Decidability of Parameterized Probabilistic Information Flow

Danièle Beauquier[1], Marie Duflot[1] and Yury Lifshits[2]

[1] University Paris 12, France
[2] Steklov Institute of Mathematics, St.Petersburg, Russia

Abstract. In this paper, we consider the decidability of two problems related to information flow in a system with respect to some property. A flow occurs in a system if the conditional probability of the property under some partial observation differs from the a priori probability of that property. For systems modelled as finite Markov chains we prove that the two following problems are decidable: does a system has information flow for a given regular property? is it true that the system has no information flow for any (sequential) property?

1 Introduction

In this paper we study the following *Security of Information Flow* problem: verify that no partial observation of a system behavior does leak an information that should be hidden.

Statement of the problem and our results. We use the framework of [15] and its formalization from [4]. In our trace-based approach, we assume a set of observable low-level events L and a set of (not directly observable) high-level events H. The question is whether observing a certain low-level trace can give information about the occurrence of high-level events in a probabilistic sense, yielding quantitative information about high-level activity. More precisely we propose a *parameterized* view of information flow. We define information flow with respect to a *property* (a set of system traces) which is deemed important for the system under scrutiny. This property is our *parameter* of the problem. The system has information flow with respect to the given property if there exist two low-level observations for which the chosen property has different probabilities of occurrence. In this case, the quantitative, probabilistic knowledge about the given property is sensitive to the observation which can be made, and so there is information flow in the system with respect to this property. It is worth mentioning that this probabilistic definition of information flow is related to Shannon's original definition of information, based on probabilities. In our previous paper [4] the formalization of information flow was presented for the first time together with necessary and sufficient conditions for having no information flow *for all properties* in a given system. Here, in order to get decidability results we restrict ourselves to systems modelled by finite Markov chains (with labelled edges) and to regular properties. It has a clear practical motivation. A field of application can be for example verification of security for parallel programming. Interleaving of actions of different threads is generally managed in a probabilistic way, and can be modelled as a Markov chain. As for security properties, many of them are regular.

Our first result states that it is decidable whether a system has information flow for a given property. The key ingredient of the algorithm is a trick from linear algebra, reformulating the notion of information flow as orthogonality of the set of vectors corresponding to all possible observations to some checking vector.

Our second result states that it is decidable whether a given system has no information flow *for all properties*. We consider two subcases: no information flow for any property and no information flow for sequential properties (those that do not consider the explicit value of low level actions).

Plan of the paper. Section 2 introduces the model under study and some related notations and definitions. In sections 3, 4 we present our decidability results just mentioned above. In Section 5 an example of application in the domain of concurrent programs illustrates how the interleaving of low-level actions can give probabilistic information on what happens at the high-level.

Related work. There is an important body of work in studying definitions related to information flow, for an overview see, e.g., [11]. We restrict the comparaison to the two features of our formalization: our notion of information flow is *parameterized* and it has *probabilistic* nature.

As far as we are aware of, only the paper of J. Halpern and K. O'Neill [8] parameterizes information flow by giving a definition of secrecy in multi-agent systems. They use a modal logic of knowledge in a state-based model as compared to our approach which is trace-based. Their framework generalizes several existing approaches and can be extended to probabilistic security. Their parametrization stems from defining formulas (knowledge) of what must be kept secret.

The other probabilistic approaches are more restrictive. McLean [11] introduces the *flow model* which distinguishes mere correlation from actual causal influence. Gray [6] introduces probabilistic *interference* in a context of finite state machines and gives a more general information-theoretic framework (as compared to [11]) including probabilistic channel capacity [7]. Sabelfeld and Sands [14] define probabilistic *noninterference* in the context of schedulers for multithreaded programs, based on the concept of probabilistic bisimulation. Lowe [10] treats quantitative information flow distinguishing probabilistic aspects from *nondeterminism*. A probabilistic process-algebraic approach is given in [1], focused on *noninterference*, generalizing the possibilistic variant and allowing formal reasoning about the amount of information flow. All these works are aimed at the definition of the models and do not deal with algorithmic problems.

Very few authors studied verification problems related to information flow. Among probabilistic approaches we can cite [5] that uses a process algebra formalism to study bisimulation-based security properties. Concerning probabilistic models, [13] gives a decidability result for "nondeducibility on composition" for probabilistic timed automata, and Gray [6] gives a sufficient condition for information flow security which *seems* decidable.

2 Probabilistic Event Systems

Notations. Given a finite alphabet A, A^* (resp. A^ω) denotes the set of finite (resp. infinite) sequences (or traces) over this alphabet. The set A^∞ is the union of A^* and A^ω. The empty sequence is denoted ε.
Given a sub-alphabet $A' \subset A$ and a trace λ, $\lambda_{|A'}$ denotes the projection of λ onto this sub-alphabet.

Let $u, v \in (A^*)^n$, $u = (x_1, x_2, \dots, x_n), v = (y_1, y_2, \dots, y_n)$. We denote by $u \otimes v$ the *simple interleaving* of u and v defined as $u \otimes v = x_1 y_1 x_2 y_2 \dots x_n y_n$. If $U, V \subset (A^*)^n$, we denote by $U \otimes V$ the set: $U \otimes V = \{u \otimes v | u \in U, v \in V\}$. If $U, V \subset (A^*)^\omega$, the definition of $U \otimes V$ is extended in a standard way.

Probabilistic Event Systems The behaviour of a probabilistic event system is modelled by its set Tr of traces which are finite or infinite sequences of atomic events from a set E. A particular atomic event τ is distinguished which represents the halting of the system. For example, if λ is a sequence of atomic events, it is useful to distinguish between "λ has occurred but the system is still in action", and "λ has occurred and the system stopped". The last case is modelled by the event $\lambda\tau$. In order to unify the presentation it is convenient to use only infinite sequences and thus we use $\lambda\tau^\omega$ instead of $\lambda\tau$. Then, from now on, Tr is a set of infinite sequences which either do not contain any occurrence of τ or of the form $\lambda\tau^\omega$ where λ does not contain any occurrence of τ.

In order to deal with information flow issues, the set of atomic events E is divided into two disjoint sets, the set H of high-level (*i.e.* secret) atomic events and the set L of low-level (*i.e.* public) ones.

The set of traces Tr is equipped with a probability measure μ over the σ-algebra generated by the cylinders αE^ω, such that Tr is μ-measurable. The measure $\mu(X)$ of a measurable set X is denoted as $Pr_\mu(X)$, or shortly $Pr(X)$. Thus if we consider the infinite tree $\mathcal{T}_S$ built from Tr with edges labelled by atomic events, each edge of the tree is equipped with a non-zero probability. (We assume that every prefix of a trace in Tr has a non-zero probability).

As usual, we introduce the following notation for conditional probabilities: if P and Q are two measurable events and $Pr(Q) \neq 0$, the conditional probability $Pr(P|Q)$ is equal to $Pr(P \cap Q)/Pr(Q)$. Since we are interested only in traces of the system $\mathcal{S}$ we will deal only with the conditional probabilities relative to Tr. Thus, for each measurable event X we denote by $Pr_S(X)$ the probability $Pr(X|\mathcal{S})$ ($Pr(\mathcal{S})$ is supposed to be positive).

Definition 1 *A* probabilistic event system *is a tuple* (E, H, L, Tr, μ) *where* $E = H \cup L$, $H \cap L = \emptyset$ *and* H *(resp.* L*) is the set of high-level (resp. low-level) actions,* μ *is a probabilistic measure on* E^ω *and* $Tr \subset E^\omega$*, the set of traces of the system, is* μ*-measurable.*

We assume that only low-level actions are observable on the low-level, i.e., for a trace λ the projection $\lambda_{|L}$ is observable by low-level users. More precisely, every finite prefix of $\lambda_{|L}$ is observable. Thus, from the observation of $u \in L^*$, the low-level user who is supposed to know the entire system can construct the *bunch* $B_S(u) = \{\lambda \in Tr \mid u \text{ is a prefix of } \lambda_{|L}\}$ and possibly deduce some information on what happened or what will happen. When there is no ambiguity, we will write $B(u)$ instead of $B_S(u)$.

A *property* is a subset of E^ω. From now on we consider only μ-measurable properties.

Definition 2 *A system* S *is* without information flow *for a property* P *if for every* $u, v \in L^*$ *such that* $B(u)$ *and* $B(v)$ *are non-empty,* $Pr_S(P|B(u)) = Pr_S(P|B(v))$.

The above definition means that, whatever the low level user observes, he does not get additional information on the probability of P to hold.

A particular case of interest is when only the presence of low level events is important for P, not their value. Such a property is called *sequential* and defined below.

Definition 3 *A property* P *is* sequential *if there exists* $P' \subseteq (H \cup \{l\})^\omega$ *such that* $P = \phi^{-1}(P')$ *where* ϕ *is a morphism which is identity on* H *and for each* $l_i \in L, \phi(l_i) = l$.

3 Decidability of Information Flow for a given property

We will now state some conditions under which one can decide whether a probabilistic event system has information flow under some property.

The most common probabilistic systems described in a finite way are Markov chains, and the simplest properties are regular ones, i.e. recognized by a deterministic Muller automaton. We recall below the definition of Markov chains [9](with a small change) and Muller automata [12].

Definition 4 *We call* Markov chain with labelled edges *a system $\mathcal{A} = (\Sigma, i, A, T)$ where S is a finite set of states, $i \in S$ is the initial state, A is a finite alphabet, $T : S \times A \times S \mapsto [0,1]$ is a function such that $\forall s \in S, \sum_{s' \in S, a \in A} T(s,a,s') = 1$ and for each $(s,a) \in S \times A$ there is at most one s' such that $T(s,a,s') > 0$*[3].

This system is slightly different from a classical Markov chain for which $T : S \times S \mapsto [0,1]$. Here there can be more than one edge between two states (if they have different labels). In order to get decidability results we suppose that T has its values in the set $\mathbb{Q}$ of rational numbers.

Let $\mathcal{P}_q$ be the set of paths from state q. The set $\mathcal{P}_i$ of infinite paths from the initial state i is equipped with a probabilistic measure μ in a standard way. A trace is the infinite sequence of labels of an infinite path.

Let Tr be the set of traces. The probability measure μ' on Tr is defined as follows: for every basic cylinder uA^ω, $\mu'(uA^\omega)$ is the mesure $\mu(w\mathcal{P}_q)$ where w is the path from i labelled with u and q is the last state of w.

Thus if A is partitioned into two sets of high-level and low-level actions, H and L, the Markov chain defines a probabilistic event system (A, H, L, Tr, μ').

Definition 5 *A* Muller automaton *is a tuple $\mathcal{M} = (Q, A, q_0, \Delta, \mathcal{F})$, where Q is the finite set of states, q_0 is the initial state, Δ is the set of transitions and $\mathcal{F}$ is the set of accepting subsets. An infinite word w is accepted by the automaton if the set of infinitely visited states along some*[4] *path with label w belongs to $\mathcal{F}$.*

It is a well known result that deterministic (complete) Muller automata have the same expressive power as nondeterministic ones [12].

Now that we have defined the type of systems we consider, we can state the main result.

Theorem 1 *Given a system $\mathcal{S}$ described by a Markov chain with labelled edges, and a regular property on infinite traces P given as a deterministic Muller automaton, we can decide whether the system $\mathcal{S}$ has information flow for the property P.*

[3] this last condition means that the underlying automaton (without the probabilities) is deterministic

[4] Such a path is unique in the case of a deterministic and complete automaton.

Proof: The full proof cannot fit in the page limitation and is given in appendix A. Our algorithm works as follows:

1. first we compute a composition of the Markov chain and Muller automaton
2. then we simplify it by the rule "$H^*l \to l$" obtaining one step matrices,
3. next we reformulate the information flow problem in a linear algebraic form, showing that it is equivalent to the orthogonality of a hull and a given "check vector",
4. and to conclude we prove that the hull mentioned above is computable.

□

4 Decidability of General Information Flow

Definition 6 *A system is* without (sequential) information flow *if it is without information flow for every (sequential) property.*

In the paper [4], such systems were characterized, *i.e.* necessary and sufficient conditions were given to ensure the absence of such information flow. We will show in this section that it is possible to decide whether a system is without (sequential) information flow when the considered system is a Markov chain with labelled edges.
The plan of this section is as follows:

– we first recall some definitions and notations necessary to state the criteria,
– then we recall the theorems from [4],
– and we conclude by proving that all these criteria are decidable for the models considered

In the following, low level actions are denoted $a, b, ...$, sequences of low-level actions $u, v, ...$, sequences of high-level actions $\alpha, \beta, ...$ and traces $\lambda, \lambda', ...$

Let $\mathcal{S} = (E, H, L, Tr, \mu)$ be a system, $\mathcal{T}$ be the associated probabilistic tree and $Pref(Tr)$ denote the set of finite prefixes of traces of Tr We define:
$H_n(Tr) = \{(\alpha_1, ..., \alpha_n) \in (H^*)^n \mid \exists\, a_1, ..., a_n \in L\ \alpha_1 a_1 ... \alpha_n a_n \in Pref(Tr)\}$.
$L_n(Tr) = \{(a_1, ..., a_n) \in L^n \mid \exists\, \alpha_1, \ldots, \alpha_n \in H^*\ \alpha_1 a_1 ... \alpha_n a_n \in Pref(Tr)\}$
$Tr_n = \{\alpha_1 a_1 \ldots \alpha_n a_n \in Pref(Tr) \mid \alpha_i \in H^*,\ a_i \in L\}$.

For the decidability proof given below, we need to introduce some technical terms related to the probabilistic tree $\mathcal{T}$.

We are interested in the set of sequences of high-level actions (including the empty word) which can occur starting from a node x. To make this set of sequences more explicit we build for each such node x a probabilistic tree $\mathcal{T}_x$ in the following way: we keep only the high edges reachable in $\mathcal{T}$ from x, and for each node y (including x) accessible from x by a high path with at least one low edge starting from y, we add a node y' and an edge (y, y') labelled by ε and with a probability equal to the sum p of the probabilities of low edges starting from y in $\mathcal{T}$. The tree $\mathcal{T}_x$ is a probabilistic tree which has the following meaning: the probability of a path in $\mathcal{T}_x$ starting from x labelled by α (without ε labels) is exactly the probability that the sequence of high-level actions α occurs from x; the probability of a path in $\mathcal{T}_x$ starting from x labelled by α and ending in a leaf is the probability that from x the sequence of actions α followed by a low-level action occurs.

A tuple (x, x', y, y') of nodes of the tree $\mathcal{T}$ is *H, L-compatible* if there exist $(\alpha_1, ..., \alpha_n), (\beta_1, ..., \beta_n) \in H_n(Tr)$, and $(a_1, ..., a_n), (b_1, ..., b_n) \in L_n(Tr)$ such that the paths from the root to x, x', y, y' are labelled respectively by $\alpha_1 a_1 ... \alpha_n a_n$, $\alpha_1 b_1 ... \alpha_n b_n$, $\beta_1 a_1 ... \beta_n a_n$ and $\beta_1 b_1 ... \beta_n b_n$.

Let $p_1, ..., p_n, q_1, ..., q_n$ be the probabilities of edges labelled by $a_1, ..., a_n$ on the path from the root to x (resp. y). Let $p'_1, ..., p'_n$, $q'_1, ..., q'_n$ be the probabilities of edges labelled by $b_1, ..., b_n$ on the path from the root to x' (resp. y').

An H, L-compatible tuple (x, x', y, y') is *perfect* if for every $i = 1, ..., n$ we have $p_i/q_i = p'_i/q'_i$.

Let us now rephrase theorems from [4].

Theorem 2 *A probabilistic system such that $Tr \not\subset H^\omega$ is*

1. *without information flow iff its projection on L is reduced to a single trace.*
2. *without sequential information flow iff*
   (1) $\forall n > 0 \ Tr_n = H_n(Tr) \otimes L_n(Tr)$.
   (2a) Every H, L-compatible tuple (x, x', y, y') of the tree $\mathcal{T}$ is perfect and
   (2b) the probabilistic trees $\mathcal{T}_x$ (resp.$\mathcal{T}_y$) and $\mathcal{T}_{x'}$ (resp.$\mathcal{T}_{y'}$) are isomorphic.
   (3) For every $n > 0$ $(L_n(Tr) \neq \emptyset \rightarrow \ Pr_S(Tr \cap (H^* L)^{n-1} H^\omega) = 0)$.

We can now state our decidability result:

Theorem 3 *It is possible to decide whether a system has (sequential) information flow or not.*

Proof: Due to the page limitation, the full proof of this theorem is given in appendix B. Here are the main ideas.

First for non sequential information flow the criterion is clearly decidable.
For point (1) of sequential information flow we compute the Muller automata corresponding to $\cup_{n\in\mathbb{N}} Tr_n$ and $\cup_{n\in\mathbb{N}}(H_n(Tr) \otimes L_n(Tr))$. Checking the equality of the two languages is decidable.
For point (2) we compute an automaton whose states are the quadruples of H, L-compatible states (states corresponding to some H, L-compatible nodes) and verify on the fly that the ratio on probabilities are preserved, and that the corresponding probabilistic trees are isomorphic.

For point (3) we compute ergodic sets containing only high-level actions and, if one exists, reason on the number of low-level actions necessary to reach this ergodic set. □

Here is an example of system without sequential information flow. In particular the ratio of probabilities of events l_1 and l_2 from state 2 to state 4 and from state 3 to state 5 is the same.

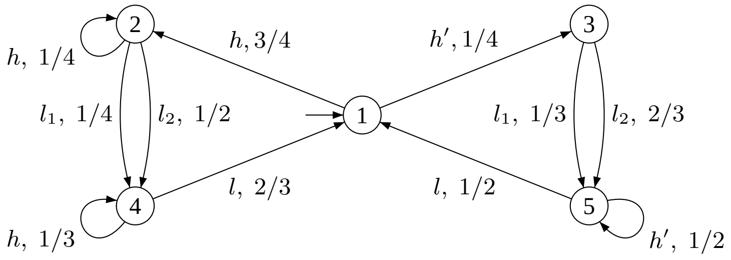

Fig. 1. A Markov chain without sequential information flow.

5 An example of application

Let us consider the problem of information flow for concurrent programs. The question is whether observing some values for its low variables we can conclude anything about high ones.

Consider the following multi-threaded program O inspired by [16]:

• Thread α:	• Thread β:
$h_0 := h_1;$	$h_0 := h_2;$
$l_0 := 1;$	$l_1 := 1;$

The low variables are l_0, l_1 initially equal to zero, h_0, h_1, h_2 are high variables. The content of h_1 and h_2 are different. Suppose that the two threads are scheduled probabilistically, with equal probabilities at each step for each thread to be run. The corresponding Markov chain is given in figure 1.

Each state contains the current state of threads (*i.e.* the set of instructions still to execute). For $i = 1, 2$, the label h_i means that the instruction $h_1 := h_i$ is executed, and the label l_i corresponds to the execution of $l_i := 1$. The actions of thread α (resp. β) correspond to the left (resp. right) edges.

For example state 5 corresponds to ($\alpha :\ l_0 := 1;\ \beta :\ l_1 := 1;$) and state 6 corresponds to ($\alpha : h_0 := h_1;\ l_0 := 1;$).

Suppose we are interested in the value of h_0 at the end of the program, more precisely in the property $P :\ h_0 = h_1$ after the execution of O. We can represent P as the language $(L \cup H \cup \perp)^* h_1 (L \cup \perp)^* \perp^\omega$. Indeed this regular

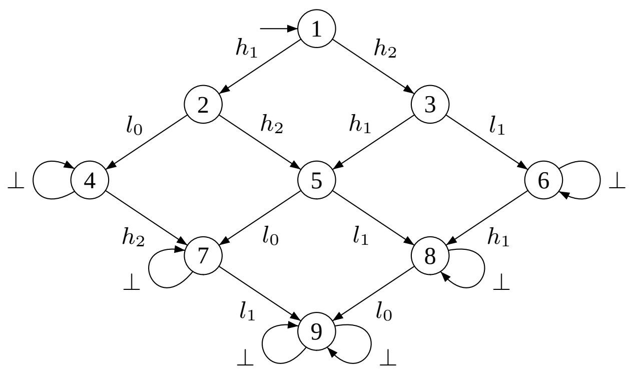

Fig. 2. The Markov chain associated to the program (each edge has a probability 1/2).

expression says that the last update of h_0 is $h_0 := h_1$. Notice that this property is sequential.

Clearly the probability of P is 1/2. But if we observe the low level, we have $Pr(P \mid l_0) = 1/4$. Thus the program has information flow for property P. In this particular case it means that, seeing the order in which low level variables are assigned, the low level user can gain (probabilistic) information on the order in which high level variables are assigned.

6 Conclusion and further work

This paper presents two decidability results for information flow when the system is a Markov chain with labelled edges and properties are regular. First we show that it is decidable whether a system has information flow for a specific regular property. Then we consider the decidability of absence of information flow for a class of properties and prove that the criteria given in [4] to ensure that the system has no information flow for two classes of properties, are decidable.

As it was noticed in the introduction, very few papers are devoted to algorithmic questions related to information flow. To our knowledge, our results are the first decidability ones for a probabilistic and parameterized model. This opens a way to quantitative evaluation of information flow.

Interesting open questions include: (1) computing quantitative estimations of information flow, (2) generalizing our algorithms to more expressive formalisms of system/property descriptions and (3) implementing and experimental testing of our algorithms in some applied domain.

References

1. A. Aldini, M. Bravetti, and R. Gorrieri. A process-algebraic approach for the analysis of probabilistic noninterference. *Journal of Computer Security*, 12:191–246, 2004.

2. R. Alur, C. Courcoubetis, and D. Dill. Verifying automata specifications of probabilistic real-time systems. In *Real Time: Theory in Practice*, volume 600 of *LNCS*, pages 28–44. Springer, 1992.
3. C. Courcoubetis and M. Yannakakis. The complexity of probabilistic verification. *Journal of the ACM*, 42:857–907, 1995.
4. M. Duflot D. Beauquier and M. Minea. A probabilistic property-specific approach to information flow. In *Proc. Int. Workshop on Mathematical Methods, Models, and Architectures for Computer Network Security (MMM-ACNS '05)*, volume 3685 of *LNCS*, pages 206–220. Springer, 2005.
5. R. Focardi and R. Gorrieri. The compositional security checker: A tool for the verification of information flow security properties. *Software Engineering*, 23(9):550–571, 1997.
6. J.W. Gray III. Probabilistic interference. In *Proc. IEEE Symp. on Security and Privacy*, pages 170–179, May 1990.
7. J.W. Gray III. Toward a mathematical foundation for information flow security. In *Proc. 1991 IEEE Symp. on Security and Privacy*, pages 21–35. IEEE Comp. Soc. Press, 1991.
8. J. Y. Halpern and K. R. O'Neill. Secrecy in multiagent systems. In *Proc. IEEE Computer Security Foundations Workshop*, pages 32–, 2002.
9. J. G. Kemeny and J. L. Snell. *Finite Markov Chains*. D Van Nostad Co., Inc., Princeton, N.J., 1960.
10. G. Lowe. Quantifying information flow. In *Proc. IEEE Computer Security Foundations Workshop*, pages 18–31, June 2002.
11. J. McLean. Security models and information flow. In *Proc. IEEE Symp. on Security and Privacy*, pages 180–187, May 1990.
12. D. Perrin and J.E. Pin. *Infinite words*, volume 141. Elsevier, 2004. Pure and Applied Mathematics.
13. A. Maggiolo-Schettini R. Lanotte and A. Troina. Information flow analysis for probabilistic timed automata. In *Workshop on Formal Aspects in Security and Trust (FAST)*, volume 12 of *IFIP International Federation for Information Processing*, pages 1–15, 2004.
14. A. Sabelfeld and D. Sands. Probabilistic noninterference for multi-threaded programs. In *Proc. IEEE Computer Security Foundations Workshop*, pages 200–214, July 2000.
15. A. Slissenko. Probability and time in measuring security. In F. L. Ţiplea E. Clarke, M. Minea, editor, *Proc. of NATO Adv. Research Work.:* Verification of Infinite-State Systems with Applications to Security (VISSAS'05), pages 169–183. IOS Press, 2006. NATO Security through Science Series. D : Information and Communication Security – Vol. 1.
16. D. Volpano and G. Smith. Probabilistic noninterference in a concurrent language. *Journal of Computer Security*, 7:231–253, 1999.

A Proof of Theorem 1

To prove the result, we state the problem in terms of matrices. We want the probability $Pr_S(P|B(u))$ to be a constant (not dependant on u). In other words we want to exhibit some constant c such that $\frac{Pr(P \cap B(u))}{Pr(B(u))} = c$ for every u, which is equivalent to:

$$Pr(P \cap B(u)) = c \times Pr(B(u)). \tag{1}$$

The system $\mathcal{S}$ is described as a Markov chain with labelled edges $\mathcal{A} = (\Sigma, s_0, A, T)$ and P is described as a deterministic Muller automaton $\mathcal{M} = (Q, A, q_0, \Delta, \mathcal{F})$. W.l.o.g., we can assume that $\mathcal{M}$ is complete. We build a Markov chain with labelled edges $\mathcal{A}'$ which is the synchronized product $\mathcal{A} \times \mathcal{M} = (\Sigma', s'_0, A, T')$ defined by:

- $S' = S \times Q$,
- $s'_0 = (s_0, q_0)$,
- the set of actions is the same: A,
- T' is defined as $T'((s,q), a, (s_1, q_1)) = p$ if $T(s, a, s_1) = p$ and (q, a, q_1) is a transition in Δ,

For a subset R of $S \times Q$, we denote its projection on Q by $\Pi_2(R)$.
Let $\mathcal{F}' = \{R \subset S \times Q | \ \Pi_2(R) \in \mathcal{F}\}$.

In order to compute $Pr(P \cap B(u))$ for a fixed word $u = l_1 l_2 ... l_n$, we decompose each trace $w \in P \cap B(u)$ as $w = w_1 w_2$ where $w_1 \in H^* l_1 H^* l_2 ... H^* l_n$.

Thus we can write:

$$P \cap B(u) = \bigcup_{(s,q) \in S \times Q} B(u)_{(s,q)} P_{(s,q)} \tag{2}$$

where $B(u)_{(s,q)}$ is the set of words $w_1 \in H^* l_1 H^* l_2 ... H^* l_n$ which have a run in $\mathcal{A}'$ from s'_0 to (s,q), and $P_{(s,q)}$ is the set of words w_2 which have a run in $\mathcal{A}'$ with a projection Π_2 of its set of infinitely repeated states belonging to $\mathcal{F}'$.

Let M_l be the matrix which contains in row $s' \in S'$ and column $t' \in S'$ the probability to reach t' from s' reading a word in $H^* l$. The set $H^* l$ being regular, this matrix is computable [2]. We define $M_u = M_{l_1} \times ... \times M_{l_n}$ which gives the probability in $\mathcal{A}'$ from each state to reach any other state after a sequence of actions in $H^* l_1 H^* l_2 ... H^* l_n$.

Let us recall what are the ergodic sets of a Markov chain. From the theory of Markov chains, it is known that the ergodic sets play a crucial role. Consider the underlying graph of the Markov chain. An ergodic set is a strongly connected component of the graph from which one cannot go out. The probability to reach an ergodic set from a given state is equal to 1 [9].

Let J be the vector which contains in the row t' the probability from t' to reach an ergodic set which belongs to $\mathcal{F}'$. The vector J is rational and is computable from a result of [3]. From (2) we have:

$$Pr(P \cap B(u)) = {}^tI_0 \times M_u \times J. \tag{3}$$

Here I_0 is the column vector corresponding to the initial distribution (probability 1 for s'_0, 0 for every other state).

The equality (1) can be formulated in terms of matrices as:

$${}^tI_0 \times M_u \times J = c \times {}^tI_0 \times M_u \times I. \tag{4}$$

Here c is a constant (a rational value) and I is the vector with all components equal to 1. A factorisation of the previous formula gives:

$${}^tI_0 \times M_u \times (J - cI) = 0 \tag{5}$$

and we need to prove that for every low level word u.

First the constant c can be easily obtained: for $u = \varepsilon$, we get ${}^tI_0 \times (J - cI) = 0$. The constant c has to be equal to the coefficient of J corresponding to s'_0.

To every word $u = l_0 l_1 .... l_n \in L^*$ is associated the matrix $M_u = M_{l_0} \times ... \times M_{l_n}$. Let us denote by ${}^tV_u$ the product ${}^tI_0 \times M_u$.

For every such word u we get a V_u. We call W the linear hull of these V_u. We then prove the following claim:

Claim *The system has no information flow for the property P if and only if the linear hull W is orthogonal to the vector $V_{check} = J - cI$.*

Since the hull contains all the vectors V_u then if W is orthogonal to V_{check} we are sure that equation (5) is satisfied for every $u \in L^*$ and that the system has no information flow. Conversely, if the system has no information flow, equation (5) is satisfied for every $u \in L^*$ and consequently for every linear combination of such V_u's, *i.e.* for every vector in W.

To prove the theorem, we only need to show that the hull W is computable. Let $\{l_1, ..., l_k\}$ be the letters of the finite alphabet L. Let W_1 be the hull generated by the set of vectors $\{I_0 = V_\varepsilon, V_{l_1}, ..., V_{l_k}\}$. From this set we only keep a subset $\{V_1, ..., V_{m_1}\}$ which is a basis of W_1. Consider the products ${}^tV_j M_{l_i}$, we get a set of $m_1 \times k$ vectors ${}^tV'_j$. If the hull W_2 of the Vs and $V's$ has the same dimension as W_1 then we are done. If not, we take some of the new V'_j to complete our basis: $\{V_1, ..., V_{m_1}, ..., V_{m_2}\}$. We repeat this operation as long as we add new vectors to our basis. As soon as we get $W_{i+1} = W_i$ for some i, it means that ${}^tW_i$ is stable under application of any of the matrices $M_{l_1}, ..., M_{l_k}$ and we have obtained W. Since the dimension of W cannot be greater than the number of states in the system, the space W is computable (we iterate our process at most $|S'|$ times). This concludes our proof of the decidability of information flow for a given property (Theorem 1).

B Proof of Theorem 3

Proof: Whereas it is obvious to prove that the criterion for the absence of information flow is decidable, we will now give a sketch of the proof of decidability for for the absence of sequential information flow.

We consider the underlying graph of the Markov chain $\mathcal{A}$ representing the system.

Property (1) is decidable:

Let us consider the underlying automaton of the Markov chain $\mathcal{A}$. First we build an automaton $\mathcal{A}'$ from $\mathcal{A}$ by adding a state q_f and, for every transition (q, l, q') (with $l \in L$), a new transition (q, l, q_f). The language recognized by this automaton $\mathcal{A}'$ with q_f as a final state is exactly $\bigcup_{n>0} Tr_n$.
We then construct a finite automaton $\mathcal{B}$ which recognizes $\bigcup_{n>0} H_n(Tr) \otimes L_n(Tr)$ as follows. Let $\mathcal{A}_L$ be a deterministic finite automaton which recognizes $\bigcup_{n>0} L_n$ (computable from $\mathcal{A}$ by abstracting away the high level actions and considering all states terminal) and let Q_1 be its set of states. The set of states of $\mathcal{B}$ is $Q \times Q_1 \cup \{q_F\}$, its initial state is the pair of initial states in Q and Q_1 and its final state is q_F. Transitions of $\mathcal{B}$ are:

- $((q, q_1), h, (q', q_1))$ if there is a transition (q, h, q') in $\mathcal{A}$ and $h \in H$
- $((q, q_1), l, (q', q_1'))$ if there is a transition (q, l', q') in $\mathcal{A}$ and a transition (q_1, l, q_1') in $\mathcal{A}_1$ with $l, l' \in L$,
- $((q, q_1), l, q_F)$ if there is a transition (q, l', q') in $\mathcal{A}$ and a transition (q_1, l, q_1') in $\mathcal{A}_1$ with $l, l' \in L$.

It is easy to prove that the language recognized by $\mathcal{B}$ is $\bigcup_{n>0} H_n(Tr) \otimes L_n(Tr)$. On the other hand we can decide whether $\mathcal{B}$ and $\mathcal{A}'$ recognize the same language which is equivalent to property (1).

Property (2) is decidable:

A H, L-compatible tuple is defined by four words in some Tr_n:
$\alpha_1 l_1 ... \alpha_n l_n$, $\beta_1 l_1 ... \beta_n l_n$, ; $\alpha_1 l_1' ... \alpha_n l_n'$, $\beta_1 l_1' ... \beta_n' l_n'$.

Consider the four paths in $\mathcal{A}$ from the initial state with these four labels, and let

$q^i \xmapsto[p^i]{l_i} q'^i$, $r^i \xmapsto[p'^i]{l_i} r'^i$, $q_1^i \xmapsto[p_1^i]{l_i'} q_1'^i$, $r_1^i \xmapsto[p_1'^i]{l_i'} r_1'^i$ the four transitions with label l_i and l_i' respectively.

We have to verify that $p^i/p'^i = p_1^i/p_1'^i$ for $i = 1, .., n$, for every $n > 0$.

It is easy to compute the set of tuples (q^i, r^i, q_1^i, r_1^i). Indeed, we build the finite automaton $\mathcal{C}$ whose set of states is $Q^4 \cup \bar{Q}^4$ where $\bar{Q}$ is a copy of Q. Transitions are:

- $(q, r, q_1, r_1) \longrightarrow (\bar{q'}, \bar{r'}, \bar{q_1'}, \bar{r_1'})$ if there exists $\alpha, \beta \in H^*$ such that
$q \xmapsto{\alpha} q'$, $r \xmapsto{\beta} r'$, $q_1 \xmapsto{\alpha} q_1'$, $r_1 \xmapsto{\beta} r_1'$,

- $(\bar{q}, \bar{r}, \bar{q_1}, \bar{r_1}) \longrightarrow (q', r', q'_1, r'_1)$ if
  $q \xmapsto{l} q',\ r \xmapsto{l} r',\ q_1 \xmapsto{l'} q'_1,\ r_1 \xmapsto{l'} r'_1$ in $\mathcal{A}$.

The initial state is (q_0, q_0, q_0, q_0) where q_0 is the initial state of $\mathcal{A}$, and we keep only the reachable states from the initial state. Let us observe that the transitions are computable. Indeed, for each (q, q'), (q_1, q'_1) let R and R_1 be the regular sets of words in H^* which are labels of paths from q to q' and from q_1 to q'_1 respectively. On can compute whether $R \cap R_1 \neq \emptyset$ and it is what we need to build the transitions of $\mathcal{C}$.

Property (2a) is satisfied iff for every transition $(\bar{q}, \bar{r}, \bar{q_1}, \bar{r_1}) \longrightarrow (q', r', q'_1, r'_1)$ in $\mathcal{C}$ we have $p/p' = p_1/p'_1$ for each (l, l') such that:
$q \xmapsto[p]{l} q',\ r \xmapsto[p']{l} r',\ q_1 \xmapsto[p_1]{l'} q'_1,\ r_1 \xmapsto[p'_1]{l'} r'_1$ in $\mathcal{A}$.

In order to prove property (2b) we have to build for each state q in the Markov chain $\mathcal{A}$ a new Markov chain $\mathcal{A}_q$ in the following way: we keep only the high edges reachable in $\mathcal{A}$ from q, and for each state s (including q) accessible from q by a high path with at least one low edge starting from s, we add a state s', an edge (s, s') labelled by ε and with a probability equal to the sum p of the probabilities of low edges starting from s in $\mathcal{A}$, and a loop in s' labelled by ε and with a probability 1. Clearly, property (2b) is satisfied iff for every state (q, r, q_1, r_1) in the automaton $\mathcal{C}$ described above, reachable from the initial state by a path of positive length, the Markov chains $\mathcal{A}_{\mathrm{II}}$ and $\mathcal{A}_{\mathrm{II}_\infty}$ taking as initial state respectively q and q_1 are isomorphic.

Property (3) can be decided looking for the ergodic sets containing only transitions labelled with events from H. If there is none then for each $n > 0$, $Pr_S(Tr \cap (H^*L)^n H^\omega) = 0$ and (3) is satisfied. If there is at least one such ergodic set, we take a path leading to one of these sets, and call n the number of low level events in this path. Then either almost all paths have exactly n low level events, and in that case $Pr_S(Tr \cap (H^*L)^n H^\omega) = 1$ and (3) is satisfied or there is:

- either a path leading to one such ergodic set with $p < n$ low level events in this path. In this last case we have $(L_{p+1}(Tr) \neq \emptyset$ and $Pr_S(Tr \cap (H^*L)^p H^\omega) \neq 0$,
- or a path with strictly more than n low level events in which case $(L_{n+1}(Tr) \neq \emptyset$ and $Pr_S(Tr \cap (H^*L)^n H^\omega) \neq 0$.

In both cases (3) is violated.
We have thus proved that all four criteria are decidable, which concludes the proof.