}(e_2)$ ($\pi_{1_a}(e_1)$) normalized with respect to all the other transitions labelled with a. Besides, according to Definition 2.1, given the parallel composition defined above, it holds that $\sum_{e \in start(q)} \pi(e) \in \{0, 1\}$ for each state q of $A_1||_L A_2$. This is done due to the last rule, that uses the auxiliary structure $f(e)$ in order to compute the normalized probabilities in π. In fact, transitions of the single automata A_1 and A_2 with label $a \in L$ are not allowed to be performed without synchronization, and therefore they are lost in the compound system together with their probabilities (and therefore probabilities of the compound system must be renormalized).

Proposition 2.12: Given the PTAs A_1 and A_2, the parallel composition $A_1||_L A_2$ is a PTA for all $L \subseteq \Sigma$.

2.1.5 Probabilistic Automata

We introduce Probabilistic Automata as a subcase of PTAs.

Definition 2.13: A *Probabilistic Automaton* (PA) is a PTA $A = (\Sigma, X, Q, q_0, \delta, Inv, \Pi)$, where $X = \emptyset$, $\phi = true$ for every $e = (q, a, \phi, B, q') \in \delta$ and $Inv(q) = true$ for every state $q \in Q$.

As $X = \emptyset$, there are no valuations of clocks, and therefore a configuration reduces to a state. Transitions and state invariants of a PA may have as a constraint only the condition $true$. Moreover, since for PAs we abstract from time, we assume that for each execution σ of a PA there is no scheduler $F \in \mathcal{F}_A$ such that $F(\sigma) \in \mathbb{R}^{>0}$.

2.1.6 From PTAs to PAs

Given a PTA $A = (\Sigma, X, Q, q_0, \delta, Inv, \Pi)$, we call $untime(A)$ the PA obtained as the *region automaton* of A, with probability functions chosen according to Π. Intuitively, the region automaton (see [5]) is obtained by considering timed regions as states. Note that in the region automaton there is a step between regions R and R' with symbol (a, π) if and only if there is an admissible run $s \xrightarrow{t} s'' \xrightarrow{(a,\pi)} s'$ of the PTA such that $t \in \mathbb{R}^{>0}$ and where $s \in R$ and $s' \in R'$. We consider the special symbol λ to label all the transitions of the PA $untime(A)$ arising from timed steps of the PTA A. More precisely, $untime(A) = (\Sigma \cup \{\lambda\}, \emptyset, Q \times [V], (q_0, [v_0]), \delta', Inv', \Pi')$ where:

- $((q,[v]), \lambda, true, \emptyset, (q', [v'])) \in \delta'$ iff $q = q'$, $v' = v + t$ for some time $t \in \mathbb{R}^{>0}$ and $v + t' \models Inv(q)\ \forall t' \in [0, t]$;
- $((q,[v]), a, true, \emptyset, (q', [v'])) \in \delta'$ iff $(q, a, \phi, B, q') \in \delta$, $v \models \phi \wedge Inv(q)$, $v' = v[B]$ and $v' \models Inv(q')$;
- $Inv'(q, [v]) = true\ \ \forall (q, [v]) \in Q \times [V]$;
- For any λ-transition $e' = ((q,[v]), \lambda, true, \emptyset, (q', [v']))$ there exists $\pi' \in \Pi'$ such that $\pi'(e') = 1$. Moreover, for all $\pi \in \Pi$ there exists $\pi' \in \Pi'$ such that, $\pi'(e') = \sum_{e \in S} \pi(e)$ where $e' = ((q,[v]), a, true, \emptyset, (q', [v'])) \in \delta'$ and $S = \{(q, a, \phi, B, q') \in \delta \mid v \models \phi,\ \ v' = v[B]\}$.

Given an execution $\sigma = (q_0, v_0) \rightarrow \ldots \rightarrow (q_n, v_n)$ of A, with $[\sigma]$ we denote the execution $(q_0, [v_0]) \rightarrow \ldots \rightarrow (q_n, [v_n])$ of $untime(A)$.

As a consequence of Lemma 4.13 in [5] and Lemma 4.8 in [36] we have the following result.

Lemma 2.14: Given a PTA A, for any scheduler F of A, there exists a scheduler F' of $untime(A)$ such that, for any $\sigma \in ExecFrag^F$, $Prob^F(\sigma) = Prob^{F'}([\sigma])$, and viceversa.

2.2 Timed Automata

We recall the definition of Timed Automata ([5]).

Definition 2.15: A *Timed Automaton* (TA) is a tuple $A = (\Sigma, X, Q, q_0, \delta, Inv)$, where Σ, X, Q, q_0, δ and Inv are defined as in Definition 2.1.

As for PTAs, a *configuration* of A is a pair (q, v), where $q \in Q$ is a state of A, and v is a valuation over X. The initial configuration of A is represented by $(q_0, \mathbf{0})$ and the set of all the configurations of A is denoted with $\mathcal{S}_A$.

There is a discrete *transition step* from a configuration $s_i = (q_i, v_i)$ to a configuration $s_j = (q_j, v_j)$ through action $a \in \Sigma \cup \{\tau\}$, written $s_i \xrightarrow{a} s_j$, if there is a transition $e = (q_i, a, \phi, B, q_j) \in \delta$ such that $v_i \models \phi \wedge Inv(q_i)$, $v_j = v_i[B]$ and $v_j \models Inv(q_j)$.

Continuous timed steps are as for PTAs, and execution fragments of a TA are, again, finite sequences of steps.

2.2.1 Weak Bisimulation for TAs

Definition 2.16: Let $A = (\Sigma, X, Q, q_0, \delta, Inv)$ be a TA. A *weak bisimulation* on A is an equivalence relation $\mathcal{R} \subseteq \mathcal{S}_\mathcal{A} \times \mathcal{S}_\mathcal{A}$ such that for all $(s, r) \in \mathcal{R}$ it holds that $\forall \alpha \in \Sigma \cup \{\tau\} \cup \mathbb{R}^{>0}$:

- if $s \xrightarrow{\alpha} s'$, then there exists r' such that $r \xRightarrow{\alpha} r'$ and $(s', r') \in \mathcal{R}$;
- conversely, if $r \xrightarrow{\alpha} r'$, then there exists s' such that $s \xRightarrow{\alpha} s'$ and $(s', r') \in \mathcal{R}$.

Two configurations s, r are called *weakly bisimilar* on A (denoted $s \approx_A r$) iff $(s, r) \in \mathcal{R}$ for some weak bisimulation $\mathcal{R}$.

Two TAs $A = (\Sigma, X, Q, q_0, \delta, Inv)$ and $A' = (\Sigma', X', Q', q_0', \delta', Inv')$ such that $Q \cap Q' = \emptyset$ and $X \cap X' = \emptyset$ are called weakly bisimilar (denoted by $A \approx A'$) if, given the TA $\hat{A} = (\Sigma \cup \Sigma', X \cup X', Q \cup Q', q_0, \delta \cup \delta', \hat{Inv})$,

it holds $(q_0, \mathbf{0}) \approx_{\hat{A}} (q_0', \mathbf{0})$, where $\hat{Inv}$ is defined as in Definition 2.4.

The following result is proved in [39] and [14].

Proposition 2.17: It is decidable whether two TAs are weakly bisimilar.

In [45], [40], Wang and Larsen introduce strong and weak bisimulation equivalences for a real time process calculus obtained by extending Milner's CCS [43] with delays. Our notion of weak bisimulation for PTAs differs form the mentioned one. In our framework, a weak transition $\stackrel{t}{\Longrightarrow}$ is given by a sequence of internal moves followed by a single time transition of duration t followed by a sequence of internal moves. In [45] both time delays and internal moves are interchangeable, the observable resulting after a series of time delays and internal moves is just the time delay obtained summing up the different timed steps: a weak timed transition consists of a sequence of steps (either internal or timed steps) such that the sum of the different time steps is t. Thus, the main difference is that our notion of weak bisimulation is *weak* only with respect to internal moves, while the notion of weak bisimulation in [45] is *weak* with respect to both internal and timed steps.

In [45] also a notion of strong bisimulation is given (which is similar to our notion when abstracting from internal moves). In a sense, we relax this equivalence by introducing internal invisible moves, which then lead to the definition of our *intermediate* weak bisimulation.

As a consequence, we are treating as *visible* consequent timed steps. Such an assumption makes finer the classes of our weak bisimulation relation but allows us to take into account the fact that the scheduler is invoked again after a timed step[2]. Such an invocation would actually make distinguishable, to the eyes of a malicious scheduler, a system which can perform a single timed step of length t from a system which can perform two timed steps (maybe interleaved by some internal move) with length t_1 and t_2 such that $t_1 = t_2$, and so alter its observable behaviour (see [16], [17] for examples on how malicious schedulers may collaborate with an attacker allowing him to distinguish two bisimilar processes)[3]. We believe that in the context of security analysis our stricter notion of bisimulation is more suitable than the one presented in [45].

2.2.2 Auxiliary Operators for TAs

We define operations of *restriction*, *hiding* and parallel composition on TAs.

We assume a TA $A = (\Sigma, X, Q, q_0, \delta, Inv)$ and a set $L \subseteq \Sigma$ of actions.

Definition 2.18: The *restriction* of a TA A with respect to the set of actions L is $A \setminus L = (\Sigma, X, Q, q_0, \delta', Inv)$, where $\delta' = \{(q, a, \phi, B, q') \in \delta \mid a \notin L\}$.

Definition 2.19: The *hiding* of a TA A with respect to the set of actions L is given by $A/L = (\Sigma, X, Q, q_0, \delta', Inv)$, where $\delta' = \{e/L \mid e \in \delta\}$.

Proposition 2.20: Given a TA A, $A \setminus L$ and A/L are TAs for all $L \subseteq \Sigma$.

We assume two TAs $A_1 = (\Sigma, X_1, Q_1, r_0, \delta_1, Inv_1)$ and $A_2 = (\Sigma, X_2, Q_2, u_0, \delta_2, Inv_2)$ with $Q_1 \cap Q_2 = \emptyset$ and $X_1 \cap X_2 = \emptyset$.

Definition 2.21: The *parallel composition* of two TAs A_1 and A_2, with respect to the synchronization set L, is defined as $A_1 || _L A_2 = (\Sigma, X, Q, (r_0, u_0), \delta, Inv)$, where $Q = Q_1 \times Q_2$ and $X = X_1 \cup X_2$. State invariants are defined as $Inv : Q_1 \times Q_2 \to \Phi(X_1 \cup X_2)$, where, for any $r \in Q_1$ and $u \in Q_2$ such that $Inv_1(r) = \phi_1$ and $Inv_2(u) = \phi_2$, we have that $Inv(r, u) = \phi_1 \wedge \phi_2$. The set of transitions δ is defined as follows:

- if from state r the automaton A_1 has a transition $e_1 = (r, a, \phi, B, r')$ with $a \notin L$, then $e = ((r, u), a, \phi, B, (r', u)) \in \delta$;
- if from state u the automaton A_2 has a transition $e_2 = (u, a, \phi, B, u')$ with $a \notin L$, then $e = ((r, u), a, \phi, B, (r, u')) \in \delta$;
- if from state r the automaton A_1 has a transition $e_1 = (r, a, \phi_1, B_1, r')$ with $a \in L$ and from state u the automaton A_2 has a transition $e_2 = (u, a, \phi_2, B_2, u')$, then A_1 and A_2 can synchronize and $e = ((r, u), \tau, \phi_1 \wedge \phi_2, B_1 \cup B_2, (r', u')) \in \delta$.

Proposition 2.22: Given the TAs A_1 and A_2, $A_1 || _L A_2$ is a TA for all $L \subseteq \Sigma$.

2.2.3 From PTAs to TAs

Given a PTA $A = (\Sigma, X, Q, q_0, \delta, Inv, \Pi)$, $unprob(A) = (\Sigma, X, Q, q_0, \delta, Inv)$ gives the TA obtained from A by simply removing Π.

2.2.4 Nondeterministic Systems

We introduce Nondeterministic Systems as a subcase of TAs.

Definition 2.23: A *Nondeterministic System* (NS) is a TA $A = (\Sigma, X, Q, q_0, \delta, Inv)$, where $X = \emptyset$, $\phi = true$ for every $e = (q, a, \phi, B, q') \in \delta$ and $Inv(q) = true$ for every state $q \in Q$.

As $X = \emptyset$, there are no valuations of clocks, and therefore a configuration reduces to a state. Transitions and state invariants of a NS may have only the condition $true$ as a constraint. The class of NSs coincides with the class of Nondeterministic Automata.

Operators and bisimulation defined for TAs reduce in the subcase of NSs to the analogous operators and bisimulation defined in [26].

2.2.5 From TAs and PAs to NSs

Given a TA $A = (\Sigma, X, Q, q_0, \delta, Inv)$, we call $untime(A)$ the NS obtained as the *region automaton* of A and by considering an empty set of clocks X. Note that in the region automaton there is a step between regions R and R' with symbol a if and only if there is an admissible run

2. No assumption about the scheduler is needed in [45]; since this model is purely possibilistic, nondeterminism could be treated in the classical way. Here we used schedulers to combine probabilities and nondeterminism.

3. Note that in our framework the scheduler also decides the amount of time to pass.

$s \xrightarrow{t} s'' \xrightarrow{a} s'$ of the TA such that $t \in \mathbb{R}^{>0}$ and where $s \in R$ and $s' \in R'$. We consider the special symbol λ to label all the transitions of the NS $untime(A)$ arising from timed steps of the TA A. More precisely, $untime(A) = (\Sigma \cup \{\lambda\}, \emptyset, Q \times [V], (q_0, [v_0]), \delta', Inv')$ where:

- $((q,[v]), \lambda, true, \emptyset, (q',[v'])) \in \delta'$ iff $q = q'$, $v' = v + t$ for some time $t \in \mathbb{R}^{>0}$ and $v+t' \models Inv(q)\ \forall t' \in [0,t]$;
- $((q,[v]), a, true, \emptyset, (q',[v'])) \in \delta'$ iff $(q, a, \phi, B, q') \in \delta$, $v \models \phi \wedge Inv(q)$, $v' = v[B]$ and $v' \models Inv(q')$;
- $Inv'(q,[v]) = true\ \ \forall (q,[v]) \in Q \times [V]$.

Given an execution $\sigma = (q_0, v_0) \rightarrow \ldots \rightarrow (q_n, v_n)$ of A, with $[\sigma]$ we denote the execution $(q_0, [v_0]) \rightarrow \ldots \rightarrow (q_n, [v_n])$ of $untime(A)$.

As a consequence of Lemma 4.13 in [5] we have the following result.

Lemma 2.24: Given a TA A, if σ is an execution fragment of A, then $[\sigma]$ is an execution fragment of $untime(A)$. Moreover, if $[\sigma]$ is an execution fragment of $untime(A)$, then there exists $\sigma' \in [\sigma]$ such that σ' is an execution fragment of A.

Given a PA $A = (\Sigma, \emptyset, Q, q_0, \delta, Inv, \Pi)$, $unprob(A) = (\Sigma, \emptyset, Q, q_0, \delta, Inv)$ gives the NS obtained from A by removing Π.

2.3 Relations among Weak Bisimulations

We shall use the same terminology for operators and bisimulation in the different models when this does not give rise to ambiguity.

Lemma 2.25: The following statements hold:

1) PTAs A,A': $A \approx A' \Rightarrow unprob(A) \approx unprob(A')$
2) PTAs A,A': $A \approx A' \Rightarrow untime(A) \approx untime(A')$
3) PAs A,A': $A \approx A' \Rightarrow unprob(A) \approx unprob(A')$
4) TAs A,A': $A \approx A' \Rightarrow untime(A) \approx untime(A')$.

Proof: For cases (1) and (3), let us assume $A = (\Sigma, X, Q, q_0, \delta, Inv, \Pi)$, $A' = (\Sigma', X', Q', q_0', \delta', Inv, \Pi')$ and $\hat{A}$ constructed as in Definition 2.4. Since $A \approx A'$ for a weak bisimulation $\mathcal{R}$, we have that for all $(s,r) \in \mathcal{R}$, $\mathcal{C} \in \mathcal{S}_{\hat{A}}/\mathcal{R}$ and schedulers F, there exists a scheduler F' such that $Prob^F_{\hat{A}}(s, \tau^*\alpha, \mathcal{C}) = Prob^{F'}_{\hat{A}}(r, \tau^*\alpha, \mathcal{C})$ for every $\alpha \in \Sigma \cup \{\tau\} \cup \mathbb{R}^{>0}$. Now, if $Prob^F_{\hat{A}}(s, \alpha, s') > 0$ for some $s' \in \mathcal{C}$ there exists a configuration r' and a scheduler F' such that $Prob^{F'}_{\hat{A}}(r, \tau^*\alpha, r') = Prob^F_{\hat{A}}(s, \alpha, s') > 0$. Therefore if $s \xrightarrow{\alpha} s'$, then there exists r' such that $r \stackrel{\alpha}{\Longrightarrow} r'$ and, since s' and r' are in the same equivalence class, there exists also a bisimulation $\mathcal{R}'$ on $\mathcal{S}_{\hat{A}_{np}}$ such that $(s', r') \in \mathcal{R}'$, where $\hat{A}_{np}$ is constructed as in Definition 2.16 by starting from $unprob(A)$ and $unprob(A')$. The same holds if we exchange the roles of s and r.

For cases (2) and (4), the implications hold by the construction of the region automaton and by Lemmata 2.14 and 2.24. Actually, for each run of a PTA (or TA), there exists an analogous run for the PA (or NS) obtained with $untime(A)$. Weak bisimulations are, therefore, preserved. □

3 Security Properties

Given a system model with the basic operators of restriction, hiding and parallel composition together with a notion of observational equivalence, it is easy to set up a framework for the analysis of information flow.

In all of the formalisms presented in Section 2, a finite alphabet Σ of visible actions is assumed. A multilevel system interacts with agents confined in different levels of clearance. In order to analyze the information flow between parties with different levels of confidentiality, the set of visible actions is partitioned into high level actions and low level actions. Formally, we assume the set of possible actions $\Sigma = \Sigma_H \cup \Sigma_L$, with $\Sigma_H \cap \Sigma_L = \emptyset$. In the following, with $l, l' \ldots$ and $h, h', \ldots$ we denote actions of Σ_L and Σ_H respectively. With Γ_H and Γ_L we denote the set of high level agents and low level agents. Formally, an automaton A with a set of action labels Σ' is in Γ_H (Γ_L) if $\Sigma' \subseteq \Sigma_H$ ($\Sigma' \subseteq \Sigma_L$). For simplicity, we specify only two-level systems; note, however, that this is not a real limitation, since it is always possible to deal with the case of more levels by iteratively grouping them into two clusters.

A low level agent is able to observe the execution of all the steps labeled with actions in Σ_L and all the timed steps. The basic idea of Non-Interference is that the high level does not interfere with the low level if the effects of high level communications are not visible by a low level agent. Finally, an important assumption when dealing with Non-Interference analysis is that a system is considered to be *secure* (no information flow can occur) if there is no interaction with high level agents (if high level actions are prevented).

Other properties have been introduced in the literature in order to capture different behaviour of systems that have to be considered not secure. In [26] Focardi and Gorrieri promote the classification of a set of properties capturing the idea of information flow and Non-Interference. One of the most interesting and intuitive security properties is *Non Deducibility on Composition* (*NDC*), which states that a system A in isolation has not to be altered when considering all the potential interactions of A with the high level agents of the external environment.

3.1 Non-Interference

We define Non-Interference properties, *Probabilistic Timed Non-Interference* (*PTNI*), *Probabilistic Non-Interference* (*PNI*), *Timed Non-Interference* (*TNI*) and *Nondeterministic Non-Interference* (*NNI*).

Definition 3.1: Given a system A in PTAs (PAs, TAs, NSs, resp.) A is *PTNI* (*PNI*, *TNI*, *NNI*, resp.)-secure if and only if $A/\Sigma_H \approx A \setminus \Sigma_H$. We write $A \in PTNI$ ($A \in PNI$, $A \in TNI$, $A \in NNI$, resp.) when the system A is *PTNI* (*PNI*, *TNI*, *NNI*, resp.)-secure.

In the definition above, $A \setminus \Sigma_H$ represents the isolated system, where all high level actions are prevented. As we have seen, such a system is considered secure due to

the notion of Non-Interference. If the observational behaviour of the isolated system is equal to the behaviour of A/Σ_H, which represents the system that communicates with high level agents in an invisible manner for the low agents point of view, A satisfies the security property.

Note that the *PNI* property is the *BSPNI* property defined in [2], the *TNI* property is an analogous of the *tBSNNI* property defined in [28], and *NNI* is the *BSNNI* property of [26].

The *PTNI* property, defined in an environment where both probability and time are studied, is able to detect information flow that may occur either due to the probabilistic behaviour of the system or due to the time when an action occurs or due to a combination of them.

Proposition 3.2: It is decidable whether a PTA (PA, TA, NS, resp.) A satisfies the *PTNI* (*PNI*, *TNI*, *NNI*, resp.) property.

Proof: The result derives directly by the decidability of weak bisimulation for all the models, and by the computable definitions of the operators of hiding and restriction. □

The security properties defined in the probabilistic and/or timed settings are conservative extensions of the security properties defined in the possibilistic and/or untimed settings.

Proposition 3.3: The following implications hold:

- $A \in PNI \Rightarrow unprob(A) \in NNI$.
- $A \in TNI \Rightarrow untime(A) \in NNI$.
- $A \in PTNI \Rightarrow unprob(A) \in TNI \wedge untime(A) \in PNI$.

Proof: The implications follow by the bisimulation based definitions of the security properties and by the conservativeness of the notions of weak bisimulation (Lemma 2.25). Consider a PA A. If $A \in PNI$, by Lemma 2.25 we have that $unprob(A/\Sigma_H) \approx unprob(A \setminus \Sigma_H)$. Now, by definitions of $unprob$, hiding and restriction, it is easy to see that $unprob(A/\Sigma_H) = unprob(A)/\Sigma_H$ and that $unprob(A \setminus \Sigma_H) = unprob(A) \setminus \Sigma_H$. Therefore, we have that $unprob(A)/\Sigma_H \approx unprob(A) \setminus \Sigma_H$, proving that $unprob(A) \in NNI$. The proof is similar for the other cases. □

The converse implications do not hold. The integration of probability and time adds new information that extends what is already known in the nondeterministic case. Therefore, systems considered to be secure in a purely possibilistic setting, may turn out to be insecure when considering aspects either of probability or of time. This is shown in Examples 3.4 and 3.5.

Example 3.4: In Figure 4 we show a case of probabilistic information flow presented in [2]. We assume $Inv(q_i) = true$ for every $i \in [0,6]$. Abstracting away from probability, the system A could be considered secure. In a purely possibilistic setting, in both systems $unprob(A)/\Sigma_H$ and $unprob(A)\setminus\Sigma_H$ a low level agent can observe the action l or the sequence ll' without further information about the execution of action h. It holds that $unprob(A)/\Sigma_H \approx unprob(A) \setminus \Sigma_H$ and, therefore, $unprob(A) \in NNI$ and $unprob(A) \in TNI$. In a probabilistic framework, given $\mathbf{p} + \mathbf{r} + \mathbf{q} = 1$, the high level action h interferes with the probability of observing either a single action l or the sequence ll'. Formally, in $A \setminus \Sigma_H$, a low level agent observes either the single action l with probability $\mathbf{p} + \mathbf{r}$ or the sequence ll' with probability $\mathbf{q}$. However, in A/Σ_H the single event l is observed with probability $\mathbf{p}$ and the sequence ll' with probability $\mathbf{r}+\mathbf{q}$. As a consequence we have $A/\Sigma_H \not\approx A \setminus \Sigma_H$, so that the *PNI* and the *PTNI* properties reveal the probabilistic covert channel.

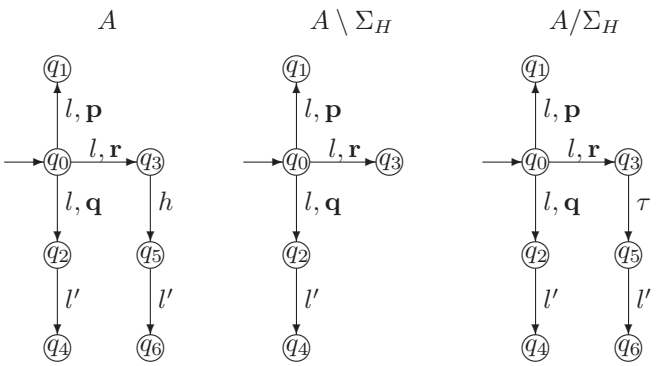


Fig. 4. A probabilistic covert channel.

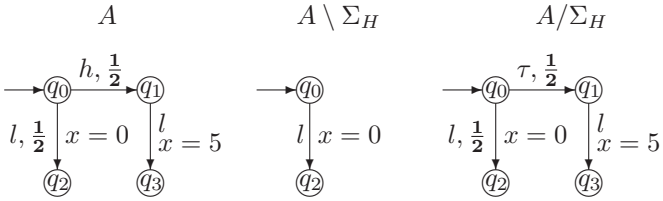


Fig. 5. A timing covert channel.

Example 3.5: In Figure 5 we show a case of timing information flow. We assume $Inv(q_i) = true$ for $i \in [2,3]$ and $Inv(q_i) = x \leq 5$ for $i \in [0,1]$. Abstracting away from time, the system A could be considered secure. In an untimed setting, in both systems $untime(A)/\Sigma_H$ and $untime(A) \setminus \Sigma_H$ a low level agent can observe only the action l executed with probability 1 without further information about the execution of action h. It holds that $untime(A)/\Sigma_H \approx untime(A) \setminus \Sigma_H$, and, therefore, $untime(A) \in PNI$. In a timed framework, given a clock $x \in \mathbb{R}^{\geq 0}$, the high level action h interferes with the time of observing the action l. Formally, in $A\setminus\Sigma_H$, a low level agent observes the single action l executed immediately. However, in A/Σ_H the single event l could either be observed immediately or when the clock x reaches value 5. A low level agent, observing the event l when clock x has value 5 knows that action h has occurred. As a consequence, we have $A/\Sigma_H \not\approx A\setminus\Sigma_H$, so that the *PTNI* property reveals the timing covert channel. The same holds for $unprob(A)$; in this case the covert channel is detected by the *TNI* property.

For system A in Figure 4, $untime(A)$ is not *PNI*, but $unprob(A) \in$ *TNI*. On the contrary, for system A in Figure 5, $unprob(A)$ is not *TNI*, but $untime(A) \in$ *PNI*. This shows that the discerning power of time and probability, as regards the Non-Interference property, are incomparable as stated in the next proposition.

Proposition 3.6: The following implications hold:

- $\exists\ PTA\ A:\ unprob(A) \in TNI \wedge untime(A) \notin PNI$;
- $\exists\ PTA\ A\ : untime(A) \in PNI \wedge unprob(A) \notin TNI$.

If we can express both time and probability as in PTAs, we are able to describe systems exhibiting information flow that neither a formalism with only probability nor a formalism with only time can express. For such systems we are able to show that they are not *PTNI*, even if they are both *PNI* and *TNI*, and therefore we are able to reveal a new covert channel.

Proposition 3.7: There exists a PTA A such that A is not *PTNI* while $unprob(A)$ is *TNI* and $untime(A)$ is *PNI*.

Example 3.8: Consider the PTA A in Figure 6. We assume $Inv(q_i) = true$ for $i \in \{3,4,8,9\}$, $Inv(q_i) = x \leq 3$ for $i \in \{0,1,5,7\}$ and $Inv(q_i) = x \leq 4$ for $i \in \{2,6\}$. It is easy to see that $untime(A) \in PTI$. In both $untime(A)/\Sigma_H$ and $untime(A) \setminus \Sigma_H$, a low level agent observes the single event l taken with probability 1, and therefore $untime(A)/\Sigma_H \approx untime(A) \setminus \Sigma_H$. It is also easy to see that $unprob(A) \in TNI$. In both $unprob(A)/\Sigma_H$ and $unprob(A) \setminus \Sigma_H$, a low level agent could either observe the single event l taken when $x = 3$ or the event l taken when $x = 4$, and therefore $unprob(A)/\Sigma_H \approx unprob(A) \setminus \Sigma_H$. Finally, we show that A is not *PTNI*. In a probabilistic and timed framework, the high level action h interferes with the probability of observing the action l executed either when $x = 3$ or when $x = 4$. Formally, in $A \setminus \Sigma_H$, a low level agent observes the action l either when $x = 3$ or when $x = 4$ with probability $\frac{1}{2}$, respectively. However, in A/Σ_H the event l taken when $x = 3$ is observed with probability $\frac{19}{30}$, while the action l taken when $x = 4$ is observed with probability $\frac{11}{30}$. As a consequence, we have $A/\Sigma_H \not\approx A \setminus \Sigma_H$, so that the *PTNI* property reveals the probabilistic timing covert channel.

There are other attacks on probabilistic timed systems which do not arise in systems where only time or only probability is taken into account. Let A be the PTA in Figure 7. It is easy to see that $unprob(A)$ is *TNI* and $untime(A)$ is *PNI*, as in a purely timed system or in a purely probabilistic system, action h cannot alter the low level observation. As for PTAs, we have however that A is not *PTNI*. For a probabilistic timed system, in fact, a scheduler may exploit the probabilistic execution of h to alter the observation of the passage of time (see Example 2.7). Namely, we have $A/\Sigma_H = A_2$ and $A \setminus \Sigma_H = A_1$ where A_1 and A_2 are the PTAs in Figure 3.

3.2 Non Deducibility on Composition

We define the Non Deducibility on Composition properties *Probabilistic Timed Non Deducibility on Composition* (*PTNDC*), *Probabilistic Non Deducibility on Composition* (*PNDC*), *Timed Non Deducibility on Composition* (*TNDC*) and *Nondeterministic Non Deducibility on Composition* (*NNDC*).

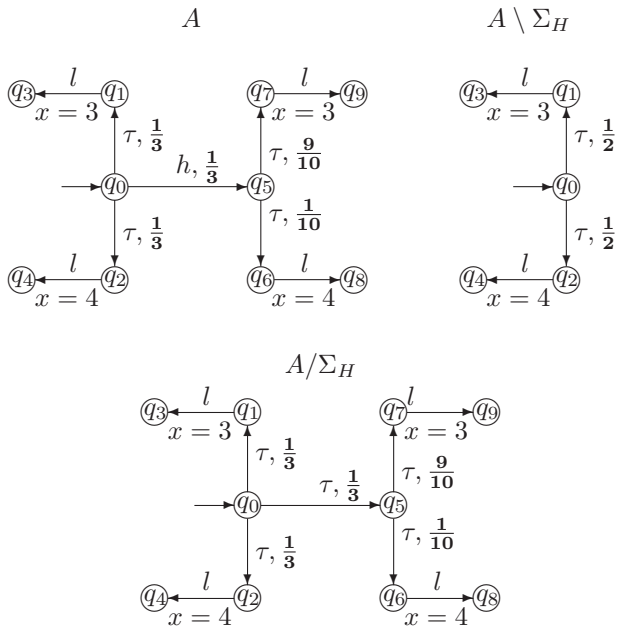


Fig. 6. A probabilistic timing covert channel.

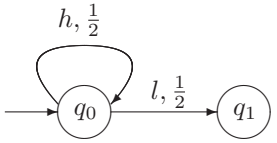


Fig. 7. Yet another probabilistic timed attack.

Definition 3.9: Given a system A in PTAs (PAs, TAs, NSs, resp.), A is *PTNDC* (*PNDC*, *TNDC*, *NNDC*, resp.)-secure if and only if $\forall E \in \Gamma_H, \forall L \subseteq \Sigma_H \quad A/\Sigma_H \approx (A||_L E) \setminus \Sigma_H$. We write $A \in PTNDC$ ($A \in PNDC$, $A \in TNDC$, $A \in NNDC$, resp.) when the system A is *PTNDC* (*PNDC*, *TNDC*, *NNDC*, resp.)-secure.

As we have seen, A/Σ_H represents the observable behaviour of A from a low level agent point of view (i.e. the isolated system where all high level actions are hidden). System $(A||_L E) \setminus \Sigma_H$ represents, instead, system A communicating with the high agent E and then prevented from executing other high level actions. If the observational behaviour of the isolated system is equal to the behaviour of the system communicating with any high level agent, A satisfies the security property.

Theorem 3.10: $A \in PTNDC$ (*PNDC*, *TNDC*, *NNDC*, resp.) $\Rightarrow A \in PTNI$ (*PNI*, *TNI*, *NNI*, resp.).

Proof: If A is a PTA, consider $E = (\emptyset, \emptyset, \{q\}, q, \emptyset, Inv(q) = false, \Pi) \in \Gamma_H$, while if A is a TA consider $E = (\emptyset, \emptyset, \{q\}, q, \emptyset, Inv(q) = false) \in \Gamma_H$. Intuitively, E is an automaton representing a high level agent which does not perform any transition. Consider then the set $L = \emptyset$. If system A is *PTNDC* (*PNDC*, *TNDC*, *NNDC*, resp.), then $A/\Sigma_H \approx (A||_L E) \setminus \Sigma_H$. Now,

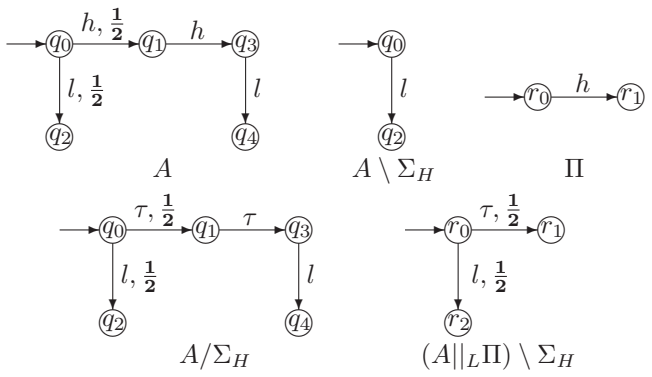


Fig. 8. $A \in$ *PTNI*, but $A \notin$ *PTNDC*.

by the definition of parallel composition, $(A||_L E) = A'$ where A' is an automaton behaving like A with only different state names (i.e., the states of A paired with the only state q of E). Now it is easy to see that $A' = A$ after a renaming of the states (i.e. by renaming each state (q', q) of A' with q'). As a consequence we have that $(A||_L E) = A$ and, therefore, $A/\Sigma_H \approx A \setminus \Sigma_H$, stating that $A \in$ *PTNI* (*PNI*, *TNI*, *NNI*, resp.). The converse implication does not hold, as it is shown in Example 3.11. □

Example 3.11: Consider the PTA A of Figure 8. We assume $Inv(q_i) = (x = 0)$ for $i \in \{0, 1, 3\}$ and $Inv(q_i) = true$ for $i \in \{2, 4\}$. It is easy to see that A is *PTNI* secure, since $A/\Sigma_H \approx A \setminus \Sigma_H$. In both A/Σ_H and $A \setminus \Sigma_H$, a low level agent observes the single event l taken when $x = 0$ with probability 1. If we consider, instead, the high level agent Π of Figure 8 and the set $L = \{h\}$, we can observe that $A/\Sigma_H \not\approx (A||_L\Pi) \setminus \Sigma_H$. In fact, system A/Σ_H always performs action l with probability 1, while system $(A||_L\Pi) \setminus \Sigma_H$ may reach a deadlock state r_1 and does not perform any visible action with probability $\frac{1}{2}$ (as it turns out after the parallel composition of A and Π). As a consequence, automaton A is not *PTNDC* secure. We also have that $untime(A) \in$ *PNI* but $untime(A) \notin$ *PNDC*, $unprob(A)$ is *TNI* but not *TNDC*, and, finally, $unprob(untime(A)) \in$ *NNI*, but $unprob(untime(A)) \notin$ *NNDC*.

Theorem 3.10 and Example 3.11 show that the *PTNI* property is not able to detect some potential deadlocks due to high level activities, as put into evidence in [26] for the nondeterministic setting. For this reason we resort to the *PTNDC* property, which implies *PTNI*, in order to capture these finer undesirable behaviour.

Note that the PTAs in Figure 4 and Figure 5 are also not *PTNDC*; this can be proven by considering two simple classes of high level agents. For the first class we consider just a high level agent that may synchronize with the two PTAs by performing an action h, whereas for the second class we consider an inactive high level agent that does not perform action h (see proof of Theorem 3.10).

As we did for the Non-Interference security properties in the previous section, now we distinguish the discerning power of time and probability with the Non Deducibility on Composition properties.

Proposition 3.12: The following implications hold:

- $A \in$ *PNDC* $\Rightarrow unprob(A) \in$ *NNDC*;
- $A \in$ *TNDC* $\Rightarrow untime(A) \in$ *NNDC*;
- $A \in$ *PTNDC* $\Rightarrow unprob(A) \in$ *TNDC* $\wedge untime(A) \in$ *PNDC*.

Proof: As for Proposition 3.3, the implications follow by the bisimulation based definitions of the security properties and by the conservativeness of the notions of weak bisimulation. Consider again a PA A. If $A \in$ *PNDC*, by Lemma 2.25 we have that for all $E \in \Gamma_H$, $p \in]0, 1[$, $L \subseteq \Sigma_H$, $unprob(A/\Sigma_H) \approx unprob((A||_L E) \setminus \Sigma_H)$. Now, by definitions of $unprob$, hiding and restriction, it is easy to see that $unprob(A/\Sigma_H) = unprob(A)/\Sigma_H$ and that $unprob((A||_L E) \setminus \Sigma_H) = unprob(A||_L E) \setminus \Sigma_H$. Moreover, by definition of parallel composition, we have that $unprob(A||_L E) = unprob(A)||_L unprob(E)$. Therefore, we have that $unprob(A)/\Sigma_H \approx (unprob(A)||_L unprob(E)) \setminus \Sigma_H$. Now, since this condition holds for each PA $E \in \Gamma_H$ and since any NS E' may be derived by $unprob(E)$ for some PA E, we also have that for all NSs $E' \in \Gamma_H$, $unprob(A)/\Sigma_H \approx (unprob(A)||_L E') \setminus \Sigma_H$, thus proving that $unprob(A) \in$ *NNDC*. The proof is similar for the other cases. □

For system A in Figure 4, $untime(A)$ is not *PNDC*, but $unprob(A) \in$ *TNDC*. On the contrary, for system A in Figure 5, $unprob(A)$ is not *TNDC*, but $untime(A) \in$ *PNDC*. This shows that also for the Non Deducibility on Composition, time and probability add discerning power, as stated in the next proposition.

Proposition 3.13: The following implications hold:

- $\exists$ *PTA* A : $unprob(A) \in$ *TNDC* $\wedge$ $untime(A) \notin$ *PNDC*;
- $\exists$ *PTA* A : $untime(A) \in$ *PNDC* $\wedge$ $unprob(A) \notin$ *TNDC*.

If we can express both time and probability as in PTAs, we are able to describe systems exhibiting information flow that neither a formalism with only probability nor a formalism with only time can express. For such systems we are able to show that they are not *PTNDC*, even if they are both *PNDC* and *TNDC*, and therefore we are able to reveal a new covert channel.

Proposition 3.14: $\exists A$: $A \notin$ *PTNDC* $\wedge$ $unprob(A) \in$ *TNDC* $\wedge$ $untime(A) \in$ *PNDC*.

Proof: If we consider again the PTA A in Figure 6 we may show that $untime(A) \in$ *PNDC* and $unprob(A) \in$ *TNDC* by considering only two classes of high level systems (one containing high systems which may synchronize with A and one with systems which may not). But, as we have seen, $A \notin$ *PTNI*, therefore, for Theorem 3.10 we also have that $A \notin$ *PTNDC*. □

The diagram in Figure 9 summarizes our results.

It is worth noticing that, as it happens for the analogous properties defined in [26], [28], [2], the definition of the *PTNDC* property is difficult to be used in practice because of the universal quantification on the high

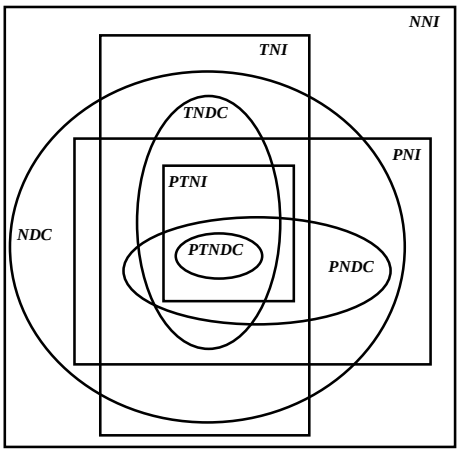


Fig. 9. Relations among security properties.

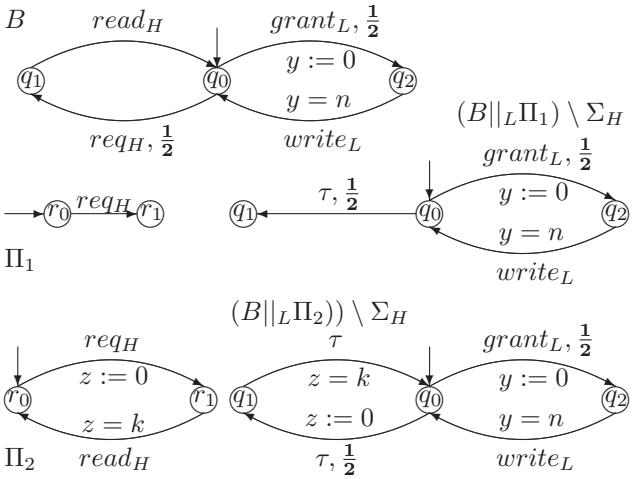


Fig. 10. Device specification with timing covert channels.

level agents. As we have seen, decidability of *PTNDC* depends, in fact, on the possibility of reducing all the high level automata in Γ_H to a finite case suitable for the particular automaton A we would like to study. In [26], [28], [2] other properties have been defined, stronger than the *PTNDC* property, which are easier to check. Such properties, defined for a CCS-like process algebra, discrete-time process algebra and probabilistic process algebra respectively, could be easily translated within our framework of PTAs.

4 An Application

As an application we consider a network device, also studied in [28] in a timed framework, that manages the access to a shared buffer by following a mutual exclusion policy. Assuming that the agents on the network are classified as low and high level agents, the device implements the *no-write-down no-read-up* policy [29]. Intuitively, the policy states that high level users can only read the buffer, while low level users can only write on it. Such a policy avoids direct information flow from high level to low level, however malicious agents can exploit some covert channel in order to transmit information indirectly. For example, a low level user could get information about the high level activity by observing the amount of time the device is locked (non accessible by the low level) when high agents are reading, or by observing the frequencies with which high level agents make access on it. We would like to check whether some covert channel can be exploited, by giving a specification of the network device and then checking the *PTNDC* property.

In the following we consider only a low level user and a high level user communicating with the network device. We assume that the low level user is always ready to write in the buffer, so we consider an agent that infinitely waits for a grant from the device and then writes in the buffer. In this manner we are considering a low level user that continuously monitors the activity of the device. We also assume that the entire procedure of receiving a grant in the network and writing in the buffer is executed in a time n. In Figure 10, we model the specification of a simple device (see the PTA B). Actions req_H, $read_H$, $grant_L$ and $write_L$ model high level read requests, high level reads, low level write grants and low level writes, respectively. The set Σ_H of high level actions is $\{req_H, read_H\}$. We assume $Inv(q_0) = Inv(q_1) = true$ and $Inv(q_2) = y \leq n$. The device B is always ready to accept an access request from the high level agent with probability $\frac{1}{2}$ and to grant a write access to the low level user with the same probability. Obviously, we always consider the device composed with a high level agent according to $||_L$ (we assume $L = \{req_H, read_H\}$). On the one hand, when the device is composed with a high level agent that performs action req_H with probability 1, it synchronizes with the high agent accepting his request with probability $\frac{1}{2}$. On the other hand, if the high level agent does not perform req_H, the composed system performs action $grant_L$ with probability 1. As a consequence, we can find out the following covert channels. Consider the high agent Π_1 of Figure 10, which executes a read request without performing the reading afterwards. System $(B||_L\Pi_1) \setminus \Sigma_H$ reaches a deadlock state that is not reached by B/Σ_H. In this way, the high level agent could transmit the bit 0 or 1 by alternatively blocking or not the device. Such a covert channel can be detected by the *PTNDC* property, in fact we have that $B/\Sigma_H \not\approx (B||_L\Pi_1) \setminus \Sigma_H$, so that $B \notin PTNDC$. Another interesting covert channel arises if one considers Π_2, which locks the buffer and executes a reading only after a time k. A low level user observing the behaviour of $(B||_L\Pi_2) \setminus \Sigma_H$ does not receive any grant access for a time k when a req_H action is performed. In this way the high level agent could indirectly transmit value k to the low level user. We obviously have again that $B/\Sigma_H \not\approx (B||_L\Pi_2) \setminus \Sigma_H$.

The two covert channels discussed above could be avoided by introducing a timeout mechanism which releases the device if $read_H$ is not performed and by always releasing the device after a fixed amount of time

has passed. In Figure 11 we show a device B' that accepts a high level request, and uses a clock x as timer and t as timeout. When it executes action req_H the timer is set to 0, action $read_H$ could be performed only when $x < t$, and when x reaches value t the device unlocks the buffer going back to q_0. When transitions starting from a given state have disjoint conditions we omit probabilities since their execution depends on the time configuration, rather than on the effective probability. We assume $Inv(q_0) = true$, $Inv(q_1) = Inv(q_3) = x \leq t$ and $Inv(q_2) = y \leq n$. The timing covert channels shown in the previous case could not be exploited anymore, however device B' is still insecure. In fact the device is unavailable for the fixed amount of time when a high level access is performed, and this is clearly observable by the low level user that has to wait the termination of the high level request before obtaining access to the buffer. This represents a typical situation where the unavailability of a shared resource can be encoded as 0 or 1 in order to transmit data. Such a situation is captured by the *PTNDC* property by considering again the automaton Π_2 and assuming $k < t$. In fact we have again that $B'/\Sigma_H \not\approx (B'||_L\Pi_2) \setminus \Sigma_H$.

The capacity of such a covert channel could be reduced, but not totally avoided, by considering a buffer that probabilistically locks himself without any high level request. In this manner the low level user could not be sure whether the buffer is really locked by the high user or not. In Figure 11, B'' represents a device that behaves in such a manner, locking himself with a probability $\mathbf{r}$. Again, $Inv(q_0) = true$, $Inv(q_2) = y \leq n$ and $Inv(q_1) = Inv(q_3) = Inv(q_4) = x \leq t$. As we have said, this does not avoid entirely the covert channel, but the knowledge the low level user acquires is affected by some uncertainty. In fact, if the device is locked, the low level user could deduce that the high user locked the device with a certain probability while with probability $\mathbf{r}$ the device has locked himself for simulating a false higher user's activity. In this case, if we resorted to a ε-tolerant weak bisimulation (i.e. for which the probabilities of reaching a certain class from certain configurations are allowed to be different up to the threshold ε [2]), we would be able to give a measure of the probabilistic covert channel, by estimating the probability that the information flow arises according to $\mathbf{r}$, and therefore a measure of the security level of the device.

We can completely hide the high level activity to the low level user by partitioning into two sessions the time in which users can access the buffer. During a *low session*, lasting a fixed amount of time n, only the low level user can access the buffer, then the device goes to the *high session*, where access is reserved, for the same amount of time, to the high level user. This makes impossible for the low level user to discover something about the high level activity, since the same fixed amount of time is reserved to the high session even if the high user does nothing. In Figure 12 we specify a buffer B_s that behaves

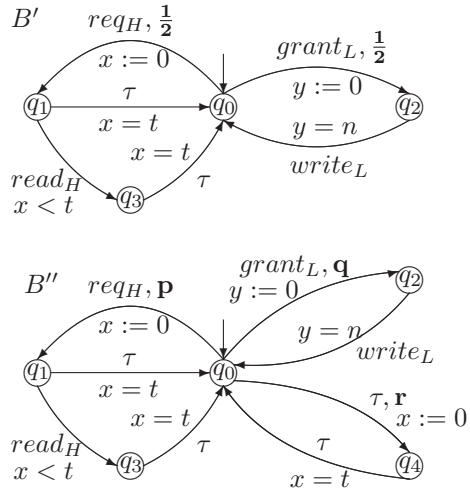


Fig. 11. Improved device specifications.

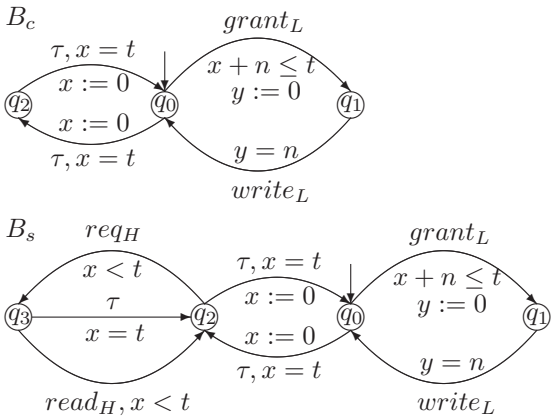


Fig. 12. Secure Device.

in such a manner: the buffer is reserved for a time t to the low level user and to the high level user alternatively. We assume $Inv(q_0) = Inv(q_2) = Inv(q_3) = x \leq t$ and $Inv(q_1) = y \leq n$. Automaton B_s is *PTNDC*, in fact, for every possible high level user Π, $(B_s||_L\Pi) \setminus \Sigma_H \approx B_c \approx B_s/\Sigma_H$. Intuitively, automaton B_c of Figure 12 represents the automaton resulting after the parallel composition between B_s and any high level user Π, and, therefore, B_c is weakly bisimilar to B_s composed with any possible high level user Π. Finally, it easy to see that $B_c \approx B_s/\Sigma_H$.

5 Conclusions

The classical theory of Non-Interference must be extended to cope with real systems which may exhibit probabilistic and timing covert channels that are not captured by non quantitative security models. In this paper we have developed a general framework where both probability and time are taken into account. By defining some information flow security property, we have shown how to detect with our model both probabilistic and timing covert channels.

Time and probability allow discovering that systems which in a nondeterministic setting are considered to be secure, are instead insecure. Time only and probability only give incomparable discerning power, while having both time and probability gives a discerning power greater than the ones given by each of them. We have seen in Figure 6 an example of a PTA where a covert channel arises by combining the probabilistic and the timed behaviour of a system. Such a covert channel could not be detected by analysing only the timing of the system or the frequencies of its actions. Such an academic example could work as a basis for more complex covert channels. In [38], a flow on the dining cryptographers protocol [18] has been detected as a combination of the probabilistic and the timed behaviour of the cryptographers. The anonymity property studied in this case could be reformulated in terms of a noninterference property and analysed in our setting.

In general, realistic probabilistic protocols do not always guarantee full security. Actually, one may usually assure that a property is satisfied up to a certain security threshold. The problem with this kind of properties is that one may end up classifying many realistic systems as insecure.

Moreover, in the recent literature (see, e.g., [16], [17]) the interest also moved on the analysis of the possible interactions of nondeterministic and probabilistic choices, and their possibility to affect the security of a system (examples on how nondeterminism does actually play a role in asserting the security of a system are given in the above mentioned papers). Actually, it has been shown that in certain cases, a scheduler that collaborates with an attacker can allow him to distinguish two bisimilar systems. In this cases, the solution is either to define stricter notions of behavioral equivalences (as in this paper and in [16]), or to limit the information a scheduler can access [17].

While our notion of weak bisimulation is extremely strict (in our case study, in order to get secure systems, we have to basically disallow any flow of information between levels and thus we partition the buffer in two different security clearance levels), there are also secure non pathological systems which are not partitioned between security levels (see, e.g., the dining cryptographers study in [38]). Note, however, that our notion could be made even stricter by following the approach in [16] and requiring two systems to simulate each other with the same scheduler.

On the other hand, the notion of weak bisimulation we introduced in Definition 2.3 is quite strict also because exact probabilities are required. Actually, an approximate relation could be extremely useful when analyzing security aspects of probabilistic systems. More precisely, within a purely qualitative setting one might establish whether a system is secure or not; in a probabilistic model, one can also verify the *security level* of a system by establishing with which probability certain insecure behaviour might arise (for example in the case of the buffer presented in Section 4 where a fake lock was considered).

In the classical approach to security analysis, some probabilistic protocols turn out to be insecure even though the probability of executing an insecure behaviour is close to 0. As a consequence, an approximate quantitative study of the unwanted behaviour is crucial for the evaluation of the security level of probabilistic systems.

In order to introduce a quantitative measure for insecure behaviour and to estimate the probability that a certain insecure behaviour arises, we could introduce an approximate relation for deciding if two systems behave *almost* in the same way (up to a given threshold). A similar approach is developed in [3].

In [12], [19], [21], [22], [23] metrics are introduced in order to quantify the similarity of the behaviour of probabilistic transition systems that are not strictly bisimilar. In [2] the authors introduce an enriched notion of weak probabilistic bisimulation, which is able to tolerate fluctuations making the security conditions less restrictive and relating systems that may have largely different possible behaviour under the condition that such behaviour is observable with a negligible probability. Other more information theoretical approaches also go in this direction, see, e.g., [15], [10].

Finally, another line of research to be carried out as a future extension of this work could consist in the development of a technique for the automatic correction of PTAs which do not satisfy a security property. Such a technique could be easier for the automatic correction of PTAs which do not satisfy the Non-interference security property. It should automatically adjust the automaton A in such a way that A/Σ_H becomes bisimilar to $A \setminus \Sigma_H$. In this direction, Agat considers the aspect of time in isolation for deterministic systems [1], Di Pierro et al. propose a framework where probabilistic systems are transformed to become time equivalent [20], finally, in [24] systems are transformed to become time probabilistic bisimilar.

REFERENCES

[1] J. Agat: *Transforming out Timing Leaks*. Proc. of POPL'00, ACM Press, 40–53, 2000.

[2] A. Aldini, M. Bravetti, R. Gorrieri: *A Process-algebraic Approach for the Analysis of Probabilistic Non-interference*. Journal of Computer Security 12, 191–245, 2004.

[3] A. Aldini, A. Di Pierro: *Estimating the Maximum Information Leakage*. International Journal of Information Security 7, 219–242, 2008.

[4] R. Alur, C. Courcoubetis, D. L. Dill: *Verifying Automata Specifications of Probabilistic Real-Time Systems*. Real-Time: Theory in Practice, Springer LNCS 600, 28–44, 1992.

[5] R. Alur, D. L. Dill: *A Theory of Timed Automata*. Theoretical Computer Science 126, 183–235, 1994.

[6] C. Baier, H. Hermanns: *Weak Bisimulation for Fully Probabilistic Processes*. Proc. of CAV'97, Springer LNCS 1254, 119–130, 1997.

[7] R. Barbuti, L. Tesei: *A Decidable Notion of Timed Non-interference*. Fundamenta Informaticae 54, 137–150, 2003.

[8] D. Beauquier: *On Probabilistic Timed Automata*. Theoretical Computer Science 292, 65–84, 2003.

[9] R. E. Bellman: *Dynamic Programming*. Princeton University Press, 1957.

[10] M. Boreale: *Quantifying Information Leakage in Process Calculi*. Information and Computation 207, 699–725, 2009.

[11] P. Bouyer: *Timed Automata May Cause Some Troubles*. BRICS RS-02-35.

[12] F. van Breugel, J. Worrel: *Towards Quantitative Verification of Probabilistic Systems (extended abstract)*. Proc. of ICALP'01, Springer LNCS 2076, 421–432, 2001.

[13] S. Cattani, R. Segala: *Decision Algorithm for Probabilistic Bisimulation*. Proc. of CONCUR '02, Springer LNCS 2421, 371–385, 2002.

[14] K. Cerans: *Decidability of Bisimulation Equivalences for Parallel Timer Processes*. Proc. of CAV'92, Springer LNCS 663, 302–315, 1992.

[15] D. Clark, S. Hunt, P. Malacaria: *Quantitative Information Flow, Relations and Polymorphic Types*. Journal of Logic and Computation 18, 181–199, 2005.

[16] K. Chatzikokolakis, G. Norman, D. Parker: *Bisimulation for Demonic Schedulers*. Proc. of FOSSACS'09, Springer LNCS 5504, 318–332, 2009.

[17] K. Chatzikokolakis, C. Palamidessi: *Making Random Choices Invisible to the Scheduler*. Proc. of CONCUR'07, Springer LNCS 4703, 42–58, 2007.

[18] D. Chaum: *The Dining Cryptographers Problem: Unconditional Sender and Recipient Untrace- ability*. Journal of Cryptology 1, 65-75, 1988.

[19] J. Desharnais, V. Gupta, R. Jagadeesan, P. Panangaden: *The Metric Analogue of Weak Bisimulation for Probabilistic Processes*. Proc. of LICS'02, IEEE CS Press, 2002.

[20] A. Di Pierro, C. Hankin, I. Siveroni, H. Wiklicky: *Tempus fugit: How to plug it*. Journal of Logic and Algebraic Programming 72, 173–190, 2007.

[21] A. Di Pierro, C. Hankin, H. Wiklicky: *Quantitative Relations and Approximate Process Equivalences*. Proc. of CONCUR'03, Springer LNCS 2761, 508–522, 2003.

[22] A. Di Pierro, C. Hankin, H. Wiklicky: *Approximate Non-Interference*. Journal of Computer Security 12, 37-82, 2004.

[23] A. Di Pierro, C. Hankin, H. Wiklicky: *Measuring the Confinement of Probabilistic Systems*. Theoretical Computer Science 340, 3–56, 2005.

[24] A. Di Pierro, C. Hankin, H. Wiklicky: *Quantifying Timing Leaks and Cost Optimisation*. Proc. of ICICS'08, Springer LNCS 5308, 81–96, 2008.

[25] N. Evans, S. Schneider: *Analysing Time Dependent Security Properties in CSP Using PVS*. Proc. of Symp. on Research in Computer Security, Springer LNCS 1895, 222–237, 2000.

[26] R. Focardi, R. Gorrieri: *A Classification of Security Properties*. Journal of Computer Security 3, 5–33, 1995.

[27] R. Focardi, R. Gorrieri, R. Lanotte, A. Maggiolo-Schettini, F. Martinelli, S. Tini, E. Tronci: *Formal Models of Timing Attacks on Web Privacy*. Elsevier ENTCS 62, 2001.

[28] R. Focardi, R. Gorrieri, F. Martinelli: *Information Flow Analysis in a Discrete-Time Process Algebra*. Proc. of 13th CSFW, IEEE CS Press, 170–184, 2000.

[29] J. A. Goguen, J. Meseguer: *Security Policy and Security Models*. Proc. of Symp. on Research in Security and Privacy, IEEE CS Press, 11–20, 1982.

[30] J. W. Gray III. *Toward a Mathematical Foundation for Information Flow Security*. Journal of Computer Security 1, 255–294, 1992.

[31] P. R. Halmos: *Measure Theory*. Springer-Verlag, 1950.

[32] T. .A Henzinger, X. Nicollin, J. Sifakis, S. Yovine: *Symbolic Model Checking for Real-time Systems*. Information and Computation **111**, 193–244, 1994.

[33] H. Howard: *Dynamic Programming and Markov Processes*. MIT Press, 1960.

[34] M. Kwiatkowska, G. Norman, R. Segala, J. Sproston: *Automatic Verification of Real-time Systems with Discrete Probability Distribution*. Theoretical Computer Science 282, 101–150, 2002.

[35] M. Kwiatkowska, R. Norman, J. Sproston: *Symbolic Model Checking of Probabilistic Timed Automata Using Backwards Reachability*. Tech. rep. CSR-03-10, University of Birmingham, 2003.

[36] R. Lanotte, D. Beauquier: *A Decidable Probability Logic for Timed Probabilistic Systems*. CoRR cs.LO/0411100 (2004).

[37] R. Lanotte, A. Maggiolo-Schettini, A. Troina: *Information Flow Analysis for Probabilistic Timed Automata*. Proc. of FAST'04, Springer IFIP 173, Toulouse, France, August 2004.

[38] R. Lanotte, A. Maggiolo-Schettini, A. Troina: *Weak Bisimulation for Probabilistic Timed Automata*. `http://www.di.unito.it/~troina/PTA_WB.pdf`.

[39] F. Laroussinie, K. G. Larsen, C. Weise: *From Timed Automata to Logic - and Back*. Proc. of MFCS'95, Springer LNCS 969, 27–41, 1995.

[40] K. G. Larsen, Y. Wang: *Time abstracted bisimulation: Implicit specifications and decidability*. Information and Computation 134, 75–101, 1997.

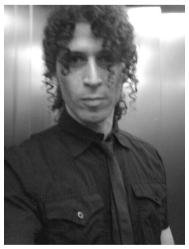
[41] D. McCullough: *Noninterference and the Composability of Security Properties*. Proc. of Symp. on Research in Security and Privacy, IEEE CS Press, 177–186, 1988.

[42] J. K. Millen: *Hookup Security for Synchronous Machines*. Proc. of CSFW'90, IEEE CS Press, 84–90, 1990.

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

[44] A. Troina: *Probabilistic Timed Automata for Security Analysis and Design*. Ph.D. Thesis, University of Pisa, 2006.

[45] Y. Wang: *Real–Time Behaviour of Asynchronous Agents*. Proc. of CONCUR'90, Springer LNCS 458, 502–520, 1990.



Ruggero Lanotte is Assistant Professor of Computer Science at the University of Insubria (Como) since 2003. Prior to his appointment in Como, he was affiliated with the University of Paris XII and the University of Pisa where he received a Ph.D. degree on Computer Science in 2003. His current research focuses on automaton-theoretic modelling of distributed and real-time systems. The most recent results regard the study of probabilistic behaviour of these systems and security analysis.

Andrea Maggiolo-Schettini is full professor of Computer Science at the University of Pisa since 1983. Prior to this appointment from 1968 to 1981 he was a researcher of the Italian National Research Council (C.N.R.) in Naples and in Pisa, and from 1981 to 1983 he was full professor of Computer Science at the University of Turin. He has done research in computability theory, semantics of programming languages, specification and verification of concurrent and distributed systems. His present interests include also systems biology and natural computing. He is member of the steering committee of the IEEE International Conference on Software Engineering and Formal Methods.

Angelo Troina is assistant professor of Computer Science at the University of Turin since October 2007. He got his Master degree in Computer Science at the University of Bologna in 2002 and obtained a Ph.D. in Computer Science at the University of Pisa in 2006. Then he moved to France for completing a postdoc at LIX (Ecole Polytechnique) and LSV (ENS-Cachan). His research topics include: formal description techniques of concurrent systems, modelling and verification of real-time and probabilistic systems, foundations of security analysis and systems biology.