Math. Struct. in Comp. Science (2015), *vol.* 25, *pp.* 361–403. © Cambridge University Press 2014
doi:10.1017/S0960129513000637 First published online 12 November 2014

Quantifying opacity[†]

BÉATRICE BÉRARD[§], JOHN MULLINS[¶] and
MATHIEU SASSOLAS[§,‖,‡]

[§]*Sorbonne Université, Université P. & M. Curie, LIP6, CNRS UMR 7606, Paris, France*
Email: beatrice.berard@lip6.fr
[¶]*École Polytechnique de Montréal, Dept. of Comp. & Soft. Eng., Montreal (Quebec), Canada*
Email: john.mullins.polymtl.ca
[‖]*Université Paris-Est, LACL, Créteil, France*
Email: mathieu.sassolas@u-pec.fr

Received December 21, 2010; revised July 3, 2013

Opacity is a general language-theoretic framework in which several security properties of a system can be expressed. Its parameters are a predicate, given as a subset of runs of the system, and an observation function, from the set of runs into a set of observables. The predicate describes secret information in the system and, in the possibilistic setting, it is opaque if its membership cannot be inferred from observation.
In this paper, we propose several notions of quantitative opacity for probabilistic systems, where the predicate and the observation function are seen as random variables. Our aim is to measure (i) the probability of opacity leakage relative to these random variables and (ii) the level of uncertainty about membership of the predicate inferred from observation. We show how these measures extend possibilistic opacity, we give algorithms to compute them for regular secrets and observations, and we apply these computations on several classical examples. We finally partially investigate the non-deterministic setting.

1. Introduction

1.1. *Motivations*

Opacity (Mazaré 2005) is a very general framework where a wide range of security properties can be specified, for a system interacting with a passive attacker. This includes for instance anonymity or non-interference (Goguen and Meseguer 1982), the basic version of which states that high level actions cannot be detected by low-level observations. Non-interference alone cannot capture every type of information flow properties. Indeed, it expresses the complete absence of information flow yet many information flow properties, like anonymity, permits some information flow while peculiar piece of information is required to be kept secret. The notion of opacity was introduced with the aim to provide a uniform description for security properties e.g. non-interference, noninference, various

[†] Part of this work has been published in the proceedings of QEST'10 (Bérard *et al.* 2010).
[‡] Corresponding author: mathieu.sassolas@u-pec.fr – Université Paris-Est, LACL, 61 avenue du Général de gaulle, F-94010 Créteil Cedex, France.

notions of anonymity, key compromise and refresh, downgrading, etc (Bryans *et al.* 2008). Ensuring opacity by control was further studied in Dubreil *et al.* (2010).

The general idea behind opacity is that a passive attacker should not have worthwhile information, even though it can observe the system from the outside. The approach, as many existing information flow-theoretic approaches, is possibilistic. We mean by this that non-determinism is used as a feature to model the random mechanism generation for all possible system behaviours. As such, opacity is not accurate enough to take into account two orthogonal aspects of security properties both regarding evaluation of the information gained by a passive attacker.

The first aspect concerns the quantification of security properties. If executions leaking information are negligible with respect to the rest of executions, the overall security might not be compromised. For example, if an error may leak information, but appears only in 1% of cases, the program could still be considered safe. The definitions of opacity (Alur *et al.* 2006; Bryans *et al.* 2008) capture the existence of at least one perfect leak, but do not grasp such a measure.

The other aspect regards the category of security properties a system has to assume when interacting with an attacker able to make inferences from experiments on the basis of statistical analysis. For example, if every time the system goes *bip*, there is 99% chances that action a has been carried out by the server, then every *bip* can be guessed to have resulted from an a. Since more and more security protocols make use of randomization to reach some security objectives (Chaum 1988; Reiter and Rubin 1998), it becomes important to extend specification frameworks in order to cope with it.

1.2. *Contributions*

In this paper we investigate several ways of extending opacity to a purely probabilistic framework. Opacity can be defined either as the capacity for an external observer to deduce that a predicate was true (asymmetrical opacity) or whether a predicate is true or false (symmetrical opacity). Both notions can model relevant security properties, hence deserve to be extended. On the other hand, two directions can be taken towards the quantification of opacity. The first one, which we call *liberal*, evaluates the degree of non-opacity of a system: how big is the security hole? It aims at assessing the probability for the system to yield *perfect* information. The second direction, which is called *restrictive*, evaluates how opaque the system is: how robust is the security? The goal here is to measure how reliable is the information gained through observation. This yields up to four notions of quantitative opacity, displayed in Table 1, which are formally defined in this paper. The choice made when defining these measures was that a value 0 should be meaningful for opacity in the possibilistic sense. As a result, liberal measures are 0 when the system is opaque and restrictive ones are 0 when the system is not.

Moreover, like opacity itself, all these measures can be instantiated into several probabilistic security properties such as probabilistic non-interference and anonymity. We also show how to compute these values in some regular cases and apply the method to the dining cryptographers problem and the crowd protocols, re-confirming in passing the correctness result of Reiter and Rubin (1998).

Table 1. *The four probabilistic opacity measures.*

	Asymmetric	Symmetric
Liberal (Security hole)	LPO (PO_ℓ^A)	LPSO (PO_ℓ^S)
Restrictive (Robustness)	RPO (PO_r^A)	RPSO (PO_r^S)

Although the measures are defined in systems without non-determinism, they can be extended to the case of systems scheduled by an adversary. We show that non-memoryless schedulers are requested in order to reach optimum opacity measures.

1.3. *Related work*

Quantitative measures for security properties were first advocated in Millen (1987) and Wittbold and Johnson (1990). In Millen (1987), he makes an important step by relating the non-interference property with the notion of mutual information from information theory in the context of a system modelled by a deterministic state machine. He proves that the system satisfies the non-interference property if and only if the mutual information between the high-level input random variable and the output random variable is zero. He also proposes mutual information as a measure for information flow by showing how information flow can be seen as a noisy probabilistic channel, but he does not show how to compute this measure. In Wittbold and Johnson (1990), they introduce *nondeducibility on strategies* in the context of a non-deterministic state machine. A system satisfies nondeducibility on strategies if the observer cannot deduce information from the observation by any collusion with a secret user and using any adaptive strategies. They observe that if such a system is run multiple times with feedback between runs, information can be leaked by coding schemes across multiple runs. In this case, they show that a discrete memoryless channel can be built by associating a distribution with the noise process. From then on, numerous studies were devoted to the computation of (covert) channel capacity in various cases (see e.g. Mantel and Sudbrock (2009)) or more generally information leakage.

In Smith (2009), several measures of information leakage extending these seminal works for deterministic or probabilistic programs with probabilistic input are discussed. These measures quantify the information concerning the input gained by a passive attacker observing the output. Exhibiting programs for which the value of entropy is not meaningful, Smith proposes to consider instead the notions of vulnerability and min-entropy to take in account the fact that some execution could leak a sufficiently large amount of information to allow the environment to guess the remaining secret. As discussed in Section 6, probabilistic opacity takes this in account.

In Chatzikokolakis *et al.* (2008), in order to quantify anonymity, the authors propose to model the system (then called *Information Hiding System*) as a noisy channel in the sense of Information theory: the secret information is modelled by the inputs, the observable information is modelled by the outputs and the two sets are related by a conditional probability matrix. In this context, probabilistic information leakage is very naturally specified in terms of mutual information and capacity. A whole hierarchy of probabilistic notions of anonymity have been defined. The approach was completed in Andrés *et al.* (2010) where anonymity is computed using regular expressions. More recently, in Alvim *et al.* (2010), the authors consider interactive information hiding systems that can be viewed as channels with memory and feedback.

In Boreale *et al.* (2011a), the authors analyse the asymptotic behaviour of attacker's error probability and information leakage in information hiding systems in the context of an attacker having the capabilities to make exactly one guess after observing n independent executions of the system while the secret information remains invariant. Two cases are studied: the case in which each execution gives rise to a single observation and the case in which each state of an execution gives rise to an observation in the context of *hidden Markov models*. The relation of these sophisticated models of attacker with our attacker model is still to clarify. Similar models were also studied in McIver *et al.* (2010), where the authors define an ordering w.r.t. probabilistic non-interference.

For systems modelled by process algebras, pioneering work was presented in Lowe (2004) and Aldini *et al.* (2004), with channel capacity defined by counting behaviours in discrete time (non-probabilistic) CSP (Lowe 2004), or various probabilistic extensions of non-interference (Aldini *et al.* 2004) in a generative-reactive process algebra. Subsequent studies in this area by Aldini and Bernardo (2009), Boreale (2009) and Boreale *et al.* (2010) also provide quantitative measures of information leak, relating these measures with non-interference and secrecy. In Aldini and Bernardo (2009), the authors introduce various notions of non-interference in a Markovian process calculus extended with prioritized/probabilistic zero duration actions and untimed actions. In Boreale (2009), the author introduces two notions of information leakage in the (non-probabilistic) π-calculus differing essentially in the assumptions made on the power of the attacker. The first one, called *absolute leakage*, corresponds to the average amount of information that was leaked to the attacker by the program in the context of an attacker with unlimited computational resources. It is defined in terms of conditional mutual information and follows the earlier results of Millen (1987). The second notion, called *leakage rate*, corresponds to the maximal number of bits of information that could be obtained per experiment in the context in which the attacker can only perform a fixed number of tries, each yielding a binary outcome representing success or failure. Boreale also studies the relation between both notions of leakage and proves that they are consistent. The author also investigates compositionality of leakage. Boreale *et al.* (2010), propose a very general framework for reasoning about information leakage in a sequential process calculus over a semiring with some appealing applications to information leakage analysis when instantiating and interpreting the semiring. It appears to be a promising scheme for specifying and analysing regular quantitative information flow like we do in Section 5.

Although the literature on quantifying information leakage or channel capacity is dense, few works actually tried to extend general opacity to a probabilistic setting. A notion of probabilistic opacity is defined in Lakhnech and Mazaré (2005), but restricted to properties whose satisfaction depends only on the initial state of the run. The opacity there corresponds to the probability for an observer to guess from the observation whether the predicate holds for the run. In that sense our restrictive opacity (Section 4) is close to that notion. However, the definition of Lakhnech and Mazaré (2005) lacks clear ties with the possibilistic notion of opacity. Probabilistic opacity is somewhat related to the notion of *view* presented in Boreale *et al.* (2011b) as authors include, like we do, a predicate to their probabilistic model and observation function but probabilistic opacity can hardly be compared with view. Indeed, on one hand, although their setting is different (they work on information hiding systems extended with a view), our predicates over runs could be viewed as a generalization of the predicates over a finite set of states (properties). On the other hand, a view in their setting is an arbitrary partition of the state space, whereas we partition the runs into only two equivalence classes (corresponding to *true* and *false*).

1.4. *Organization of the paper*

In Section 2, we recall the definitions of opacity and the probabilistic framework used throughout the paper. Sections 3 and 4 present respectively the liberal and the restrictive version of probabilistic opacity, both for the asymmetrical and symmetrical case. We present in Section 5, how to compute these measures automatically if the predicate and observations are regular. Section 6 compares the different measures and what they allow to detect about the security of the system, through abstract examples and a case study of the Crowds protocol. In Section 7, we present the framework of probabilistic systems dealing with non-determinism, and open problems that arise in this setting.

2. Preliminaries

In this section, we recall the notions of opacity, entropy and probabilistic automata.

2.1. *Possibilistic opacity*

The original definition of opacity was given in Bryans *et al.* (2008) for transition systems.

Recall that a transition system is a tuple $\mathcal{A} = \langle \Sigma, Q, \Delta, I \rangle$ where Σ is a set of actions, Q is a set of states, $\Delta \subseteq Q \times \Sigma \times Q$ is a set of transitions and $I \subseteq Q$ is a subset of initial states. A *run* in $\mathcal{A}$ is a finite sequence of transitions written as: $\rho = q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \cdots \xrightarrow{a_n} q_n$. For such a run, $\mathrm{fst}(\rho)$ (resp. $\mathrm{lst}(\rho)$) denotes q_0 (resp. q_n). We will also write $\rho \cdot \rho'$ for the run obtained by concatenating runs ρ and ρ' whenever $\mathrm{lst}(\rho) = \mathrm{fst}(\rho')$. The set of runs starting in state q is denoted by $Run_q(\mathcal{A})$ and $Run(\mathcal{A})$ denotes the set of runs starting from some initial state: $Run(\mathcal{A}) = \bigcup_{q \in I} Run_q(\mathcal{A})$.

Opacity qualifies a predicate φ, given as a subset of $Run(\mathcal{A})$ (or equivalently as its characteristic function $\mathbf{1}_\varphi$), with respect to an *observation function* $\mathcal{O}$ from $Run(\mathcal{A})$ onto a (possibly infinite) set *Obs* of *observables*. Two runs ρ and ρ' are equivalent w.r.t. $\mathcal{O}$ if they produce the same observable: $\mathcal{O}(\rho) = \mathcal{O}(\rho')$. The set $\mathcal{O}^{-1}(o)$, for o in Obs, is called an *observation class*. We sometimes write $[\rho]_\mathcal{O}$ for $\mathcal{O}^{-1}(\mathcal{O}(\rho))$.

A predicate φ is opaque on $\mathcal{A}$ for $\mathcal{O}$ if for every run ρ satisfying φ, there is a run ρ' not satisfying φ equivalent to ρ.

Definition 2.1 (opacity). Let $\mathcal{A}$ be a transition system and $\mathcal{O} : Run(\mathcal{A}) \to Obs$ a surjective function called observation. A predicate $\varphi \subseteq Run(\mathcal{A})$ is *opaque* on $\mathcal{A}$ for $\mathcal{O}$ if, for any $o \in Obs$, the following holds:

$$\mathcal{O}^{-1}(o) \nsubseteq \varphi.$$

However, detecting whether an event *did not* occur can give as much information as the detection that the same event *did* occur. In addition, as argued in Alur *et al.* (2006), the asymmetry of this definition makes it impossible to use with refinement: opacity would not be ensured in a system derived from a secure one in a refinement-driven engineering process. More precisely, if $\mathcal{A}'$ refines $\mathcal{A}$ and a property φ is opaque on $\mathcal{A}$ (w.r.t $\mathcal{O}$), φ is not guaranteed to be opaque on $\mathcal{A}'$ (w.r.t $\mathcal{O}$).

Hence, we use the symmetric notion of opacity, where a predicate is symmetrically opaque if it is opaque as well as its negation. More precisely:

Definition 2.2 (symmetrical opacity). A predicate $\varphi \subseteq Run(\mathcal{A})$ is *symmetrically opaque* on system $\mathcal{A}$ for observation function $\mathcal{O}$ if, for any $o \in Obs$, the following holds:

$$\mathcal{O}^{-1}(o) \nsubseteq \varphi \text{ and } \mathcal{O}^{-1}(o) \nsubseteq \overline{\varphi}.$$

The symmetrical opacity is a stronger security requirement. Security goals can be expressed as either symmetrical or asymmetrical opacity, depending on the property at stake.

For example, non-interference and anonymity can be expressed by opacity properties. Non-interference states that an observer cannot know whether an action h of high-level accreditation occurred only by looking at the actions with low-level of accreditation in the set L. So non-interference is equivalent to the opacity of predicate φ_{NI}, which is true when h occurred in the run, with respect to the observation function $\mathcal{O}_L$ that projects the trace of a run onto the letters of L; see Section 3.2 for a full example. We refer to Bryans *et al.* (2008) and Lin (2011) for other examples of properties using opacity.

When the predicate breaks the symmetry of a model, the asymmetric definition is usually more suited. Symmetrical opacity is however used when knowing φ or $\overline{\varphi}$ is equivalent from a security point of view. For example, a noisy channel with binary input can be seen as a system $\mathcal{A}$ on which the input is the truth value of φ and the output is the observation $o \in Obs$. If φ is symmetrically opaque on $\mathcal{A}$ with respect to $\mathcal{O}$, then this channel is not perfect: there would always be a possibility of erroneous transmission. The ties between channels and probabilistic transition systems are studied in Andrés *et al.* (2010) (see discussion in Section 4.2).

2.2. *Probabilities and information theory*

Recall that, for a countable set Ω, a *discrete distribution* (or *distribution* for short) is a mapping $\mu : \Omega \to [0, 1]$ such that $\Sigma_{\omega\in\Omega}\mu(\omega) = 1$. For any subset E of Ω, $\mu(E) = \Sigma_{\omega\in E}\mu(\omega)$. The set of all discrete distributions on Ω is denoted by $\mathcal{D}(\Omega)$. A *discrete random variable* with values in a set Γ is a mapping $Z : \Omega \to \Gamma$ where $[Z = z]$ denotes the event $\{\omega \in \Omega \mid Z(\omega) = z\}$.

The *entropy* of Z is a measure of the uncertainty or dually, information about Z, defined by the expected value of $\log(\mu(Z))$:

$$H(Z) = -\sum_{z} \mu(Z = z) \cdot \log(\mu(Z = z))$$

where log is the base 2 logarithm.

For two random variables Z and Z' on Ω, the *conditional entropy* of Z *given the event* $[Z' = z']$ such that $\mu(Z' = z') \neq 0$ is defined by:

$$H(Z|Z' = z') = -\sum_{z} \big(\mu(Z = z|Z' = z') \cdot \log(\mu(Z = z|Z' = z'))\big)$$

where $\mu(Z = z|Z' = z') = \frac{\mu(Z=z, Z'=z')}{\mu(Z'=z')}$.

The *conditional entropy* of Z *given the random variable* Z' can be interpreted as the average entropy of Z that remains after the observation of Z'. It is defined by:

$$H(Z|Z') = \sum_{z'} \mu(Z' = z') \cdot H(Z|Z' = z').$$

The *vulnerability* of a random variable Z, defined by $V(Z) = \max_z \mu(Z = z)$ gives the probability of the likeliest event of a random variable. Vulnerability evaluates the probability of a correct guess in one attempt and can also be used as a measure of information by defining *min-entropy* and *conditional min-entropy* (see discussions in Smith (2009); Andrés *et al.* (2010)).

2.3. *Probabilistic models*

In this work, systems are modelled using probabilistic automata behaving as finite automata where non-deterministic choices for the next action and state or termination are randomized: this is why they are called '*fully* probabilistic'. We follow the model definition of Segala (1995), which advocates for the use of this special termination action (denoted here by $\surd$ instead of δ there). However, the difference lies in the model semantics: we consider only finite runs, which involves a modified definition for the (discrete) probability on the set of runs. Extensions to the non-deterministic setting are discussed in Section 7.

Recall that a finite automaton (FA) is a tuple $\mathcal{A} = \langle \Sigma, Q, \Delta, I, F \rangle$ where $\langle \Sigma, Q, \Delta, I \rangle$ is a finite transition system and $F \subseteq Q$ is a subset of final states. The automaton is deterministic if I is a singleton and for all $q \in Q$ and $a \in \Sigma$, the set $\{q' \mid (q, a, q') \in \Delta\}$ is a singleton. Runs in $\mathcal{A}$, $Run_q(\mathcal{A})$ and $Run(\mathcal{A})$ are defined like in a transition system. A run of an FA is *accepting* if it ends in a state of F. The *trace* of a run $\rho = q_0 \xrightarrow{a_1} q_1 \cdots \xrightarrow{a_n} q_n$ is the word $\mathrm{tr}(\rho) = a_1 \cdots a_n \in \Sigma^*$. The *language* of $\mathcal{A}$, written $\mathcal{L}(\mathcal{A})$, is the set of traces of accepting runs starting in some initial state.

Replacing in a FA non-deterministic choices by choices based on a discrete distribution and considering only finite runs result in a *fully probabilistic finite automaton* (FPFA). Consistently with the standard notion of substochastic matrices, we also consider a more general class of automata, *substochastic automata* (SA), which allow us to describe subsets of behaviours from FPFAs, see Figure 1 for examples. In both models, no non-determinism remains, thus the system is to be considered as autonomous: its behaviours do not depend on an exterior probabilistic agent acting as a scheduler for non-deterministic choices.

Definition 2.3 (substochastic automaton). Let $\surd$ be a new symbol representing a termination action. A *substochastic automaton* (SA) is a tuple $\langle \Sigma, Q, \Delta, q_0 \rangle$ where Σ is a finite set of actions, Q is a finite set of states, with $q_0 \in Q$ the initial state and $\Delta : Q \to ((\Sigma \times Q) \uplus \{\surd\} \to [0,1])$ is a mapping such that for any $q \in Q$,

$$\sum_{x \in (\Sigma \times Q) \uplus \{\surd\}} \Delta(q)(x) \leqslant 1$$

Δ defines substochastically the action and successor from the current state, or the termination action $\surd$.

In SA, we write $q \to \mu$ for $\Delta(q) = \mu$ and $q \xrightarrow{a} r$ whenever $q \to \mu$ and $\mu(a,r) > 0$. We also write $q \cdot \surd$ whenever $q \to \mu$ and $\mu(\surd) > 0$. In the latter case, q is said to be a *final state*.

Definition 2.4 (fully probabilistic finite automaton). A FPFA is a particular case of SA where for all $q \in Q$, $\Delta(q) = \mu$ is a distribution in $\mathcal{D}((\Sigma \times Q) \uplus \{\surd\})$ *i.e.*

$$\sum_{x \in (\Sigma \times Q) \uplus \{\surd\}} \Delta(q)(x) = 1$$

and for any state $q \in Q$ there exists a path (with non-zero probability) from q to some final state.

Note that we only target finite runs and therefore we consider a restricted case, where any infinite path has probability 0.

Since FPFA is a subclass of SA, we overload the metavariable $\mathcal{A}$ for both SA and FPFA. The notation above allows to define a run for an SA like in a transition system as a finite sequence of transitions written $\rho = q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \cdots \xrightarrow{a_n} q_n$. The sets $Run_q(\mathcal{A})$ and $Run(\mathcal{A})$ are defined like in a transition system. A *complete run* is a (finite) sequence denoted by $\rho \cdot \surd$ where ρ is a run and $\Delta(\mathrm{lst}(\rho))(\surd) > 0$. The set $CRun(\mathcal{A})$ denotes the set of complete runs starting from the initial state. In this work, we consider only such complete runs.

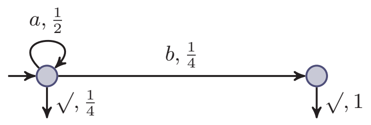
The *trace* of a run for an SA $\mathcal{A}$ is defined like in finite automata. The *language* of a substochastic automaton $\mathcal{A}$, written $\mathcal{L}(\mathcal{A})$, is the set of traces of complete runs starting in the initial state.

For an SA $\mathcal{A}$, a mapping $\mathbf{P}_{\mathcal{A}}$ into $[0,1]$ can be defined inductively on the set of complete runs by:

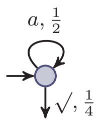
$$\mathbf{P}_{\mathcal{A}}(q\surd) = \mu(\surd) \qquad \text{and} \qquad \mathbf{P}_{\mathcal{A}}(q \xrightarrow{a} \rho) = \mu(a,r) \cdot \mathbf{P}_{\mathcal{A}}(\rho)$$

where $q \to \mu$ and $\mathrm{fst}(\rho) = r$.

The mapping $\mathbf{P}_{\mathcal{A}}$ is then a discrete distribution on $CRun(\mathcal{A})$. Indeed, the $\surd$ action can be seen as a transition label towards a new sink state $q_\surd$. Then, abstracting from the labels yields a finite Markov chain, where $q_\surd$ is the only absorbing state and the coefficients of the transition matrix are $M_{q,q'} = \Sigma_{a \in \Sigma} \Delta(q)(a, q')$. The probability for a complete run to have length n is then the probability p_n to reach $q_\surd$ in exactly n steps.



(a) FPFA $\mathcal{A}_1$



(b) SA $\mathcal{A}_2$

Fig. 1. (Colour online) $\mathcal{A}_2$ is the restriction of $\mathcal{A}_1$ to a^*.

Therefore, the probability of all finite complete runs is $\mathbf{P}(CRun(\mathcal{A})) = \Sigma_n p_n$ and a classical result (Kemeny and Snell 1976) on absorbing chains ensures that this probability is equal to 1.

Since the probability space is not generated by (prefix-closed) cones, this definition does not yield the same probability measure as the one from Segala (1995). Since opacity properties are not necessarily prefix-closed, this definition is consistent with our approach.

When $\mathcal{A}$ is clear from the context, $\mathbf{P}_{\mathcal{A}}$ will simply be written $\mathbf{P}$.

Since $\mathbf{P}_{\mathcal{A}}$ is a (sub-)probability on $CRun(\mathcal{A})$, for any predicate $\varphi \subseteq CRun(\mathcal{A})$, we have $\mathbf{P}(\varphi) = \Sigma_{\rho\in\varphi}\mathbf{P}(\rho)$. The measure is extended to languages $K \subseteq \mathcal{L}(\mathcal{A})$ by $\mathbf{P}(K) = \mathbf{P}\left(\mathrm{tr}^{-1}(K)\right) = \Sigma_{\mathrm{tr}(\rho)\in K}\mathbf{P}(\rho)$.

In the examples of Figure 1, restricting the complete runs of $\mathcal{A}_1$ to those satisfying $\varphi = \{\rho \mid \mathrm{tr}(\rho) \in a^*\}$ yields the SA $\mathcal{A}_2$, and $\mathbf{P}_{\mathcal{A}_1}(\varphi) = \mathbf{P}_{\mathcal{A}_2}(CRun(\mathcal{A}_2)) = \frac{1}{2}$.

A non-probabilistic version of any SA is obtained by forgetting any information about probabilities.

Definition 2.5. Let $\mathcal{A} = \langle \Sigma, Q, \Delta, q_0\rangle$ be an SA. The (non-deterministic) finite automaton $unProb(\mathcal{A}) = \langle \Sigma, Q, \Delta', q_0, F\rangle$ is defined by:

— $\Delta' = \{(q, a, r) \in Q \times \Sigma \times Q \mid q \to \mu,\ \mu(a, r) > 0\}$,
— $F = \{q \in Q \mid q \to \mu, \mu(\surd) > 0\}$ is the set of final states.

It is easily seen that $\mathcal{L}(unProb(\mathcal{A})) = \mathcal{L}(\mathcal{A})$.

An observation function $\mathcal{O} : CRun(\mathcal{A}) \to Obs$ can also be easily translated from the probabilistic to the non-probabilistic setting. For $\mathcal{A}' = unProb(\mathcal{A})$, we define $unProb(\mathcal{O})$ on $Run(\mathcal{A}')$ by $unProb(\mathcal{O})(q_0 \xrightarrow{a_1} q_1 \cdots q_n) = \mathcal{O}(q_0 \xrightarrow{a_1} q_1 \cdots q_n\surd)$.

3. Measuring non-opacity

3.1. *Definition and properties*

One of the aspects in which the definition of opacity could be extended to probabilistic automata is by relaxing the universal quantifiers of Definitions 2.1 and 2.2. Instead of wanting that *every* observation class should not be included in φ (resp. φ or $\overline{\varphi}$ for the symmetrical case), we can just require that *almost all* of them do. To obtain this, we give a measure for the set of runs leaking information. To express properties of probabilistic opacity in an FPA $\mathcal{A}$, the observation function $\mathcal{O}$ is considered as a random variable, as well as the characteristic function $\mathbf{1}_\varphi$ of φ. Both the asymmetrical and the symmetrical notions of opacity can be generalized in this manner.

Definition 3.1 (liberal probabilistic opacity). The *liberal probabilistic opacity* or LPO of predicate φ on FPA $\mathcal{A}$, with respect to (surjective) observation function $\mathcal{O} : CRun \to Obs$ is defined by:

$$\mathsf{PO}_\ell^A(\mathcal{A},\varphi,\mathcal{O}) = \sum_{\substack{o\in Obs\\ \mathcal{O}^{-1}(o)\subseteq\varphi}} \mathbf{P}(\mathcal{O}=o).$$

The *liberal probabilistic symmetrical opacity* or LPSO is defined by:

$$\begin{aligned}\mathsf{PO}_\ell^S(\mathcal{A},\varphi,\mathcal{O}) &= \mathsf{PO}_\ell^A(\mathcal{A},\varphi,\mathcal{O}) + \mathsf{PO}_\ell^A(\mathcal{A},\overline{\varphi},\mathcal{O})\\ &= \sum_{\substack{o\in Obs\\ \mathcal{O}^{-1}(o)\subseteq\varphi}} \mathbf{P}(\mathcal{O}=o) + \sum_{\substack{o\in Obs\\ \mathcal{O}^{-1}(o)\subseteq\overline{\varphi}}} \mathbf{P}(\mathcal{O}=o).\end{aligned}$$

This definition provides a measure of how insecure the system is. The following propositions shows that a null value for these measures coincides with (symmetrical) opacity for the system, which is then secure.

For LPO, it corresponds to classes either overlapping both φ and $\overline{\varphi}$ or included in $\overline{\varphi}$ as in Figure 2a. LPO measures only the classes that leak their inclusion in φ. So classes included in $\overline{\varphi}$ are not taken into account. On the other extremal point, $\mathsf{PO}_\ell^A(\mathcal{A},\varphi,\mathcal{O}) = 1$ when φ is always true.

When LPSO is null, it means that each equivalence class $\mathcal{O}^{-1}(o)$ overlaps both φ and $\overline{\varphi}$ as in Figure 2c. On the other hand, the system is totally insecure when, observing through $\mathcal{O}$, we have all information about φ. In that case, the predicate φ is a union of equivalence classes $\mathcal{O}^{-1}(o)$ as in Figure 2e and this can be interpreted in terms of conditional entropy relatively to $\mathcal{O}$. The intermediate case occurs when some, but not all, observation classes contain only runs satisfying φ or only runs not satisfying φ, as in Figure 2d.

Proposition 3.2.

1. $0 \leqslant \mathsf{PO}_\ell^A(\mathcal{A},\varphi,\mathcal{O}) \leqslant 1$ and $0 \leqslant \mathsf{PO}_\ell^S(\mathcal{A},\varphi,\mathcal{O}) \leqslant 1$.
2. $\mathsf{PO}_\ell^A(\mathcal{A},\varphi,\mathcal{O}) = 0$ if and only if φ is opaque on $unProb(\mathcal{A})$ with respect to $unProb(\mathcal{O})$. $\mathsf{PO}_\ell^S(\mathcal{A},\varphi,\mathcal{O}) = 0$ if and only if φ is symmetrically opaque on $unProb(\mathcal{A})$ with respect to $unProb(\mathcal{O})$.
3. $\mathsf{PO}_\ell^A(\mathcal{A},\varphi,\mathcal{O}) = 1$ if and only if $\varphi = CRun(\mathcal{A})$.
   $\mathsf{PO}_\ell^S(\mathcal{A},\varphi,\mathcal{O}) = 1$ if and only if $H(\mathbf{1}_\varphi|\mathcal{O}) = 0$.

Proof of Proposition 3.2.

1. The considered events are mutually exclusive, hence the sum of their probabilities never exceeds 1.
2. First observe that a complete run $r_0a,\ldots,r_n\surd$ has a non-null probability in $\mathcal{A}$ iff $r_0a,\ldots,r_n$ is a run in $unProb(\mathcal{A})$. Suppose $\mathsf{PO}_\ell^A(\mathcal{A},\varphi,\mathcal{O}) = 0$. Recall that $\mathcal{O}$ is assumed surjective. Then there is no observable o such that $\mathcal{O}^{-1}(o) \subseteq \varphi$. Conversely, if φ is opaque on $unProb(\mathcal{A})$, there is no observable $o \in Obs$ such that $\mathcal{O}^{-1}(o) \subseteq \varphi$, hence the null value for $\mathsf{PO}_\ell^A(\mathcal{A},\varphi,\mathcal{O})$. The case of LPSO is similar, also taking into account the dual case of $\overline{\varphi}$ in the above.

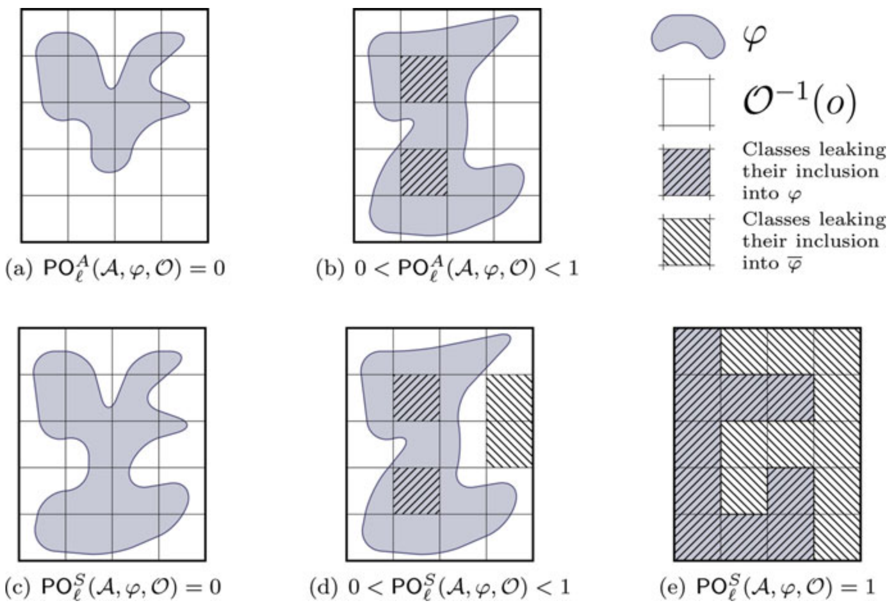

φ
$\mathcal{O}^{-1}(o)$
Classes leaking their inclusion into φ
Classes leaking their inclusion into $\overline{\varphi}$
(a) $\mathsf{PO}^A_\ell(\mathcal{A},\varphi,\mathcal{O}) = 0$
(b) $0 < \mathsf{PO}^A_\ell(\mathcal{A},\varphi,\mathcal{O}) < 1$
(c) $\mathsf{PO}^S_\ell(\mathcal{A},\varphi,\mathcal{O}) = 0$
(d) $0 < \mathsf{PO}^S_\ell(\mathcal{A},\varphi,\mathcal{O}) < 1$
(e) $\mathsf{PO}^S_\ell(\mathcal{A},\varphi,\mathcal{O}) = 1$

Fig. 2. (Colour online) Liberal probabilistic asymmetrical and symmetrical opacity.

3. For LPO, this is straightforward from the definition. For LPSO, $H(\mathbf{1}_\varphi|\mathcal{O}) = 0$ iff

$$\sum_{\substack{o\in Obs\\ i\in\{0,1\}}} \mathbf{P}(\mathbf{1}_\varphi = i|\mathcal{O} = o)\cdot\log(\mathbf{P}(\mathbf{1}_\varphi = i|\mathcal{O} = o)) = 0$$

Since all the terms have the same sign, this sum is null if and only if each of its term is null. Setting for every $o \in Obs$, $f(o) = \mathbf{P}(\mathbf{1}_\varphi = 1|\mathcal{O} = o) = 1 - \mathbf{P}(\mathbf{1}_\varphi = 0|\mathcal{O} = o)$, we have: $H(\mathbf{1}_\varphi|\mathcal{O}) = 0$ iff $\forall\, o \in Obs$, $f(o)\cdot\log(f(o)) + (1 - f(o))\cdot\log(1 - f(o)) = 0$. Since the equation $x \cdot \log(x) + (1 - x)\cdot\log(1 - x) = 0$ only accepts 1 and 0 as solutions, it means that for every observable o, either all the runs ρ such that $\mathcal{O}(\rho) = o$ are in φ, or they are all not in φ. Therefore, $H(\mathbf{1}_\varphi|\mathcal{O}) = 0$ iff for every observable o, $\mathcal{O}^{-1}(o) \subseteq \varphi$ or $\mathcal{O}^{-1}(o) \subseteq \overline{\varphi}$, which is equivalent to $\mathsf{PO}^S_\ell(\mathcal{A},\varphi,\mathcal{O}) = 1$.

□

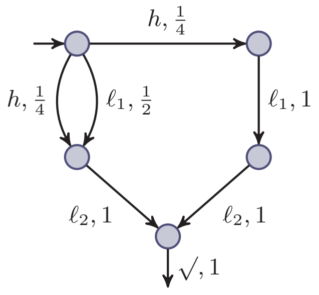
3.2. *Example: non-interference*

For the systems $\mathcal{A}_3$ and $\mathcal{A}_4$ of Figure 3, we use the predicate φ_{NI} which is true if the trace of a run contains the letter h. In both cases, the observation function $\mathcal{O}_L$ returns the projection of the trace onto the alphabet $\{\ell_1, \ell_2\}$. Remark that this example is an interference property (Goguen and Meseguer 1982) seen as opacity. Considered unprobabilistically, both systems are interferent since an ℓ_2 not preceded by an ℓ_1 betrays the presence of an h. However, they differ by how often this case happens.

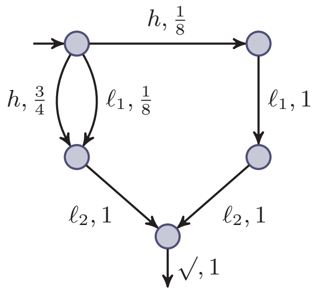
The runs of $\mathcal{A}_3$ and $\mathcal{A}_4$ and their properties are displayed in Table 2. Then we can see that $[\rho_1]_{\mathcal{O}_L} = [\rho_2]_{\mathcal{O}_L}$ overlaps both φ_{NI} and $\overline{\varphi_{\mathrm{NI}}}$, while $[\rho_3]_{\mathcal{O}_L}$ is contained totally in φ.

Table 2. *Runs of $\mathcal{A}_3$ and $\mathcal{A}_4$.*

$\mathrm{tr}(\rho)$	$\mathbf{P}_{\mathcal{A}_3}(\rho)$	$\mathbf{P}_{\mathcal{A}_4}(\rho)$	$\in \varphi_{NI}$?	$\mathcal{O}_L(\rho)$
$\mathrm{tr}(\rho_1) = \ell_1\ell_2\surd$	$1/2$	$1/8$	0	$\ell_1\ell_2$
$\mathrm{tr}(\rho_2) = h\ell_1\ell_2\surd$	$1/4$	$1/8$	1	$\ell_1\ell_2$
$\mathrm{tr}(\rho_3) = h\ell_2\surd$	$1/4$	$3/4$	1	ℓ_2



(a) FPA $\mathcal{A}_3$



(b) FPA $\mathcal{A}_4$

Fig. 3. (Colour online) Interferent FPAs $\mathcal{A}_3$ and $\mathcal{A}_4$.

Hence the LPO can be computed for both systems:

$$\mathsf{PO}_\ell^A(\mathcal{A}_3, \varphi_{\mathrm{NI}}, \mathcal{O}_L) = \frac{1}{4} \qquad \mathsf{PO}_\ell^A(\mathcal{A}_4, \varphi_{\mathrm{NI}}, \mathcal{O}_L) = \frac{3}{4}.$$

Therefore, $\mathcal{A}_3$ is more secure than $\mathcal{A}_4$. Indeed, the run that is interferent occurs more often in $\mathcal{A}_4$, leaking information more often.

Note that in this example, LPO and LPSO coincide. This is not always the case. Indeed, in the unprobabilistic setting, both symmetrical and asymmetrical opacity of φ_{NI} with respect to $\mathcal{O}_L$ express the intuitive notion that 'an external observer does not know whether an action happened or not'. The asymmetrical notion corresponds to the definition of *strong non-deterministic non-interference* in Goguen and Meseguer (1982) while the symmetrical one was defined as *perfect security property* in Alur *et al.* (2006).

4. Measuring the robustness of opacity

The completely opposite direction that can be taken to define a probabilistic version is a more paranoid one: how much information is leaked through the system's uncertainty? For example, on Figure 2c, even though each observation class contains a run in φ and one in $\overline{\varphi}$, some classes are *nearly* in φ. In some other classes, the balance between the runs satisfying φ and the ones not satisfying φ is more even. Hence for each observation class, we will not ask if it is included in φ, but how likely φ is to be true inside this class with a probabilistic measure taking into account the likelihood of classes. This amounts to measuring, inside each observation class, $\overline{\varphi}$ in the case of asymmetrical opacity, and the balance between φ and $\overline{\varphi}$ in the case of symmetrical opacity. Note that these new measures are relevant only for opaque systems, where the previous liberal measures are equal to zero.

In Bérard *et al.* (2010), another measure was proposed, based on the notion of mutual information (from information theory, along similar lines as in Smith (2009)). However, this measure had a weaker link with possibilistic opacity (see discussion in Section 6). What we call here restrictive probabilistic opacity (RPO) is a new measure, whose relation with possibilistic opacity is expressed by the second item in Proposition 4.2.

4.1. *Restricting asymmetrical opacity*

In this section, we extend the notion of asymmetrical opacity in order to measure how secure the system is.

4.1.1. *Definition and properties.*

In this case, an observation class is more secure if φ is less likely to be true. That means, that it is easy (as in 'more likely') to find a run not in φ in the same observation class. Dually, a high probability for φ inside a class means that few (again probabilistically speaking) runs will be in the same class yet not in φ.

Restrictive probabilistic opacity is defined to measure this effect globally on all observation classes. It is tailored to fit the definition of opacity in the classical sense: indeed, if one class totally leaks its presence in φ, RPO will detect it (second point in Proposition 4.2).

Definition 4.1. Let φ be a predicate on the complete runs of an FPA $\mathcal{A}$ and $\mathcal{O}$ an observation function. The RPO of φ on $\mathcal{A}$, with respect to $\mathcal{O}$, is defined by

$$\frac{1}{\mathsf{PO}_r^A(\mathcal{A},\varphi,\mathcal{O})} = \sum_{o \in Obs} \mathbf{P}(\mathcal{O}=o) \cdot \frac{1}{\mathbf{P}(\mathbf{1}_\varphi = 0 \mid \mathcal{O}=o)}$$

RPO is the harmonic means (weighted by the probabilities of observations) of the probability that φ is false in a given observation class. The harmonic means averages the leakage of information inside each class. Since security and robustness are often evaluated on the weakest link, more weight is given to observation classes with the higher leakage, *i.e.* those with probability of φ being false closest to 0.

The following proposition gives properties of RPO.

Proposition 4.2.

1. $0 \leqslant \mathsf{PO}_r^A(\mathcal{A},\varphi,\mathcal{O}) \leqslant 1$.
2. $\mathsf{PO}_r^A(\mathcal{A},\varphi,\mathcal{O}) = 0$ if and only if φ is not opaque on $unProb(\mathcal{A})$ with respect to $unProb(\mathcal{O})$.
3. $\mathsf{PO}_r^A(\mathcal{A},\varphi,\mathcal{O}) = 1$ if and only if $\varphi = \emptyset$.

Proof. The first point immediately results from the fact that RPO is a means of values between 0 and 1.

From the definition above, RPO is null if and only if there is one class that is contained in φ. Indeed, this corresponds to the case where the value of $\frac{1}{\mathbf{P}(\mathbf{1}_\varphi=0|\mathcal{O}=o)}$ goes to $+\infty$, for some o, as well as the sum.

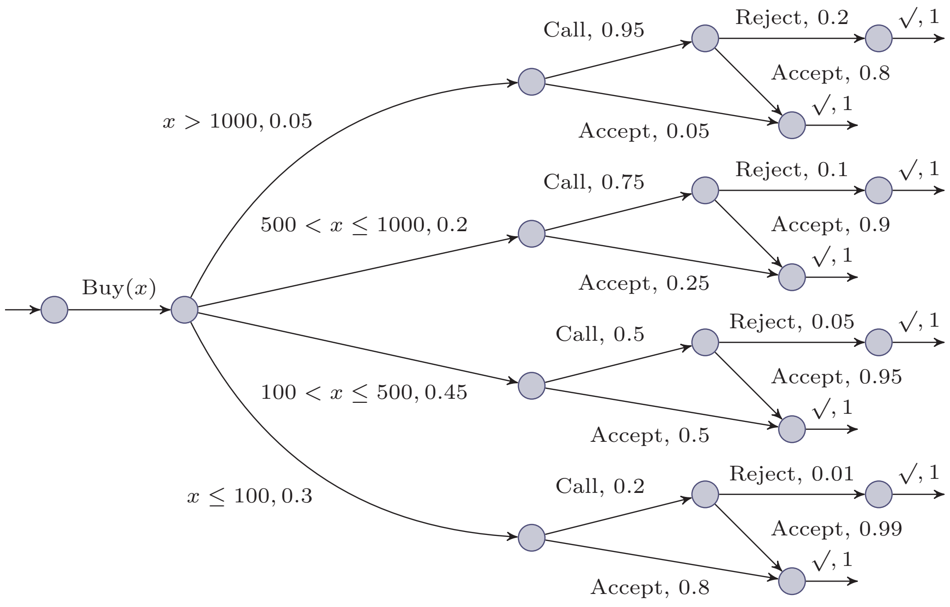


Fig. 4. (Colour online) The Debit Card system $\mathcal{A}_{\text{card}}$.

Thirdly, if φ is always false, then RPO is 1 since it is a means of probabilities all of value 1. Conversely, if RPO is 1, because it is defined as an average of values between 0 and 1, then all these values must be equal to 1, hence for each o, $\mathbf{P}(\mathbf{1}_\varphi = 0 \mid \mathcal{O} = o) = 1$ which means that $\mathbf{P}(\mathbf{1}_\varphi = 0) = 1$ and φ is false. □

4.1.2. *Example: debit card system.* Consider a debit card system in a store. When a card is inserted, an amount of money x to be debited is entered, and the client enters his PIN number (all this being gathered as the action Buy(x)). The amount of the transaction is given probabilistically as an abstraction of the statistics of such transactions. Provided the PIN is correct, the system can either directly allow the transaction, or interrogate the client's bank for solvency. In order to balance the cost associated with this verification (bandwidth, server computation, etc.) with the loss induced if an insolvent client was debited, the decision to interrogate the bank's servers is taken probabilistically according to the amount of the transaction. When interrogated, the bank can reject the transaction with a certain probability[†] or accept it. This system is represented by the FPA $\mathcal{A}_{\text{card}}$ of Figure 4.

Now assume, an external observer can only observe if there has been a call or not to the bank server. In practice, this can be achieved, for example, by measuring the time taken for the transaction to be accepted (it takes longer when the bank is called), or by spying on the telephone line linking the store to the bank's servers (detecting activity on

[†] Although the bank process to allow or forbid the transaction is deterministic, the statistics of the result can be abstracted into probabilities.

the network or idleness). Suppose, what the external observer wants to know is whether the transaction was worth more than 500€. By using RPO, one can assess how this knowledge can be derived from observation.

Formally, in this case the observables are $\{\varepsilon, \text{Call}\}$, the observation function $\mathcal{O}_{\text{Call}}$ being the projection on $\{\text{Call}\}$. The predicate to be hidden to the user is represented by the regular expression $\varphi_{>500} = \Sigma^*(\text{'}x > 1000\text{'}+\text{'}500 < x \leqslant 1000\text{'})\Sigma^*$ (where Σ is the whole alphabet). By definition of RPO:

$$\begin{aligned}\frac{1}{\mathsf{PO}_r^A(\mathcal{A}_{\text{card}}, \varphi_{>500}, \mathcal{O}_{\text{Call}})} &= \mathbf{P}(\mathcal{O}_{\text{Call}} = \varepsilon) \cdot \frac{1}{\mathbf{P}(\neg\varphi_{>500}|\mathcal{O}_{\text{Call}} = \varepsilon)} \\ &\quad + \mathbf{P}(\mathcal{O}_{\text{Call}} = \text{Call}) \cdot \frac{1}{\mathbf{P}(\neg\varphi_{>500}|\mathcal{O}_{\text{Call}} = \text{Call})}\end{aligned}$$

Computing successively $\mathbf{P}(\mathcal{O}_{\text{Call}} = \varepsilon)$, $\mathbf{P}(\neg\varphi_{>500}|\mathcal{O}_{\text{Call}} = \varepsilon)$, $\mathbf{P}(\mathcal{O}_{\text{Call}} = \text{Call})$, and $\mathbf{P}(\neg\varphi_{>500}|\mathcal{O}_{\text{Call}} = \text{Call})$ (see Appendix A), we obtain:

$$\mathsf{PO}_r^A(\mathcal{A}_{\text{card}}, \varphi_{>500}, \mathcal{O}_{\text{Call}}) = \frac{28,272}{39,377} \simeq 0.718.$$

The notion of asymmetrical opacity, however, fails to capture security in terms of opacity for both φ and $\overline{\varphi}$. And so does the RPO measure. Therefore, we define in the next section a quantitative version of symmetrical opacity.

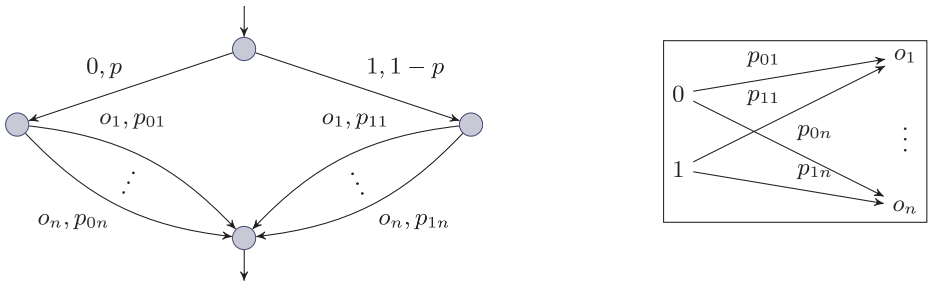
4.2. *Restricting symmetrical opacity*

Symmetrical opacity offers a sound framework to analyse the secret of a binary value. For example, consider a binary channel with n outputs. It can be modelled by a tree-like system branching on 0 and 1 at the first level, then branching on observables $\{o_1, \ldots, o_n\}$, as in Figure 5. If the system wishes to prevent communication, the secret of predicate 'the input of the channel was 1' is as important as the secret of its negation; in this case 'the input of the channel was 0'. Such case is an example of *initial opacity* (Bryans *et al.* 2008), since the secret appears only at the start of each run. Note that any system with initial opacity and a finite set of observables can be transformed into a channel (Andrés *et al.* 2010), with input distribution $(p,\ 1 - p)$, which is the distribution of the secret predicate over $\{\varphi, \overline{\varphi}\}$.

4.2.1. *Definition and properties.*

Symmetrical opacity ensures that for each observation class o (reached by a run), the probability of both $\mathbf{P}(\varphi \mid o)$ and $\mathbf{P}(\overline{\varphi} \mid o)$ is strictly above 0. That means that the lower of these probabilities should be above 0. In turn, the lowest of these probability is exactly the complement of the vulnerability (since $\mathbf{1}_\varphi$ can take only two values). That is, the security is measured with the probability of error in one guess (inside a given observation class). Hence, a system will be secure if, in each observation class, φ is *balanced* with $\overline{\varphi}$.

Definition 4.3 (restrictive probabilistic symmetric opacity). Let φ be a predicate on the complete runs of an FPA $\mathcal{A}$ and $\mathcal{O}$ an observation function. The *restrictive probabilistic*



(a) A system with initial secret (b) Channel with binary input

Fig. 5. (Colour online) A system and its associated channel.

symmetric opacity (RPSO) of φ on $\mathcal{A}$, with respect to $\mathcal{O}$, is defined by

$$\mathsf{PO}_r^S(\mathcal{A}, \varphi, \mathcal{O}) = \frac{-1}{\sum_{o \in Obs} \mathbf{P}(\mathcal{O} = o) \cdot \log\left(1 - V(\mathbf{1}_\varphi \mid \mathcal{O} = o)\right)}$$

where $V(\mathbf{1}_\varphi \mid \mathcal{O} = o) = \max_{i \in \{0,1\}} \mathbf{P}(\mathbf{1}_\varphi = i \mid \mathcal{O} = o)$.

Remark that the definition of RPSO has very few ties with the definition of RPO. Indeed, it is linked more with the notion of possibilistic symmetrical opacity than with the notion of quantitative asymmetrical opacity, and thus RPSO is not to be seen as an extension of RPO.

In the definition of RPSO, taking $-\log(1 - V(\mathbf{1}_\varphi \mid \mathcal{O} = o))$ allows to give more weight to very imbalanced classes, up to infinity for classes completely included either in φ or in $\overline{\varphi}$. Along the lines of Smith (2009), the logarithm is used in order to produce a measure in terms of bits instead of probabilities. These measures are then averaged with respect to the probability of each observation class. The final inversion ensures that the value is between 0 and 1, and can be seen as a normalization operation. The above motivations for the definition of RPSO directly yield the following properties:

Proposition 4.4.

1. $0 \leqslant \mathsf{PO}_r^S(\mathcal{A}, \varphi, \mathcal{O}) \leqslant 1$.
2. $\mathsf{PO}_r^S(\mathcal{A}, \varphi, \mathcal{O}) = 0$ if and only if φ is *not* symmetrically opaque on $unProb(\mathcal{A})$ with respect to $unProb(\mathcal{O})$.
3. $\mathsf{PO}_r^S(\mathcal{A}, \varphi, \mathcal{O}) = 1$ if and only if $\forall o \in Obs,\ \mathbf{P}(\mathbf{1}_\varphi = 1 \mid \mathcal{O} = o) = \frac{1}{2}$.

Proof of Proposition 4.4.

1. Since the vulnerability of a random variable that takes only two values is between $\frac{1}{2}$ and 1, we have $1 - V(\mathbf{1}_\varphi \mid \mathcal{O} = o) \in [0, \frac{1}{2}]$ for all $o \in Obs$. So $-\log(1 - V(\mathbf{1}_\varphi \mid \mathcal{O} = o)) \in [1, +\infty[$ for any o. The (arithmetic) means of these values is thus contained within the same bounds. The inversion therefore yields a value between 0 and 1.
2. If φ is not symmetrically opaque, then for some observation class o, $\mathcal{O}^{-1}(o) \subseteq \varphi$ or $\mathcal{O}^{-1}(o) \subseteq \overline{\varphi}$. In both cases, $V(\mathbf{1}_\varphi \mid \mathcal{O}) = 1$, so $-\log(1 - V(\mathbf{1}_\varphi \mid \mathcal{O} = o)) = +\infty$ and the average is also $+\infty$. Taking the limit for the inverse gives the value 0 for RPSO.

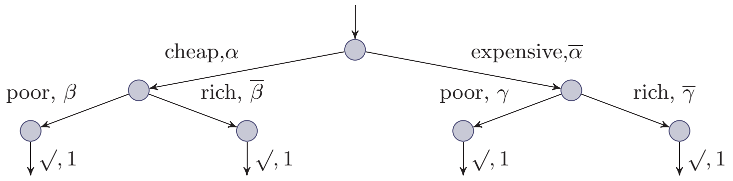


Fig. 6. (Colour online) A simple sale protocol represented as an FPA $\mathcal{S}ale$.

Conversely, if RPSO is 0, then its inverse is $+\infty$, which can only occur if one of the $-\log(1 - V(\mathbf{1}_\varphi \mid \mathcal{O} = o))$ is $+\infty$ for some o. This, in turn, means that some $V(\mathbf{1}_\varphi \mid \mathcal{O} = o)$ is 1, which means the observation class of o is contained either in φ or in $\overline{\varphi}$.

3. $\mathsf{PO}_r^S(\mathcal{A}, \varphi, \mathcal{O}) = 1$ iff $\Sigma_{o \in Obs}\mathbf{P}(\mathcal{O} = o) \cdot \left(-\log\left(1 - V(\mathbf{1}_\varphi \mid \mathcal{O} = o)\right)\right) = 1$. Since this is an average of values above 1, this is equivalent to $-\log\left(1 - V(\mathbf{1}_\varphi \mid \mathcal{O} = o)\right) = 1$ for all $o \in Obs$, *i.e.* $V(\mathbf{1}_\varphi \mid \mathcal{O} = o) = \frac{1}{2}$ for all o. In this particular case, we also have $V(\mathbf{1}_\varphi \mid \mathcal{O} = o) = \frac{1}{2}$ iff $\mathbf{P}(\mathbf{1}_\varphi = 1 \mid \mathcal{O} = o) = \frac{1}{2}$ which concludes the proof.

□

4.2.2. *Example 1: sale protocol.* We consider the sale protocol from Alvim *et al.* (2010), depicted in Figure 6. Two products can be put on sale, either a cheap or an expensive one, and two clients, either a rich or a poor one, may want to buy it. The products are put on sale according to a distribution (α and $\overline{\alpha} = 1 - \alpha$) while buyers behave probabilistically (through β and γ) although differently according to the price of the item on sale. The price of the item is public, but the identity of the buyer should remain secret. Hence the observation function *Price* yields *cheap* or *expensive*, while the secret is, without loss of symmetry, the set φ_{poor} of runs ending with *poor*. The bias introduced by the preference of, say, a cheap item by the poor client betrays the secret identity of the buyer. RPSO allows to measure this bias, and more importantly, to compare the bias obtained globally for different values of the parameters α, β, and γ.

More formally, we have:

$$\mathbf{P}(Price = \text{cheap}) = \alpha \qquad \mathbf{P}(Price = \text{expensive}) = \overline{\alpha}$$

$$V(\mathbf{1}_{\varphi_{\text{poor}}} \mid \mathcal{O} = \text{cheap}) = \max(\beta, \overline{\beta}) \qquad V(\mathbf{1}_{\varphi_{\text{poor}}} \mid \mathcal{O} = \text{expensive}) = \max(\gamma, \overline{\gamma})$$

$$\mathsf{PO}_r^S(\mathcal{S}ale, \varphi_{\text{poor}}, Price) = \frac{-1}{\alpha \cdot \log(\min(\beta, \overline{\beta})) + \overline{\alpha} \cdot \log(\min(\gamma, \overline{\gamma}))}.$$

The variations of RPSO w.r.t. to β and γ is depicted for several values of α in Figure 7, red meaning higher value for RPSO. Thus, while the result is symmetric for $\alpha = \frac{1}{2}$, the case where $\alpha = \frac{1}{8}$ gives more importance to the fluctuations of γ.

4.2.3. *Example 2: dining cryptographers protocol.* Introduced in Chaum (1988), this problem involves three cryptographers C_1, C_2 and C_3 dining in a restaurant. At the end of the meal, their master secretly tells each of them if they should be paying: $p_i = 1$ iff

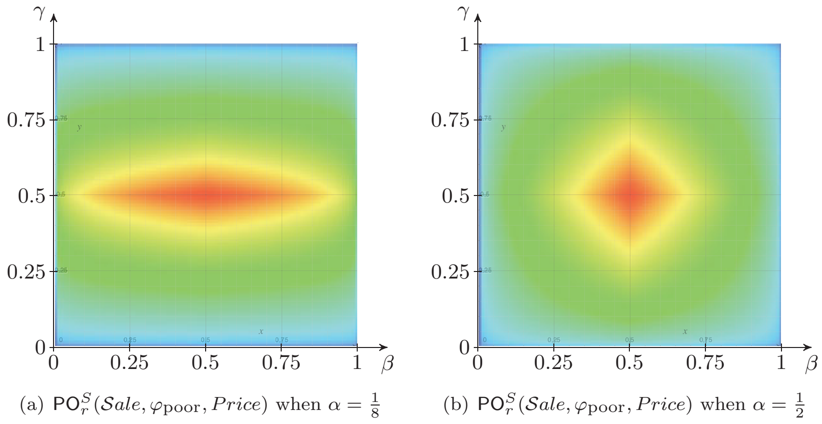


Fig. 7. (Colour online) RPSO for the sale protocol.

cryptographer C_i pays, and $p_i = 0$ otherwise. Wanting to know if one of the cryptographers paid or if the master did, they follow the following protocol. They flip a coin with each of their neighbour, the third one not seeing the result of the flip, marking $f_{i,j} = 0$ if the coin flip between i and j was heads and $f_{i,j} = 1$ if it was tails. Then each cryptographer C_i, for $i \in \{1, 2, 3\}$, announces the value of $r_i = f_{i,i+1} \oplus f_{i,i-1} \oplus p_i$ (where '$3 + 1 = 1$', '$1 - 1 = 3$' and '$\oplus$' represents the XOR operator). If $\bigoplus_{i=1}^{3} r_i = 0$ then no one (*i.e.* the master) paid, if $\bigoplus_{i=1}^{3} r_i = 1$, then one of the cryptographers paid, but the other two do not know who he is.

Here we will use a simplified version of this problem to limit the size of the model. We consider that some cryptographer paid for the meal, and adopt the point of view of C_1 who did not pay. The anonymity of the payer is preserved if C_1 cannot know if C_2 or C_3 paid for the meal. In our setting, the predicate φ_2 is, without loss of symmetry, 'C_2 paid'. Note that predicate φ_2 is well suited for analysis of symmetrical opacity, since detecting that φ_2 is false gives information on who paid (here C_3). The observation function lets C_1 know the results of its coin flips ($f_{1,2}$ and $f_{1,3}$), and the results announced by the other cryptographers (r_2 and r_3). We also assume that the coin used by C_2 and C_3 has a probability of q to yield heads, and that the master flips a fair coin to decide if C_2 or C_3 pays. It can be assumed that the coins C_1 flips with its neighbours are fair, since it does not affect anonymity from C_1's point of view. In order to limit the (irrelevant) interleaving, we have made the choice to fix the ordering between the coin flips.

The corresponding FPA $\mathcal{D}$ is depicted on Figure 8, where all $\surd$ transitions with probability 1 have been omitted from final (rectangular) states. On $\mathcal{D}$, the runs satisfying predicate φ_2 are the ones where action p_2 appears. The observation function $\mathcal{O}_1$ takes a run and returns the sequence of actions over the alphabet $\{h_{1,2}, t_{1,2}, h_{1,3}, t_{1,3}\}$ and the final state reached, containing the value announced by C_2 and C_3.

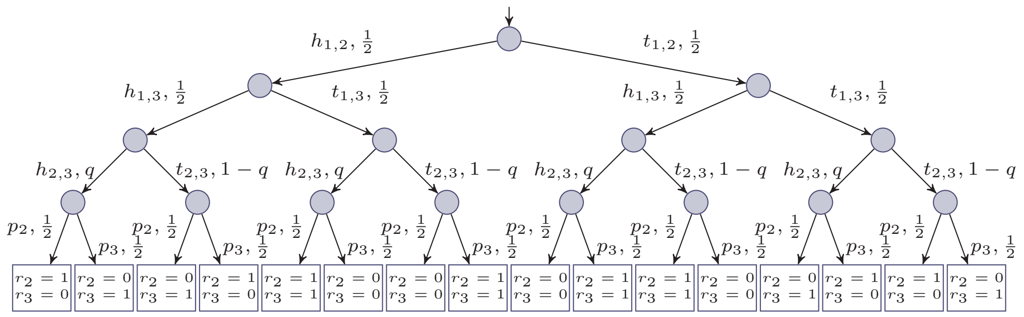


Fig. 8. (Colour online) The FPA corresponding to the Dining Cryptographers protocol.

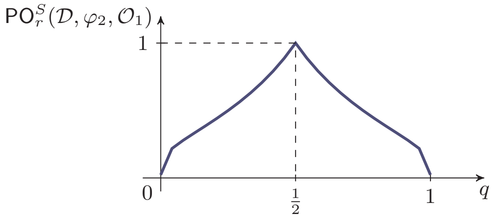


Fig. 9. (Colour online) Evolution of the restrictive probabilistic symmetric opacity of the dining cryptographers protocol when changing the bias on the coin.

There are 16 possible complete runs in this system, that yield 8 equiprobable observables:

$$\begin{aligned} Obs = \{&(h_{1,2}h_{1,3}(r_2 = 1, r_3 = 0)), \quad (h_{1,2}h_{1,3}(r_2 = 0, r_3 = 1)),\\ &(h_{1,2}t_{1,3}(r_2 = 0, r_3 = 0)), \quad (h_{1,2}t_{1,3}(r_2 = 1, r_3 = 1)),\\ &(t_{1,2}h_{1,3}(r_2 = 0, r_3 = 0)), \quad (t_{1,2}h_{1,3}(r_2 = 1, r_3 = 1)),\\ &(t_{1,2}t_{1,3}(r_2 = 1, r_3 = 0)), \quad (t_{1,2}t_{1,3}(r_2 = 0, r_3 = 1))\}. \end{aligned}$$

Moreover, each observation results in a run in which C_2 pays and a run in which C_3 pays, this difference being masked by the secret coin flip between them. For example, runs $\rho_h = h_{1,2}h_{1,3}h_{2,3}p_2(r_2 = 1, r_3 = 0)$ and $\rho_t = h_{1,2}h_{1,3}t_{2,3}p_3(r_2 = 1, r_3 = 0)$ yield the same observable $o_0 = h_{1,2}h_{1,3}(r_2 = 1, r_3 = 0)$, but the predicate is true in the first case and false in the second one. Therefore, if $0 < q < 1$, the unprobabilistic version of $\mathcal{D}$ is opaque. However, if $q \neq \frac{1}{2}$, for each observable, one of them is *more likely* to be lying, therefore paying. In the aforementioned example, when observing o_0, ρ_h has occurred with probability q, whereas ρ_t has occurred with probability $1 - q$. RPSO can measure this advantage globally.

For each observation class, the vulnerability of φ_2 is $\max(q, 1 - q)$. Hence the RPSO will be

$$\mathsf{PO}_r^S(\mathcal{D}, \varphi_2, \mathcal{O}_1) = \frac{-1}{\log(\min(q, 1 - q))}$$

The variations of the RPSO when changing the bias on q are depicted in Figure 9. Analysis of RPSO according to the variation of q yields that the system is perfectly secure if there is no bias on the coin, and insecure if $q = 0$ or $q = 1$.

5. Computing opacity measures

We now show how all measures defined above can be computed for regular predicates and simple observation functions. The method relies on a synchronized product between an SA $\mathcal{A}$ and a deterministic FA $\mathcal{K}$, similarly to Courcoubetis and Yannakakis (1998). This product (which can be considered pruned of its unreachable states and states not reaching a final state) constrains the unprobabilistic version of $\mathcal{A}$ by synchronizing it with $\mathcal{K}$. The probability of $\mathcal{L}(\mathcal{K})$ is then obtained by solving a system of equations associated with this product. The computation of all measures results in applications of this operation with several automata.

5.1. *Computing the probability of a substochastic automaton*

Given an SA $\mathcal{A}$, a system of equations can be derived on the probabilities for each state to yield an accepting run. This allows to compute the probability of all complete runs of $\mathcal{A}$ by a technique similar to those used in Courcoubetis and Yannakakis (1998); Hansson and Jonsson (1994); Bianco and de Alfaro (1995) for probabilistic verification.

Definition 5.1 (linear system of a substochastic automata). Let $\mathcal{A} = \langle \Sigma, Q, \Delta, q_0 \rangle$ be a substochastic automaton where any state can reach a final state. The *linear system associated with* $\mathcal{A}$ is the following system $\mathcal{S}_\mathcal{A}$ of linear equations over $\mathbb{R}$:

$$\mathcal{S}_\mathcal{A} = \left(X_q = \sum_{q' \in Q} \alpha_{q,q'} X_{q'} + \beta_q \right)_{q \in Q}$$

$$\text{where} \quad \alpha_{q,q'} = \sum_{a \in \Sigma} \Delta(q)(a,q') \text{ and } \beta_q = \Delta(q)(\surd).$$

When non-determinism is involved, for instance in Markov decision processes (Courcoubetis and Yannakakis 1998; Bianco and de Alfaro 1995), two systems of inequations are needed to compute maximal and minimal probabilities. Here, without non-determinism, both values are the same, hence Lemma 5.2 is a particular case of the results in Courcoubetis and Yannakakis (1998); Bianco and de Alfaro (1995), where uniqueness is ensured by the hypothesis (any state can reach a final state). The probability can thus be computed in polynomial time by solving the linear system associated with the SA.

Lemma 5.2. Let $\mathcal{A} = \langle \Sigma, Q, \Delta, q_0 \rangle$ be a substochastic automaton and define for all $q \in Q$, $L_q^\mathcal{A} = \mathbf{P}(CRun_q(\mathcal{A}))$. Then $(L_q^\mathcal{A})_{q \in Q}$ is the unique solution of the system $\mathcal{S}_\mathcal{A}$.

5.2. *Computing the probability of a regular language*

In order to compute the probability of a language inside a system, we build a substochastic automaton that corresponds to the intersection of the system and the language, then compute the probability as above.

Definition 5.3 (synchronized product). Let $\mathcal{A} = \langle \Sigma, Q, \Delta, q_0 \rangle$ be a substochastic automaton and let $\mathcal{K} = \langle Q \times \Sigma \times Q, Q_K, \Delta_K, q_K, F \rangle$ be a deterministic finite automaton. The synchronized product $\mathcal{A}||\mathcal{K}$ is the substochastic automaton $\langle \Sigma, Q \times Q_K, \Delta', (q_0, q_K) \rangle$ where transitions in Δ' are defined by: if $q_1 \to \mu \in \Delta$, then $(q_1, r_1) \to \nu \in \Delta'$ where for all $a \in \Sigma$ and $(q_2, r_2) \in Q \times Q_K$,

$$\nu(a, (q_2, r_2)) = \begin{cases} \mu(a, q_2) & \text{if } r_1 \xrightarrow{q_1, a, q_2} r_2 \in \Delta_K \\ 0 & \text{otherwise} \end{cases}$$

$$\text{and} \quad \nu(\surd) = \begin{cases} \mu(\surd) & \text{if } r_1 \in F \\ 0 & \text{otherwise} \end{cases}$$

In this synchronized product, the behaviours are constrained by the finite automaton. Actions not allowed by the automaton are trimmed, and states can accept only if they correspond to a valid behaviour of the DFA. Note that this product is defined on SA in order to allow several intersections. The correspondence between the probability of a language in a system and the probability of the synchronized product is laid out in the following lemma.

Lemma 5.4. Let $\mathcal{A} = \langle \Sigma, Q, \Delta, q_0 \rangle$ be an SA and K a regular language over $Q \times \Sigma \times Q$ accepted by a deterministic finite automaton $\mathcal{K} = \langle Q \times \Sigma \times Q, Q_K, \Delta_K, q_K, F \rangle$. Then

$$\mathbf{P}_{\mathcal{A}}(K) = L^{\mathcal{A}||\mathcal{K}}_{(q_0, q_K)}.$$

Proof. Let $\rho \in CRun(\mathcal{A})$ with $\mathrm{tr}(\rho) \in K$ and $\rho = q_0 \xrightarrow{a_1} q_1 \cdots \xrightarrow{a_n} q_n \surd$. Since $\mathrm{tr}(\rho) \in K$ and $\mathcal{K}$ is deterministic, there is a unique run $\rho_K = q_K \xrightarrow{a_1} r_1 \cdots \xrightarrow{a_n} r_n$ in $\mathcal{K}$ with $r_n \in F$. Then the sequence $\rho' = (q_0, q_K) \xrightarrow{a_1} (q_1, r_1) \cdots \xrightarrow{a_n} (q_n, r_n)$ is a run of $\mathcal{A}||\mathcal{K}$. There is a one-to-one match between runs of $\mathcal{A}||\mathcal{K}$ and pairs of runs in $\mathcal{A}$ and $\mathcal{K}$ with the same trace. Moreover,

$$\begin{aligned} \mathbf{P}_{\mathcal{A}||\mathcal{K}}(\rho') &= \Delta'(q_0, q_K)(a_1, (q_1, r_1)) \times \cdots \times \Delta'(q_n, r_n)(\surd) \\ &= \Delta(q_0)(a_1, q_1) \times \cdots \times \Delta(q_n)(\surd) \\ \mathbf{P}_{\mathcal{A}||\mathcal{K}}(\rho') &= \mathbf{P}_{\mathcal{A}}(\rho). \end{aligned}$$

Hence

$$\mathbf{P}_{\mathcal{A}}(K) = \sum_{\{\rho | \mathrm{tr}(\rho) \in K\}} \mathbf{P}_{\mathcal{A}}(\rho) = \sum_{\rho' \in Run(\mathcal{A}||\mathcal{K})} \mathbf{P}_{\mathcal{A}||\mathcal{K}}(\rho') = \mathbf{P}_{\mathcal{A}||\mathcal{K}}(Run(\mathcal{A}||\mathcal{K}))$$

and therefore from Lemma 5.2, $\mathbf{P}_{\mathcal{A}}(K) = L^{\mathcal{A}||\mathcal{K}}_{(q_0, q_0')}$. □

5.3. *Computing all opacity measures*

All measures defined previously can be computed as long as, for $i \in \{0, 1\}$ and $o \in Obs$, all probabilities

$$\mathbf{P}(\mathbf{1}_{\varphi} = i) \qquad \mathbf{P}(\mathcal{O} = o) \qquad \mathbf{P}(\mathbf{1}_{\varphi} = i, \mathcal{O} = o)$$

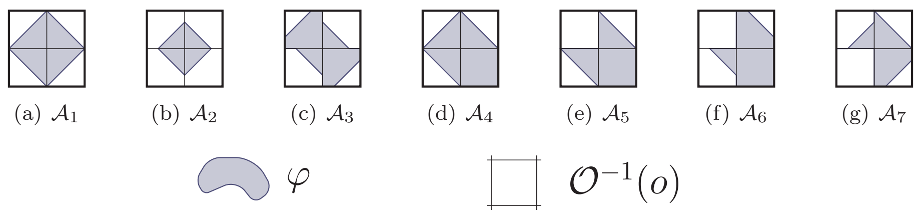


Fig. 10. (Colour online) Example of repartition of probabilities of $\mathbf{1}_\varphi$ and $\mathcal{O}$ in 7 cases.

can be computed. Indeed, even deciding whether $\mathcal{O}^{-1}(o) \subseteq \varphi$ can be done by testing $\mathbf{P}(\mathcal{O} = o) > 0 \wedge \mathbf{P}(\mathbf{1}_\varphi = 0, \mathcal{O} = o) = 0$.

Now suppose *Obs* is a finite set, φ and all $\mathcal{O}^{-1}(o)$ are regular sets. Then one can build deterministic finite automata $\mathcal{A}_\varphi$, $\mathcal{A}_{\overline{\varphi}}$, $\mathcal{A}_o$ for $o \in Obs$ that accept respectively φ, $\overline{\varphi}$, and $\mathcal{O}^{-1}(o)$.

Synchronizing automaton $\mathcal{A}_\varphi$ with $\mathcal{A}$ and pruning it yields a substochastic automaton $\mathcal{A}||\mathcal{A}_\varphi$. By Lemma 5.4, the probability $\mathbf{P}(\mathbf{1}_\varphi = 1)$ is then computed by solving the linear system associated with $\mathcal{A}||\mathcal{A}_\varphi$. Similarly, one obtain $\mathbf{P}(\mathbf{1}_\varphi = 0)$ (with $\mathcal{A}_{\overline{\varphi}}$), $\mathbf{P}(\mathcal{O} = o)$ (with $\mathcal{A}_o$), $\mathbf{P}(\mathbf{1}_\varphi = 1, \mathcal{O} = o)$ (synchronizing $\mathcal{A}||\mathcal{A}_\varphi$ with $\mathcal{A}_o$), and $\mathbf{P}(\mathbf{1}_\varphi = 0, \mathcal{O} = o)$ (synchronizing $\mathcal{A}||\mathcal{A}_{\overline{\varphi}}$ with $\mathcal{A}_o$).

Theorem 5.5. Let $\mathcal{A}$ be an FPA. If *Obs* is a finite set, φ is a regular set and for $o \in Obs$, $\mathcal{O}^{-1}(o)$ is a regular set, then for $\mathsf{PO} \in \{\mathsf{PO}_\ell^A, \mathsf{PO}_\ell^S, \mathsf{PO}_r^A, \mathsf{PO}_r^S\}$, $\mathsf{PO}(\mathcal{A}, \varphi, \mathcal{O})$ can be computed.

The computation of opacity measures is done in polynomial time in the size of *Obs* and DFAs $\mathcal{A}_\varphi$, $\mathcal{A}_{\overline{\varphi}}$, $\mathcal{A}_o$.

A prototype tool implementing this algorithm was developed in Java (Eftenie 2010), yielding numerical values for measures of opacity.

6. Comparison of the measures of opacity

In this section, we compare the discriminating power of the measures discussed above. As described above, the liberal measures evaluate the leak, hence 0 represents the best possible value from a security point of view, producing an opaque system. For such an opaque system, the restrictive measure evaluate the robustness of this opacity. As a result, 1 is the best possible value.

6.1. *Abstract examples*

The values of these metrics are first compared for extremal cases of Figure 10. These values are displayed in Table 3.

First, the system $\mathcal{A}_1$ of Figure 10a is intuitively very secure since, with or without observation, an attacker has no information whether φ was true or not. This security is reflected in all symmetrical measures, with highest scores possible in all cases. It is nonetheless deemed more insecure for RPO, since opacity is perfect when φ is always false.

Table 3. *Values of the different opacity measures for systems of Figure 10a–g.*

System	(a) $\mathcal{A}_1$	(b) $\mathcal{A}_2$	(c) $\mathcal{A}_3$	(d) $\mathcal{A}_4$	(e) $\mathcal{A}_5$	(f) $\mathcal{A}_6$	(g) $\mathcal{A}_7$
LPO	0	0	0	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	0
LPSO	0	0	0	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{4}$
RPO	$\frac{1}{2}$	$\frac{3}{4}$	$\frac{3}{8}$	0	0	0	$\frac{12}{25}$
RPSO	1	$\frac{1}{2}$	$\frac{1}{2}$	0	0	0	0

The case of $\mathcal{A}_2$ of Figure 10b only differs from $\mathcal{A}_1$ by the global repartition of φ in $Run(\mathcal{A})$. The information an attacker gets comes not from the observation, but from φ itself. Therefore RPSO, which does not remove the information available before observation, evaluates this system as less secure than $\mathcal{A}_1$. Measures based on the information theory (Smith 2009; Bérard *et al.* 2010) would consider this system as secure. However, such measures lack strong ties with opacity, which depend only on the information available to the observer, wherever this information comes from. In addition, RPO finds $\mathcal{A}_2$ more secure than $\mathcal{A}_1$: φ is verified less often. Note that the complement would not change the value for symmetrical measures, while being insecure for RPO (with $\mathsf{PO}_r^A = \frac{1}{4}$).

However, since each observation class is considered individually, RPSO does not discriminate $\mathcal{A}_2$ and $\mathcal{A}_3$ of Figure 10c. Here, the information is the same in each observation class as for $\mathcal{A}_2$, but the repartition of φ gives no advantage at all to an attacker without observation.

When the system is not opaque (resp. symmetrically opaque), RPO (resp. RPSO) cannot discriminate them, and LPO (resp. LPSO) becomes relevant. For example, $\mathcal{A}_4$ is not opaque for the classical definitions, therefore $\mathsf{PO}_r^A = \mathsf{PO}_r^S = 0$ and both $\mathsf{PO}_\ell^A > 0$ and $\mathsf{PO}_\ell^S > 0$.

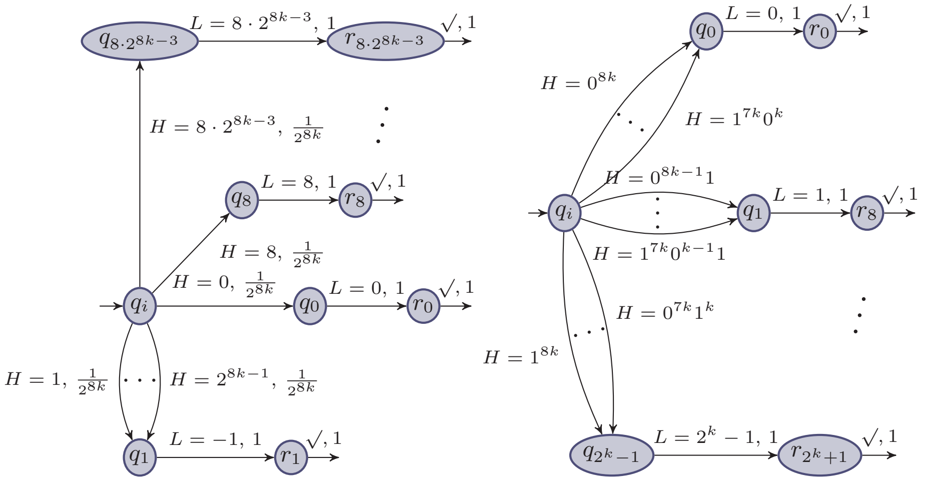
System $\mathcal{A}_5$ of Figure 10e has a greater PO_ℓ^S than $\mathcal{A}_4$. However, LPO is unchanged since the class completely out of φ is not taken into account. Remark that system $\mathcal{A}_7$ is opaque but not symmetrically opaque, hence the relevant measures are PO_ℓ^S and PO_r^A. Also note that once a system is not opaque, the repartition of classes that do not leak information is not taken into account, hence equal values in the cases of $\mathcal{A}_5$ and $\mathcal{A}_6$.

6.2. *A more concrete example*

Consider the following programs P_1 and P_2, inspired from Smith (2009), where k is a given parameter, `random` selects uniformly an integer value (in binary) between its two arguments and & is the bitwise *and*:

```
P_1: $H$ := random(0, $2^{8k} - 1$);
     if $H$ mod 8 = 0 then
       $L := H$
     else
       $L := -1$
     fi
```

```
P_2: $H$ := random(0, $2^{8k} - 1$);
     $L := H \;\&\; 0^{7k}1^k$
```



(a) FPA $\mathcal{A}_{P_1}$ (b) FPA $\mathcal{A}_{P_2}$; all edges stemming from q_i have equal probability $\frac{1}{2^{8k}}$

Fig. 11. (Colour online) FPAs for programs P_1 and P_2.

In both cases, the value of H, an integer over $8k$ bits, is supposed to remain secret, and cannot be observed directly, while the value of L is public. Thus the observation is the '$L := \ldots$' action. Intuitively, P_1 divulges the exact value of H with probability $\frac{1}{8}$. On the other hand, P_2 leaks the value of one eighth of its bits (the least significant ones) at every execution.

These programs can be translated into FPAs $\mathcal{A}_{P_1}$ and $\mathcal{A}_{P_2}$ of Figure 11. In order to have a boolean predicate, the secret is not the value of variable H, but whether $H = L$: $\varphi_= = \{(H = x)(L = x) \mid x \in \{0, \ldots, 2^{8k} - 1\}\}$. Non-opacity then means that the attacker discovers the secret value. Weaker predicates can also be considered, like equality of H with a particular value or H belonging to a specified subset of values, but we chose the simplest one. First remark that $\varphi_=$ is not opaque on P_1 in the classical sense (both symmetrically or not). Hence both RPO and RPSO are null. On the other hand, $\varphi_=$ is opaque on P_2, hence LPO and LPSO are null. The values for all measures are gathered in Table 4. Note that only restrictive opacity for P_2 depends on k. This comes from the fact that in all other cases, both $\varphi_=$ and the equivalence classes scale at the same rate with k. In the case of P_2, adding length to the secret variable H *dilutes* $\varphi_=$ inside each class. Hence the greater k is, the hardest it is for an attacker to know that $\varphi_=$ is true, thus to crack asymmetrical opacity. Indeed, it will tend to get false in most cases, thus providing an easy guess, and a low value for symmetrical opacity.

6.3. *Crowds protocol*

The anonymity protocol known as *crowds* was introduced in Reiter and Rubin (1998) and recently studied in the probabilistic framework in Chatzikokolakis *et al.* (2008) and

Table 4. *Opacity measures for programs P_1 and P_2.*

Program	PO_ℓ^A	PO_ℓ^S	PO_r^A	PO_r^S
P_1	$\frac{1}{8}$	1	0	0
P_2	0	0	$1-\frac{1}{2^{7k}}$	$\frac{1}{7k}$

Andrés *et al.* (2010). When a user wants to send a message (or request) to a server without the latter knowing the origin of the message, the user routes the message through a crowd of n users. To do so, it selects a user randomly in the crowd (including himself), and sends him the message. When a user receives a message to be routed according to this protocol, it either sends the message to the server with probability $1-q$ or forwards it to a user in the crowd, with probability q. The choice of a user in the crowd is always equiprobable. Under these assumptions, this protocol is known to be secure, since no user is more likely than another to be the actual initiator; indeed its RPO is very low. However, there can be c corrupt users in the crowd which divulge the identity of the person that sent the message to them. In that case, if a user sends directly a message to a corrupt user, its identity is no longer protected. The goal of corrupt users is therefore not to transmit messages, hence they cannot initiate the protocol. The server and the corrupt users cooperate to discover the identity of the initiator. RPO can measure the security of this system, depending on n and c.

First, consider our protocol as the system in Figure 12. In this automaton, states $1,\ldots,n-c$ corresponding to honest users are duplicated in order to differentiate their behaviour as initiator or as the receiver of a message from the crowd. The predicate we want to be opaque is φ_i that contains all the runs in which i is the initiator of the request. The observation function $\mathcal{O}$ returns the penultimate state of the run, *i.e.* the honest user that will be seen by the server or a corrupt user.

For sake of brevity, we write '$i \rightsquigarrow$' to denote the event 'a request was initiated by i' and '$\rightsquigarrow j$' when 'j was detected by the adversary', which means that j sent the message either to a corrupt user or to the server, who both try to discover who the initiator was. The abbreviation $i \rightsquigarrow j$ stands for $i \rightsquigarrow \wedge \rightsquigarrow j$. Notation '$\neg i \rightsquigarrow$' means that 'a request was initiated by someone else than i'; similarly, combinations of this notations are used in the sequel. We also use the Kronecker symbol δ_{ij} defined by $\delta_{ij}=1$ if $i=j$ and 0 otherwise.

6.3.1. *Computation of the probabilities.* All probabilities $\mathbf{P}(i \rightsquigarrow j)$ can be automatically computed using the method described in Section 5. For example, $\mathbf{P}(1 \rightsquigarrow (n-c))$, the probability for the first user to initiate the protocol while the last honest user is detected, can be computed from substochastic automaton $\mathcal{C}_n^c||\mathcal{A}_{1\rightsquigarrow(n-c)}$ depicted on Figure 13. In this automaton, the only duplicated state remaining is $1'$.

This SA can also be represented by a transition matrix (like a Markov chain), which is given in Table 5. An additional column indicates the probability for the $\surd$ action, which ends the run (here it is either 1 if the state is final and 0 if not).

The associated system is represented in Table 6 where L_S corresponds to the 'Server' state. Each line of the system is given by the outgoing probabilities of the corresponding

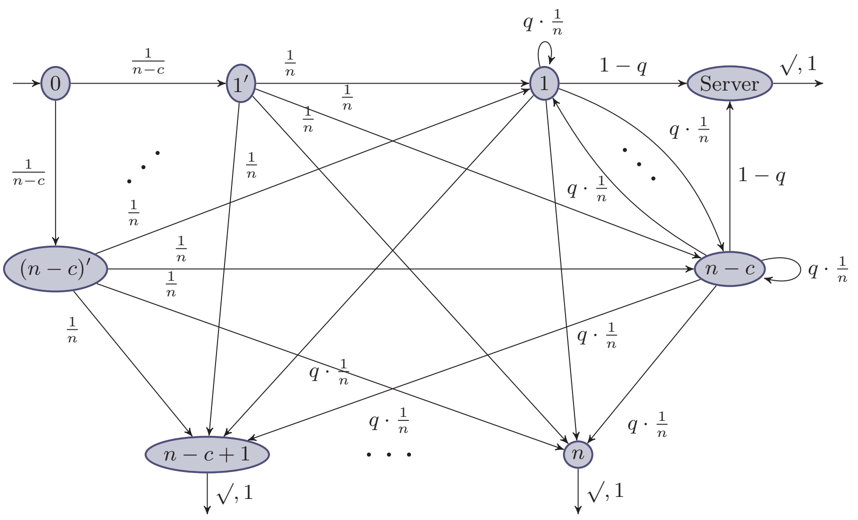


Fig. 12. (Colour online) FPA $\mathcal{C}_n^c$ for Crowds protocol with n users, among whom c are corrupted.

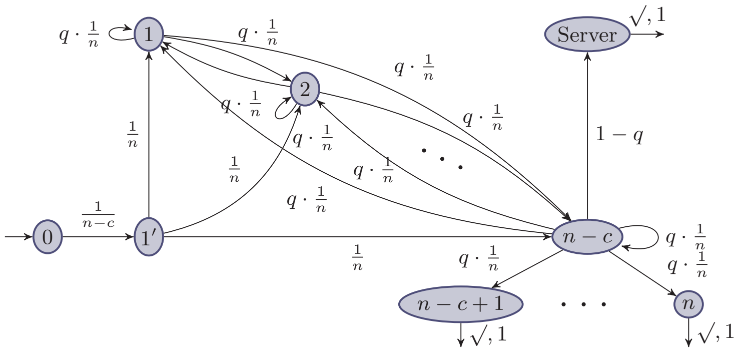


Fig. 13. (Colour online) SA $\mathcal{C}_n^c || \mathcal{A}_{1 \rightsquigarrow (n-c)}$ corresponding to runs where user 1 initiates the protocol and user $(n-c)$ is detected.

state in the SA, or alternatively by the corresponding line of the matrix. Resolving it (see Appendix B) yields, $L_i = \frac{q}{n}$ for all $i \in \{1, \ldots, n-c-1\}$, $L_{n-c} = 1 - \frac{q \cdot (n-c-1)}{n}$, $L_{1'} = \frac{1}{n}$, and $L_0 = \frac{1}{(n-c) \cdot n}$. Therefore, $\mathbf{P}(1 \rightsquigarrow (n-c)) = \frac{1}{(n-c) \cdot n}$.

In this case, simple reasoning on the symmetries of the model allows to derive other probabilities $\mathbf{P}(i \rightsquigarrow j)$. Remark that the probability for a message to go directly from initiator to the adversary (who cannot be the server) is $\frac{c}{n}$: it only happens if a corrupt user is chosen by the initiator. If a honest user is chosen by the initiator, then the length of the path will be greater, with probability $\frac{n-c}{n}$. By symmetry all honest users have equal probability to be the initiator, and equal probability to be detected. Hence $\mathbf{P}(i \rightsquigarrow) = \mathbf{P}(\rightsquigarrow j) = \frac{1}{n-c}$.

Table 5. *The matrix giving the transition probabilities between the states of* $\mathcal{C}_n^c||\mathcal{A}_{1\rightsquigarrow(n-c)}$.

	0	$1'$	1	$\cdots$	$n-c$	$n-c+1$	$\cdots$	n	*Server*	$\surd$
0	0	$\frac{1}{n-c}$	0	$\cdots$	0	0	$\cdots$	0	0	0
$1'$	0	0	$\frac{1}{n}$	$\cdots$	$\frac{1}{n}$	0	$\cdots$	0	0	0
1	0	0	$q\cdot\frac{1}{n}$	$\cdots$	$q\cdot\frac{1}{n}$	0	$\cdots$	0	0	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$
$n-c-1$	0	0	$q\cdot\frac{1}{n}$	$\cdots$	$q\cdot\frac{1}{n}$	0	$\cdots$	0	0	0
$n-c$	0	0	$q\cdot\frac{1}{n}$	$\cdots$	$q\cdot\frac{1}{n}$	$q\cdot\frac{1}{n}$	$\cdots$	$q\cdot\frac{1}{n}$	$1-q$	0
$n-c+1$	0	0	0	$\cdots$	0	0	$\cdots$	0	0	1
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$
n	0	0	0	$\cdots$	0	0	$\cdots$	0	0	1
Server	0	0	0	$\cdots$	0	0	$\cdots$	0	0	1

Table 6. *Linear system associated to SA* $\mathcal{C}_n^c||\mathcal{A}_{1\rightsquigarrow(n-c)}$ *of Figure 13.*

$$\left\{\begin{array}{rcl} L_0 &=& \frac{1}{n-c}\cdot L_{1'} \\ L_{1'} &=& \sum_{i=1}^{n-c}\frac{1}{n}\cdot L_i \\ L_1 &=& \sum_{i=1}^{n-c}\frac{q}{n}\cdot L_i \\ &\vdots& \\ L_{n-c-1} &=& \sum_{i=1}^{n-c}\frac{q}{n}\cdot L_i \end{array}\right. \qquad \left\{\begin{array}{rcl} L_{n-c} &=& (1-q)\cdot L_S + \sum_{i=1}^{n}\frac{q}{n}\cdot L_i \\ L_{n-c+1} &=& 1 \\ &\vdots& \\ L_n &=& 1 \\ L_S &=& 1 \end{array}\right.$$

Event $i \rightsquigarrow j$ occurs when i is chosen as the initiator (probability $\frac{1}{n-c}$), and either (1) if $i = j$ and i chooses a corrupted user to route its message, or (2) if a honest user is chosen and j sends the message to a corrupted user or the server (the internal route between honest users before j is irrelevant). Therefore

$$\mathbf{P}(i \rightsquigarrow j) = \frac{1}{n-c}\cdot\left(\delta_{ij}\cdot\frac{c}{n} + \frac{1}{n-c}\cdot\frac{n-c}{n}\right)$$

$$\mathbf{P}(i \rightsquigarrow j) = \frac{1}{n-c}\cdot\left(\delta_{ij}\cdot\frac{c}{n} + \frac{1}{n}\right)$$

The case when i is not the initiator is derived from this probability:

$$\mathbf{P}(\neg i \rightsquigarrow j) = \sum_{\substack{k=1\\k\neq i}}^{n-c}\mathbf{P}(k \rightsquigarrow j)$$

$$\mathbf{P}(\neg i \rightsquigarrow j) = \frac{1}{n-c}\cdot\left((1-\delta_{ij})\cdot\frac{c}{n} + \frac{n-c-1}{n}\right)$$

Conditional probabilities thus follow:

$$\mathbf{P}(i \rightsquigarrow \mid \rightsquigarrow j) = \frac{\mathbf{P}(i \rightsquigarrow j)}{\mathbf{P}(\rightsquigarrow j)} = \delta_{ij} \cdot \frac{c}{n} + \frac{1}{n}$$

$$\mathbf{P}(\neg i \rightsquigarrow \mid \rightsquigarrow j) = \frac{\mathbf{P}(\neg i \rightsquigarrow j)}{\mathbf{P}(\rightsquigarrow j)} = (1 - \delta_{ij}) \cdot \frac{c}{n} + \frac{n-c-1}{n}$$

Interestingly, these probabilities do not depend on q[†].

6.3.2. *Computation of RPO.* From the probabilities above, one can compute an analytical value for $\mathsf{PO}_r^A(\mathcal{C}_n^c, \mathbf{1}_{\varphi_i}, \mathcal{O})$.

$$\begin{aligned}
\frac{1}{\mathsf{PO}_r^A(\mathcal{C}_n^c, \varphi_i, \mathcal{O})} &= \sum_{j=1}^{n-c} \mathbf{P}(\rightsquigarrow j) \frac{1}{\mathbf{P}(\neg i \rightsquigarrow \mid \rightsquigarrow j)} \\
&= (n-c-1) \cdot \frac{1}{n-c} \cdot \frac{n}{n-1} + \frac{1}{n-c} \cdot \frac{n}{n-c-1} \\
&= \frac{n}{n-c} \left(\frac{n-c-1}{n-1} + \frac{1}{n-c-1} \right) \\
\frac{1}{\mathsf{PO}_r^A(\mathcal{C}_n^c, \varphi_i, \mathcal{O})} &= \frac{n \cdot (n^2 + c^2 - 2nc - n + 2c)}{(n-c) \cdot (n-1) \cdot (n-c-1)}
\end{aligned}$$

Hence

$$\mathsf{PO}_r^A(\mathcal{C}_n^c, \varphi_i, \mathcal{O}) = \frac{(n-c) \cdot (n-1) \cdot (n-c-1)}{n \cdot (n^2 + c^2 - 2nc - n + 2c)}$$

which tends to 1 as n increases to $+\infty$ (for a fixed number of corrupted users). The evolution of RPO is represented in Figure 14a where blue means low and red means high. If the proportion of corrupted users is fixed, say $n = 4c$, we obtain

$$\mathsf{PO}_r^A(\mathcal{C}_{4c}^c, \varphi_i, \mathcal{O}) = \frac{(4c-1) \cdot (9c-3)}{4c \cdot (9c-2)}$$

which also tends to 1 as the crowds size increases. When there are no corrupted users,

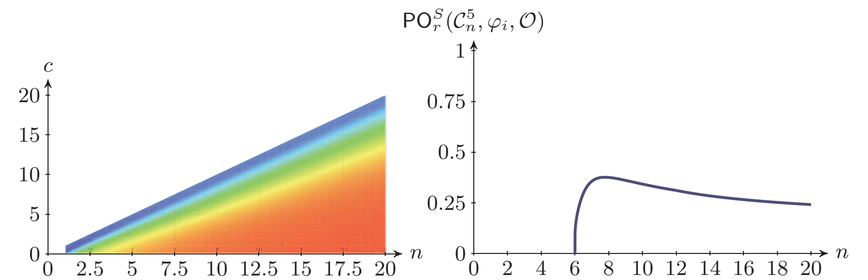
$$\mathsf{PO}_r^A(\mathcal{C}_n^0, \varphi_i, \mathcal{O}) = \frac{n-1}{n},$$

which is close to 1, but never exactly, since φ_i is not always false, although of decreasing proportion as the crowds grows. This result has to be put in parallel with the one from Reiter and Rubin (1998), which states that crowds is secure since each user is 'beyond suspicion' of being the initiator, but 'absolute privacy' is not achieved.

6.3.3. *Computation of RPSO.* From the probabilities computed above, we obtain that if $i \neq j$,

$$V(i \rightsquigarrow \mid \rightsquigarrow j) = \max\left(\frac{1}{n}, \frac{n-1}{n}\right) = \frac{1}{n}\max(1, n-1).$$

[†] This stems from the fact that the original models had either the server or the corrupt users as attackers, not both at the same time.



(a) Evolution of $\mathsf{PO}_r^A(\mathcal{C}_n^c,\varphi_i,\mathcal{O})$ with n and c. Red meaning a value close to 1 and blue meaning close to 0.

(b) Evolution of $\mathsf{PO}_r^S(\mathcal{C}_n^c,\varphi_i,\mathcal{O})$ with n when $c=5$.

Fig. 14. (Colour online) Evolution of restrictive opacity with the size of the crowd.

Except in the case of $n=1$ (when the system is non-opaque, hence $\mathsf{PO}_r^S(\mathcal{C}_1^0,\varphi_1,\mathcal{O})=0$), $V(i \rightsquigarrow \mid \rightsquigarrow j)=\frac{n-1}{n}$.

In the case when $i=j$

$$V(i \rightsquigarrow \mid \rightsquigarrow i)=\max\left(\frac{c+1}{n},\frac{n-c-1}{n}\right)=\frac{1}{n}\max(c+1,n-c-1).$$

That means the vulnerability for the observation class corresponding to the case when i is actually detected depends on the proportion of corrupted users in the crowd. Indeed, $V(i \rightsquigarrow \mid \rightsquigarrow i)=\frac{c+1}{n}$ if and only if $n\leqslant 2(c+1)$. The two cases shall be separated.

When $n\leqslant 2(c+1)$**.** The message is initially more likely to be sent to a corrupt user or to the initiator himself than to any other user in the crowd:

$$\begin{aligned}\sum_{j=1}^{n-c}\mathbf{P}(\rightsquigarrow j)\cdot\log(1-V(i\rightsquigarrow\mid\rightsquigarrow j)) &= \frac{(n-c-1)\cdot\log\left(\frac{1}{n}\right)+\log\left(\frac{n-c-1}{n}\right)}{n-c}\\ &=\frac{1}{n-c}\cdot(\log(n-c-1)-(n-c)\cdot\log(n))\\ &=\frac{\log(n-c-1)}{n-c}-\log(n)\end{aligned}$$

$$\text{Hence}\quad \mathsf{PO}_r^S(\mathcal{C}_n^c,\varphi_i,\mathcal{O})=\frac{1}{\log(n)-\frac{\log(n-c-1)}{n-c}}$$

When $n>2(c+1)$**.** The message is initially more likely to be sent to a honest user different from the initiator:

$$\begin{aligned}\sum_{j=1}^{n-c}\mathbf{P}(\rightsquigarrow j)\cdot\log(1-V(i\rightsquigarrow\mid\rightsquigarrow j)) &= \frac{(n-c-1)\cdot\log\left(\frac{1}{n}\right)+\log\left(\frac{c+1}{n}\right)}{n-c}\\ &=\frac{1}{n-c}\cdot(\log(c+1)-(n-c)\cdot\log(n))\\ &=\frac{\log(c+1)}{n-c}-\log(n)\end{aligned}$$

$$\text{Hence}\quad \mathsf{PO}_r^S(\mathcal{C}_n^c,\varphi_i,\mathcal{O})=\frac{1}{\log(n)-\frac{\log(c+1)}{n-c}}$$

The evolution of RPSO for $c=5$ is depicted in Figure 14b.

One can see that actually the RPSO decreases when n increases. That is because when there are more users in the crowd, user i is less likely to be the initiator. Hence, the predicate chosen does not model anonymity as specified in Reiter and Rubin (1998) but a stronger property since RPSO is based on the definition of symmetrical opacity. Therefore, it is meaningful in terms of security properties only when both the predicate and its negation are meaningful.

7. Dealing with non-determinism

The measures presented above were all defined in the case of fully probabilistic finite automata. However, some systems present non-determinism that cannot reasonably be abstracted away. For example, consider the case of a system, in which a malicious user Alice can control certain actions. The goal of Alice is to establish a covert communication channel with an external observer Bob. Hence she will try to influence the system in order to render communication easier. Therefore, the actual security of the system as observed by Bob should be measured against the best possible actions for Alice. Formally, Alice is a scheduler who, when facing several possible output distributions $\{\mu_1, \dots, \mu_n\}$, can choose whichever distribution ν on $\{1, \dots, n\}$ as weights for the μ_is. The security as measured by opacity is the minimal security of all possible successive choices.

7.1. *The non-deterministic framework*

Here we enlarge the setting of probabilistic automata considered before with non-determinism. There are several outgoing distribution from a given state instead of a single one.

Definition 7.1 (non-deterministic probabilistic automaton). A *non-deterministic probabilistic automaton* (NPA) is a tuple $\langle \Sigma, Q, \Delta, q_0 \rangle$ where

— Σ is a finite set of actions;
— Q is a finite set of states;
— $\Delta : Q \to \mathcal{P}(\mathcal{D}((\Sigma \times Q) \uplus \{\surd\}))$ is a non-deterministic probabilistic transition function;
— q_0 is the initial state;

where $\mathcal{P}(A)$ denotes the set of finite subsets of A.

The choice over the several possible distributions is made by the *scheduler*. It does not, however, selects one distribution to be used, but can give weight to the possible distributions.

Definition 7.2 (scheduler). A scheduler on $\mathcal{A} = \langle \Sigma, Q, \Delta, q_0 \rangle$ is a function

$$\sigma : Run(\mathcal{A}) \to \mathcal{D}(\mathcal{D}((\Sigma \times Q) \uplus \{\surd\}))$$

such that $\sigma(\rho)(\nu) > 0 \Rightarrow \nu \in \Delta(\mathrm{lst}(\rho))$.

The set of all schedulers for $\mathcal{A}$ is denoted $Sched_{\mathcal{A}}$ (the dependence on $\mathcal{A}$ will be omitted if clear from the context).

Observe that the choice made by a scheduler can depend on the (arbitrarily long) history of the execution. A scheduler σ is *memoryless* if there exists a function $\sigma' : Q \to \mathcal{D}(\mathcal{D}((\Sigma \times Q) \uplus \{\surd\}))$ such that $\sigma(\rho) = \sigma'(\mathrm{lst}(\rho))$. Hence a memoryless scheduler takes only into account the current state.

Definition 7.3 (scheduled NPA). NPA $\mathcal{A} = \langle \Sigma, Q, \Delta, q_0 \rangle$ scheduled by σ is the (infinite) FPFA $\mathcal{A}_{/\sigma} = \langle \Sigma, Run(\mathcal{A}), \Delta', \varepsilon \rangle$ where

$$\Delta'(\rho)(a, \rho') = \sum_{\mu \in \Delta(q)} \sigma(\rho)(\mu) \cdot \mu(a, q') \quad \text{if } \rho' = \overbrace{q_0 \to \cdots \to q}^{\rho} \xrightarrow{a} q'$$

and $\Delta'(\rho)(a, \rho') = 0$ otherwise.

A scheduled NPA behaves as an FPFA, where the outgoing distribution is the set of all possible distributions weighted by the scheduler.

All measures defined in this paper on fully probabilistic finite automata can be extended to non-deterministic probabilistic automata. First note that all measures can be defined on infinite systems, although they cannot in general be computed automatically, even with proper restrictions on predicate and observables. From the security point of view, opacity in the case of an NPA should be the measure for the FPFA obtained with the worst possible scheduler. Hence the leak evaluated by the liberal measures (LPO and LPSO) is the greatest possible, and the robustness evaluated by the restrictive measures (RPO and RPSO) is the weakest possible.

Definition 7.4. Let $\mathcal{A}$ be an NPA, φ a predicate, and $\mathcal{O}$ an observation function.

$$\text{For } \mathsf{PO} \in \{\mathsf{PO}_\ell^A, \mathsf{PO}_\ell^S\}, \quad \widehat{\mathsf{PO}}(\mathcal{A}, \varphi, \mathcal{O}) = \max_{\sigma \in Sched} \mathsf{PO}(\mathcal{A}_{/\sigma}, \varphi, \mathcal{O}).$$

$$\text{For } \mathsf{PO} \in \{\mathsf{PO}_r^A, \mathsf{PO}_r^S\}, \quad \widehat{\mathsf{PO}}(\mathcal{A}, \varphi, \mathcal{O}) = \min_{\sigma \in Sched} \mathsf{PO}(\mathcal{A}_{/\sigma}, \varphi, \mathcal{O}).$$

7.2. *The expressive power of schedulers*

In the context of analysis of security systems running in a hostile environment, it is quite natural to consider the scheduler to be under control of the adversary. However, if not constrained this gives the adversary an unreasonably strong power even for obviously secure systems as it can reveal certain secrets. Also several classes of schedulers have been proposed in order to avoid considering unrealistic power of unconstrained schedulers and the ability of these classes to reach supremum probabilities (Giro and D'Argenio 2009). We now investigate this problem for quantitative opacity.

First we show that memoryless schedulers are not sufficiently expressive, with the following counterexample.

Theorem 7.5. There exists an NPA $\mathcal{B}$ such that the value $\widehat{\mathsf{PO}_r^A}(\mathcal{B}, \varphi, \mathcal{O})$ cannot be reached by a memoryless scheduler.

Proof. Consider the NPA $\mathcal{B}$ of Figure 15. Transitions on a and b going to state q_1 (along with the westbound $\surd$) are part of the same probabilistic transition indicated

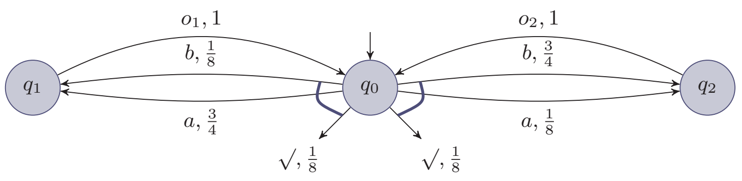
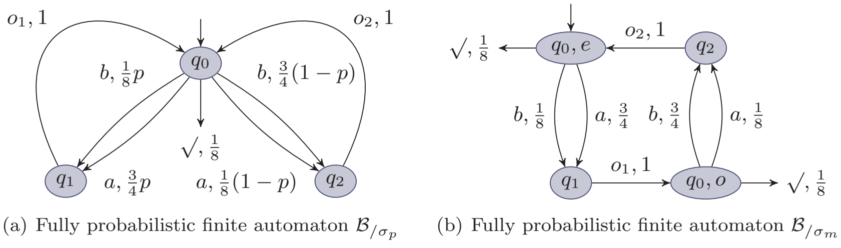


Fig. 15. (Colour online) A non-deterministic probabilistic automaton $\mathcal{B}$.



(a) Fully probabilistic finite automaton $\mathcal{B}_{/\sigma_p}$ (b) Fully probabilistic finite automaton $\mathcal{B}_{/\sigma_m}$

Fig. 16. (Colour online) Scheduled automata.

by the arc linking the outgoing edges (and similarly eastbound). Let φ be the (regular) predicate consisting of runs whose trace projected onto $\{a, b\}$ is in $(ab)^+ + (ab)^*a$ (so a and b must alternate). Let $\mathcal{O}$ be the observation function that keeps the last o_i of the run. Hence there are only three observables, ε, o_1 and o_2. Intuitively, a scheduler can introduce a bias in the next letter read from state q_0.

First consider a memoryless scheduler σ_p. It can only choose once what weight will be affected to each transition. This choice is parametrized by probability p that represents the weight of probability of the q_1 transition. The scheduled NPA $\mathcal{B}_{/\sigma_p}$ is depicted on Figure 16a. The probabilities can be computed using the technique laid out in Section 5. We obtain the following probabilities (see Appendix C.1 for details):

$$\mathbf{P}(\varepsilon) = \frac{1}{8} \qquad \mathbf{P}(o_1) = \frac{7}{8} \cdot p \qquad \mathbf{P}(o_2) = \frac{7}{8} \cdot (1-p) \qquad \mathbf{P}(\varphi, \varepsilon) = 0$$

$$\mathbf{P}(\varphi, o_1) = \frac{p}{25p^2 - 25p + 58} \cdot \frac{5p + 49}{8} \qquad \mathbf{P}(\varphi, o_2) = \frac{1-p}{25p^2 - 25p + 58} \cdot \frac{15p + 7}{4}$$

Which yields

$$\frac{1}{\mathsf{PO}_r^A(\mathcal{B}_{/\sigma_p}, \varphi, \mathcal{O})} = \frac{1}{8} + \frac{49}{8} \cdot f(p) \cdot \left(\frac{p}{7f(p) - 5p - 49} + \frac{1-p}{7f(p) - 30p - 14} \right)$$

with $f(p) = 25p^2 - 25p + 58$ (see Appendix C.2). It can be shown[†] that regardless of p, $\mathsf{PO}_r^A(\mathcal{B}_{/\sigma_p}, \varphi, \mathcal{O})$ never falls below 0.88.

Now consider a scheduler σ_m with memory who will try to maximize the realization of φ. In order to achieve that, it introduces a bias towards taking the letter which will fulfill φ: first an a, then a b, etc. Hence on the even positions, it will choose only transition to q_1 (with probability 1) while it will choose the transition to q_2 on odd positions. The

[†] With the help of tools such as WolframAlpha.

resulting FPFA is depicted on Figure 16b. In this case, the probabilities of interest are:

$$\mathbf{P}(\varepsilon) = \frac{1}{8} \qquad \mathbf{P}(o_1) = \frac{7}{15} \qquad \mathbf{P}(o_2) = \frac{7}{8} \cdot \frac{7}{15}$$

$$\mathbf{P}(\varphi, \varepsilon) = 0 \qquad \mathbf{P}(\varphi, o_1) = \frac{3}{14} \qquad \mathbf{P}(\varphi, o_2) = \frac{3}{4} \cdot \frac{3}{14}$$

Probability $\mathbf{P}(o_1)$ can be obtained by noticing that the execution has to stop after an odd number of letters from $\{a, b\}$ have been read. The probability of stopping after exactly n letters from a or b is $\frac{1}{8} \cdot \left(\frac{7}{8}\right)^n$. Therefore

$$\mathbf{P}(o_1) = \frac{1}{8} \cdot \sum_{i \geqslant 0} \left(\frac{7}{8}\right)^{2i+1} = \frac{1}{8} \cdot \frac{7}{8} \cdot \frac{1}{1 - \frac{49}{64}} = \frac{1}{8} \cdot \frac{7}{8} \cdot \frac{64}{15} = \frac{7}{15}.$$

Similar reasoning yield the other probabilities. The computation of RPO from these values (see Appendix C.3) gives $\mathsf{PO}_r^A(\mathcal{B}_{/\sigma_m}, \varphi, \mathcal{O}) = \frac{88{,}192}{146{,}509} \simeq 0.60$.

Hence a lower security is achieved by a scheduler provided it has (a finite amount of) memory. □

Note that this example used RPO, but a similar argument could be adapted for the other measures.

7.3. *Restricted schedulers*

What made a scheduler with memory more powerful than the one without in the counterexample of Section 7.2 was the knowledge of the truth value of φ and exactly what was observed. More precisely, if the predicate and the observables are regular languages represented by finite deterministic and complete automata (FDCA), schedulers can be restricted to choices according to the current state of these automata and the state of the system. We conjecture that this knowledge is sufficient to any scheduler to compromise security at the best of its capabilities.

Let $\varphi \subseteq CRun(\mathcal{A})$ be a regular predicate represented by an FDCA $\mathcal{A}_\varphi$. Let $\mathcal{O} : CRun(\mathcal{A}) \to \{o_1, \ldots, o_n\}$ be an observation function such that for $1 \leqslant i \leqslant n$, $\mathcal{O}^{-1}(o_i)$ is a regular set represented by an FDCA $\mathcal{A}_{o_i}$. Consider the synchronized product $\mathcal{A}_{\varphi,\mathcal{O}} = \mathcal{A}_\varphi || \mathcal{A}_{o_1} || \ldots || \mathcal{A}_{o_n}$, which is also an FDCA, and denote by $Q_{\varphi,\mathcal{O}}$ its set of states. Let $\mathcal{A}_{\varphi,\mathcal{O}}(\rho)$ be the state of $\mathcal{A}_{\varphi,\mathcal{O}}$ reached after reading ρ.

Definition 7.6 (restricted $(\varphi, \mathcal{O})$-scheduler). A scheduler σ for $\mathcal{A}$ is said $(\varphi, \mathcal{O})$-restricted if there exists a function $\sigma' : (Q_{\varphi,\mathcal{O}} \times Q) \to \mathcal{D}(\mathcal{D}((\Sigma \times Q) \uplus \{\surd\}))$ such that for any run $\rho \in Run(\mathcal{A})$, $\sigma(\rho) = \sigma'(\mathcal{A}_{\varphi,\mathcal{O}}(\rho), \mathrm{lst}(\rho))$.

Remark that memoryless schedulers are always $(\varphi, \mathcal{O})$-restricted.

Proposition 7.7. If σ is $(\varphi, \mathcal{O})$-restricted, then $\mathcal{A}_{/\sigma}$ is isomorphic to a finite FPFA.

These schedulers keep all information about the predicate and the observation. We conjecture that the relevant supremum is reached by a $(\varphi, \mathcal{O})$-restricted scheduler.

Sketch of proof of Proposition 7.7 It can be shown that if σ is $(\varphi, \mathcal{O})$-restricted, then:

1. σ is a memoryless scheduler for the product $\mathcal{A}||\mathcal{A}_{\varphi,\mathcal{O}}$
2. and $\left(\mathcal{A}||\mathcal{A}_{\varphi,\mathcal{O}}\right)_{/\sigma} = \mathcal{A}_{/\sigma}$.

□

8. Conclusion

In this paper we introduced two dual notions of probabilistic opacity. The liberal one measures the probability for an attacker observing a random execution of the system to be able to gain information he can be sure about. The restrictive one measures the level of certitude in the information acquired by an attacker observing the system. The extremal cases of both these notions coincide with the possibilistic notion of opacity, which evaluates the existence of a leak of sure information. These notions yield measures that generalize either the case of asymmetrical or symmetrical opacity, thus providing four measures.

However, probabilistic opacity is not always easy to compute, especially if there are an infinite number of observables. Nevertheless, automatic computation is possible when dealing with regular predicates and finitely many regular observation classes. A prototype tool was implemented in Java, and can be used for numerical computation of opacity values.

In future work we plan to explore more of the properties of probabilistic opacity, to instantiate it to known security measures (anonymity, non-interference, etc). Also, we want to extend the study of the non-deterministic case, by investigating the expressiveness of schedulers.

Acknowledgments

We wish to thank Catuscia Palamidessi for interesting discussions on this work. Thanks also to the reviewers for their helpful comments and to Adrian Eftenie for implementing the TPOT tool.

The authors are supported by projects ImpRo (ANR-2010-BLAN-0317) (French Government), CoChaT (DIGITEO-2009-27HD) (Région Île de France), inVEST (ERC Starting Grant 279499), the NSERC Discovery Individual grant No. 13321 (Government of Canada), and by the FQRNT Team grant No. 167440 (Quebec's Government).

Appendix A. Computation of RPO for the Debit Card System

We give here the details of the computation of RPO in the example of the debit cards system of Section 4.1.2.

$$\begin{aligned}\mathbf{P}(\mathcal{O}_{\mathrm{Call}} = \varepsilon) \;=\; & \mathbf{P}(\mathcal{O}_{\mathrm{Call}} = \varepsilon, x > 1000) + \mathbf{P}(\mathcal{O}_{\mathrm{Call}} = \varepsilon, 500 < x \leqslant 1000) \\ & + \mathbf{P}(\mathcal{O}_{\mathrm{Call}} = \varepsilon, 100 < x \leqslant 500) + \mathbf{P}(\mathcal{O}_{\mathrm{Call}} = \varepsilon, x \leqslant 100)\end{aligned}$$

$$\begin{aligned}
&= \mathbf{P}(\mathcal{O}_{\text{Call}} = \varepsilon) \cdot \mathbf{P}(x > 1000) \\
&\quad + \mathbf{P}(\mathcal{O}_{\text{Call}} = \varepsilon) \cdot \mathbf{P}(500 < x \leqslant 1000) \\
&\quad + \mathbf{P}(\mathcal{O}_{\text{Call}} = \varepsilon) \cdot \mathbf{P}(100 < x \leqslant 500) \\
&\quad + \mathbf{P}(\mathcal{O}_{\text{Call}} = \varepsilon) \cdot \mathbf{P}(x \leqslant 100) \\
&= 0.05 \cdot 0.05 + 0.25 \cdot 0.2 + 0.5 \cdot 0.45 + 0.8 \cdot 0.3 \\
\mathbf{P}(\mathcal{O}_{\text{Call}} = \varepsilon) &= 0.5175
\end{aligned}$$

$$\mathbf{P}(\mathcal{O}_{\text{Call}} = \text{Call}) = 1 - \mathbf{P}(\mathcal{O}_{\text{Call}} = \varepsilon) = 0.4825$$

$$\begin{aligned}
\mathbf{P}(\neg\varphi_{>500} | \mathcal{O}_{\text{Call}} = \varepsilon) &= \frac{\mathbf{P}(\neg\varphi_{>500}, \mathcal{O}_{\text{Call}} = \varepsilon)}{\mathbf{P}(\mathcal{O}_{\text{Call}} = \varepsilon)} \\
&= \frac{\mathbf{P}(x \leqslant 100, \mathcal{O}_{\text{Call}} = \varepsilon) \ + \ \mathbf{P}(100 < x \leqslant 500, \mathcal{O}_{\text{Call}} = \varepsilon)}{\mathbf{P}(\mathcal{O}_{\text{Call}} = \varepsilon)} \\
&= \frac{\mathbf{P}(\mathcal{O}_{\text{Call}} = \varepsilon | x \leqslant 100) \cdot \mathbf{P}(x \leqslant 100)}{\mathbf{P}(\mathcal{O}_{\text{Call}} = \varepsilon)} \\
&\quad + \frac{\mathbf{P}(\mathcal{O}_{\text{Call}} = \varepsilon | 100 < x \leqslant 500) \cdot \mathbf{P}(100 < x \leqslant 500)}{\mathbf{P}(\mathcal{O}_{\text{Call}} = \varepsilon)} \\
&= \frac{0.8 \cdot 0.3 + 0.5 \cdot 0.45}{0.5175} \\
\mathbf{P}(\neg\varphi_{>500} | \mathcal{O}_{\text{Call}} = \varepsilon) &= \frac{0.465}{0.5175} \ \simeq \ 0.899
\end{aligned}$$

$$\begin{aligned}
\mathbf{P}(\neg\varphi_{>500} | \mathcal{O}_{\text{Call}} = \text{Call}) &= \frac{\mathbf{P}(\neg\varphi_{>500}, \mathcal{O}_{\text{Call}} = \text{Call})}{\mathbf{P}(\mathcal{O}_{\text{Call}} = \text{Call})} \\
&= \frac{\mathbf{P}(x \leqslant 100, \mathcal{O}_{\text{Call}} = \text{Call}) \ + \ \mathbf{P}(100 < x \leqslant 500, \mathcal{O}_{\text{Call}} = \text{Call})}{\mathbf{P}(\mathcal{O}_{\text{Call}} = \text{Call})} \\
&= \frac{\mathbf{P}(\mathcal{O}_{\text{Call}} = \text{Call} | x \leqslant 100) \cdot \mathbf{P}(x \leqslant 100)}{\mathbf{P}(\mathcal{O}_{\text{Call}} = \text{Call})} \\
&\quad + \frac{\mathbf{P}(\mathcal{O}_{\text{Call}} = \text{Call} | 100 < x \leqslant 500) \cdot \mathbf{P}(100 < x \leqslant 500)}{\mathbf{P}(\mathcal{O}_{\text{Call}} = \text{Call})} \\
&= \frac{0.2 \cdot 0.3 + 0.5 \cdot 0.45}{0.4825} \\
\mathbf{P}(\neg\varphi_{>500} | \mathcal{O}_{\text{Call}} = \text{Call}) &= \frac{0.285}{0.4825} \ \simeq \ 0.591
\end{aligned}$$

$$\begin{aligned}
\frac{1}{\mathsf{PO}_r^A(\mathcal{A}_{\text{card}}, \varphi_{>500}, \mathcal{O}_{\text{Call}})} &= 0.5175 \cdot \frac{0.5175}{0.465} + 0.4825 \cdot \frac{0.4825}{0.285} \\
\frac{1}{\mathsf{PO}_r^A(\mathcal{A}_{\text{card}}, \varphi_{>500}, \mathcal{O}_{\text{Call}})} &= \frac{39377}{28272} \ \simeq \ 1.393
\end{aligned}$$

The last line was obtained by reducing the one above with the help of the formal computation tool *WolframAlpha*.

Appendix B. Resolution of the Linear System for Crowds Protocol

It can be seen in the system of Table 6 (page 387) that $L_1 = L_2 = \cdots = L_{n-c-1} = q \cdot L_{1'}$ and $L_{n-c+1} = \cdots = L_n = L_S = 1$. Therefore, it suffices to eliminate $L_{1'}$ and compute L_0, L_1 and L_{n-c}.

$$\begin{cases} L_0 &= \dfrac{1}{q(n-c)} \cdot L_1 \\ L_1 &= q\left(\dfrac{n-c-1}{n} \cdot L_1 + \dfrac{1}{n} \cdot L_{n-c}\right) \\ L_{n-c} &= 1 - \dfrac{q(n-c)}{n} + L_1 \end{cases}$$

The line for L_{n-c} is obtained as follows:

$$\begin{aligned} L_{n-c} &= (1-q) \cdot L_S + \sum_{i=1}^{n} \frac{q}{n} \cdot L_i \\ L_{n-c} &= (1-q) \cdot L_S + \sum_{i=1}^{n-c} \frac{q}{n} \cdot L_i + \sum_{i=n-c+1}^{n} \frac{q}{n} \cdot L_i \\ L_{n-c} &= 1 - q + L_1 + \frac{q \cdot c}{n} \\ L_{n-c} &= 1 - \frac{q(n-c)}{n} + L_1. \end{aligned}$$

This yields, for L_1:

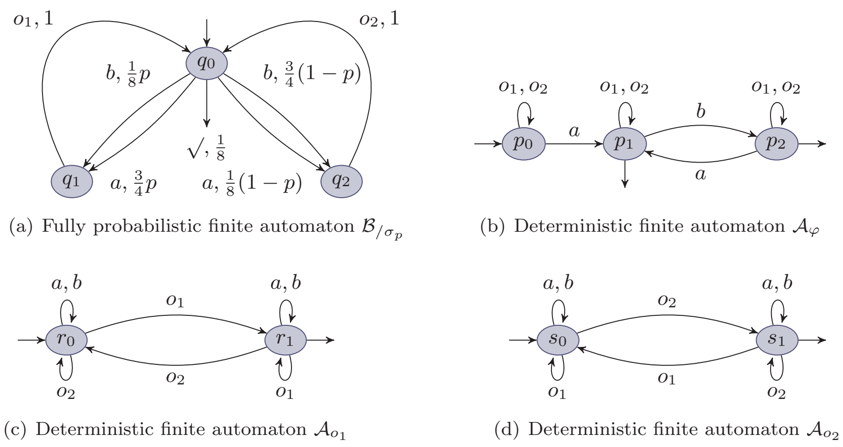
$$\begin{aligned} L_1 &= \frac{q}{n}(n-c) \cdot L_1 + \frac{q}{n}\left(1 - \frac{q(n-c)}{n}\right) \\ \frac{n}{q} \cdot L_1 &= (n-c) \cdot L_1 + 1 - \frac{q(n-c)}{n} \\ L_1\left(\frac{n}{q} - (n-c)\right) &= \frac{n - q(n-c)}{n} \\ L_1 &= \frac{q}{n}. \end{aligned}$$

The other values are easily deduced from L_1.

Appendix C. Calculations in the Proof of Theorem 7.5

C.1. *Probabilities in $\mathcal{B}_{/\sigma_p}$*

We compute the probabilities of several events in the automaton $\mathcal{B}_{/\sigma_p}$, reproduced on Figure C1a. Recall that $\varphi = (ab)^+ + (ab)^*a$ and $\mathcal{O}$ is the last letter read, so the set of observables is $Obs = \{\varepsilon, o_1, o_2\}$.



(a) Fully probabilistic finite automaton $\mathcal{B}_{/\sigma_p}$

(b) Deterministic finite automaton $\mathcal{A}_\varphi$

(c) Deterministic finite automaton $\mathcal{A}_{o_1}$

(d) Deterministic finite automaton $\mathcal{A}_{o_2}$

Fig. C1. (Colour online) Automata for the computation of $\mathbf{P}(\varphi, o_1)$ and $\mathbf{P}(\varphi, o_2)$.

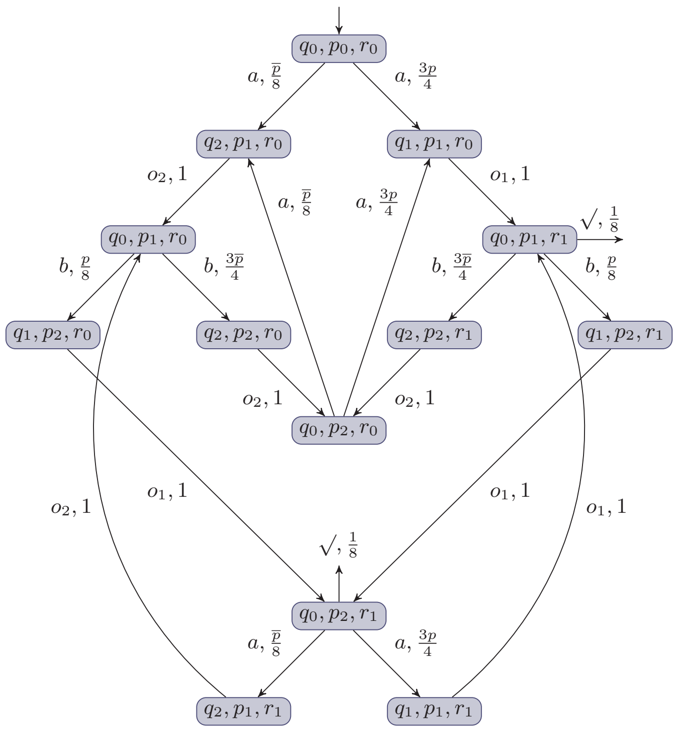


Fig. C2. (Colour online) Substochastic Automaton $\mathcal{B}_{/\sigma_p}||\mathcal{A}_\varphi||\mathcal{A}_{o_1}$.

The computation of the probability $\mathbf{P}(\varphi, o_1)$ in FPFA $\mathcal{B}_{/\sigma_p}$ goes as follows. We write $\overline{p} = 1-p$ for brevity. First we build the synchronized product $\mathcal{B}_{/\sigma_p}||\mathcal{A}_\varphi||\mathcal{A}_{o_1}$, as depicted in Figure C2. The linear system of Table C1 is built from this automaton. This system can be trimmed down in order to remove redundancy, and since only the value of $x_{000} = \mathbf{P}(\varphi, o_1)$

Table C1. *Linear system associated to the SA $\mathcal{B}_{/\sigma_p}||\mathcal{A}_\varphi||\mathcal{A}_{o_1}$. The variables names indicate the corresponding state in the automaton; for example x_{210} corresponds to state (q_2, p_1, r_0).*

$$\begin{cases} x_{000} &= \frac{1}{8}\overline{p}\ x_{210} + \frac{3}{4}p\ x_{110} \\ x_{210} &= x_{010} \\ x_{110} &= x_{011} \\ x_{010} &= \frac{1}{8}p\ x_{120} + \frac{3}{4}\overline{p}\ x_{220} \\ x_{011} &= \frac{1}{8} + \frac{3}{4}\overline{p}\ x_{221} + \frac{1}{8}p\ x_{121} \\ x_{120} &= x_{021} \\ x_{220} &= x_{020} \\ x_{221} &= x_{020} \\ x_{121} &= x_{021} \\ x_{020} &= \frac{1}{8}\overline{p}\ x_{210} + \frac{3}{4}p\ x_{110} \\ x_{021} &= \frac{1}{8} + \frac{1}{8}\overline{p}\ x_{211} + \frac{3}{4}p\ x_{111} \\ x_{211} &= x_{010} \\ x_{111} &= x_{011} \end{cases}$$

is of interest:

$$\begin{cases} x_{000} &= \frac{1}{8}\overline{p}\ x_{010} + \frac{3}{4}p\ x_{011} \\ x_{010} &= \frac{1}{8}p\ x_{021} + \frac{3}{4}\overline{p}\ x_{020} \\ x_{011} &= \frac{1}{8} + \frac{3}{4}\overline{p}\ x_{020} + \frac{1}{8}p\ x_{021} \\ x_{020} &= \frac{1}{8}\overline{p}\ x_{010} + \frac{3}{4}p\ x_{011} \\ x_{021} &= \frac{1}{8} + \frac{1}{8}\overline{p}\ x_{010} + \frac{3}{4}p\ x_{011} \end{cases} \iff \begin{cases} x_{000} &= \frac{1}{8}\overline{p}\ x_{010} + \frac{3}{4}p\ x_{011} \\ x_{010} &= \frac{1}{8}p\ x_{021} + \frac{3}{4}\overline{p}\ x_{020} \\ x_{011} &= \frac{1}{8} + x_{010} \\ x_{020} &= x_{000} \\ x_{021} &= \frac{1}{8} + x_{000} \end{cases}$$

We therefore obtain:

$$\begin{cases} x_{000} &= \frac{1}{8}\overline{p}\ x_{010} + \frac{3}{4}p\left(\frac{1}{8} + x_{010}\right) \\ x_{010} &= \frac{1}{8}p\left(\frac{1}{8} + x_{000}\right) + \frac{3}{4}\overline{p}\ x_{000} \end{cases} \text{So} \begin{cases} x_{000} &= \frac{1}{8}\overline{p}\ x_{010} + \frac{3}{4}p\left(\frac{1}{8} + x_{010}\right) \\ x_{010} &= \frac{1}{64}p + \frac{1}{8}p\ x_{000} + \frac{3}{4}\overline{p}\ x_{000} \end{cases}$$

As a result

$$x_{000} = \frac{1}{8}\overline{p}\left(\frac{1}{64}p + \frac{1}{8}p\ x_{000} + \frac{3}{4}\overline{p}\ x_{000}\right) + \frac{3}{4}p\left(\frac{1}{8} + \frac{1}{64}p + \frac{1}{8}p\ x_{000} + \frac{3}{4}\overline{p}\ x_{000}\right).$$

In the sequel, we replace x_{000} with x for readability's sake.

$$\begin{aligned} x &= \frac{1}{8}\overline{p}\left(\frac{1}{64}p + \frac{1}{8}p\ x + \frac{3}{4}\overline{p}\ x\right) + \frac{3}{4}p\left(\frac{1}{8} + \frac{1}{64}p + \frac{1}{8}p\ x + \frac{3}{4}\overline{p}\ x\right) \\ \iff x &= \frac{1}{512}p\overline{p} + \frac{1}{64}p\overline{p}x + \frac{3}{32}\overline{p}^2x + \frac{3}{32}p + \frac{3}{256}p^2 + \frac{3}{32}p^2x + \frac{9}{16}p\overline{p}x \\ \iff x\left(1 - \frac{1}{64}p\overline{p} - \frac{3}{32}\overline{p}^2 - \frac{3}{32}p^2 - \frac{9}{16}p\overline{p}\right) &= \frac{1}{512}p\overline{p} + \frac{3}{32}p + \frac{3}{256}p^2 \\ \iff x &= \frac{\frac{1}{8}p\overline{p} + 6p + \frac{3}{4}p^2}{64 - p\overline{p} - 6\overline{p}^2 - 6p^2 - 36p\overline{p}} \end{aligned}$$

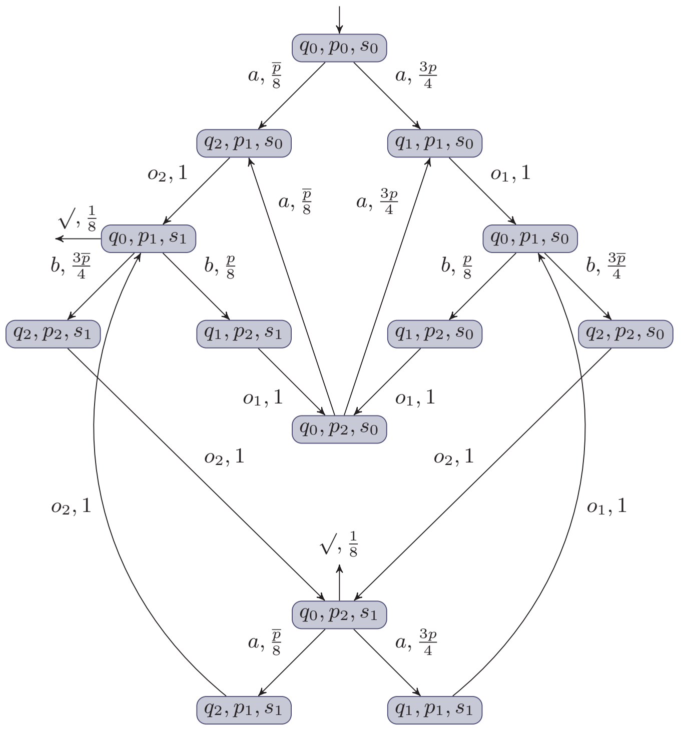


Fig. C3. (Colour online) Substochastic Automaton $\mathcal{B}_{/\sigma_p}||\mathcal{A}_\varphi||\mathcal{A}_{o_2}$.

$$\iff x = \frac{\frac{1}{8}p(1-p) + 6p + \frac{3}{4}p^2}{64 - 37p(1-p) - 6(p^2 - 2p + 1) - 6p^2}$$

$$\iff x = \frac{\frac{1}{8}p - \frac{1}{8}p^2 + 6p + \frac{3}{4}p^2}{64 - 37p + 37p^2 - 6p^2 + 12p - 6 - 6p^2}$$

$$\iff x = \frac{1}{8} \cdot p \cdot \frac{5p + 49}{25p^2 - 25p + 58}.$$

The same technique can be applied to the computation of $\mathbf{P}(\varphi, o_2)$ in $\mathcal{B}_{/\sigma_p}$. The product is depicted in Figure C3, and the linear system obtained boils down to

$$\begin{cases} x_{000} &= \frac{1}{8}\overline{p}\left(\frac{1}{8} + x_{010}\right) + \frac{3}{4}p\ x_{010} \\ x_{010} &= \frac{1}{8}p\ x_{000} + \frac{3}{4}\overline{p}\left(\frac{1}{8} + x_{000}\right). \end{cases}$$

As before, x_{000} is replaced by x for readability; we solve:

$$x = \frac{1}{8}\overline{p}\left(\frac{1}{8} + \frac{1}{8}px + \frac{3}{4}\overline{p}\left(\frac{1}{8} + x\right)\right) + \frac{3}{4}p\ \left(\frac{1}{8}px + \frac{3}{4}\overline{p}\left(\frac{1}{8} + x\right)\right)$$

$$x = \frac{1}{64}\overline{p} + \frac{1}{64}p\overline{p}x + \frac{3}{256}\overline{p}^2 + \frac{3}{32}\overline{p}^2x + \frac{3}{32}p^2x + \frac{9}{128}p\overline{p} + \frac{9}{16}p\overline{p}x$$

$$
\begin{aligned}
x &= \frac{\frac{1}{64}\overline{p} + \frac{3}{256}\overline{p}^2 + \frac{9}{128}p\overline{p}}{1 - \frac{1}{64}p\overline{p} - \frac{3}{32}\overline{p}^2 - \frac{3}{32}p^2 - \frac{9}{16}p\overline{p}} \\
x &= \frac{4\overline{p} + 3\overline{p}^2 + 18p\overline{p}}{256 - 24\overline{p}^2 - 24p^2 - 148p\overline{p}} \\
x &= \frac{(1-p)(4+3-3p+18p)}{256 - 24p^2 + 48p - 24 - 24p^2 - 148p + 148p^2} \\
x &= \frac{(1-p)(15p+7)}{232 - 100p + 100p^2} \\
x &= \frac{(1-p)(15p+7)}{4(25p^2 - 25p + 58)}
\end{aligned}
$$

C.2. *Computation of RPO for $\mathcal{B}_{/\sigma_p}$*

In the sequel, we write $f(p) = 25p^2 - 25p + 58$. We have $\mathbf{P}(\overline{\varphi}|\varepsilon) = 1$ and

$$
\begin{aligned}
\mathbf{P}(\overline{\varphi}|o_1) &= 1 - \frac{\mathbf{P}(\varphi, o_1)}{\mathbf{P}(o_1)} \\
\mathbf{P}(\overline{\varphi}|o_1) &= 1 - \frac{1}{8} \cdot p \cdot \frac{5p+49}{25p^2 - 25p + 58} \cdot \frac{8}{7p} \\
\mathbf{P}(\overline{\varphi}|o_1) &= \frac{7f(p) - 5p - 49}{7f(p)}
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{P}(\overline{\varphi}|o_2) &= 1 - \frac{\mathbf{P}(\varphi, o_2)}{\mathbf{P}(o_2)} \\
\mathbf{P}(\overline{\varphi}|o_2) &= 1 - \frac{(1-p)(15p+7)}{4(25p^2 - 25p + 58)} \cdot \frac{8}{7(1-p)} \\
\mathbf{P}(\overline{\varphi}|o_2) &= \frac{7f(p) - 30p - 14}{7f(p)}
\end{aligned}
$$

$$
\begin{aligned}
\frac{1}{\mathsf{PO}_r^A\left(\mathcal{B}_{/\sigma_p}, \varphi, \mathcal{O}\right)} &= \mathbf{P}(\varepsilon) + \mathbf{P}(o_1) \cdot \frac{1}{\mathbf{P}(\overline{\varphi}|o_1)} + \mathbf{P}(o_2) \cdot \frac{1}{\mathbf{P}(\overline{\varphi}|o_2)} \\
\frac{1}{\mathsf{PO}_r^A\left(\mathcal{B}_{/\sigma_p}, \varphi, \mathcal{O}\right)} &= \frac{1}{8} + \frac{7}{8} \cdot p \cdot \frac{7f(p)}{7f(p) - 5p - 49} + \frac{7}{8} \cdot (1-p) \cdot \frac{7f(p)}{7f(p) - 30p - 14} \\
\frac{1}{\mathsf{PO}_r^A\left(\mathcal{B}_{/\sigma_p}, \varphi, \mathcal{O}\right)} &= \frac{1}{8} + \frac{49f(p)}{8}\left(\frac{p}{7f(p) - 5p - 49} + \frac{1-p}{7f(p) - 30p - 14}\right)
\end{aligned}
$$

C.3. *Computation of RPO for $\mathcal{B}_{/\sigma_m}$*

We have:

$$
\mathbf{P}(\overline{\varphi}|\varepsilon) = 1 \qquad \mathbf{P}(\overline{\varphi}|o_1) = 1 - \frac{3}{14} \cdot \frac{15}{7} = \frac{53}{98} \qquad \mathbf{P}(\overline{\varphi}|o_2) = 1 - \frac{3}{4} \cdot \frac{3}{14} \cdot \frac{15}{7} \cdot \frac{8}{7} = \frac{208}{343}
$$

Therefore:

$$\frac{1}{\mathsf{PO}_r^A(\mathcal{B}_{/\sigma_m}, \varphi, \mathcal{O})} = \frac{1}{8} + \frac{7}{15} \cdot \frac{98}{53} + \frac{7}{8} \cdot \frac{7}{15} \cdot \frac{343}{208}$$
$$\frac{1}{\mathsf{PO}_r^A(\mathcal{B}_{/\sigma_m}, \varphi, \mathcal{O})} = \frac{146,509}{88,192}$$
$$\mathsf{PO}_r^A(\mathcal{B}_{/\sigma_m}, \varphi, \mathcal{O}) = \frac{88,192}{146,509}$$
$$\mathsf{PO}_r^A(\mathcal{B}_{/\sigma_m}, \varphi, \mathcal{O}) \simeq 0.60$$

References

Aldini, A. and Bernardo, M. (2009) A general framework for nondeterministic, probabilistic, and stochastic noninterference. In: Degano, P. and Viganò, L. (eds.) Proceedings of Foundations and Applications of Security Analysis; ARSPA-WITS 2009. *Springer Lecture Notes in Computer Science* **5511** 191–245.

Aldini, A., Bravetti, M. and Gorrieri, R. (2004) A process-algebraic approach for the analysis of probabilistic noninterference. *Journal of Computer Security* **12**(2) 191–245.

Alur, R., Černý, P. and Zdancewic, S. (2006) Preserving secrecy under refinement. In: Proceedings of the 33rd International Colloquium on Automata, Languages and Programming (ICALP'06). *Springer Lecture Notes in Computer Science* **4052** 107–118.

Alvim, M. S., Andrés, M. E. and Palamidessi, C. (2010) Information flow in interactive systems. In: Gastin, P. and Laroussinie, F. (eds.) Proceedings of the 21th International Conference on Concurrency Theory (CONCUR'10). *Springer Lecture Notes in Computer Science* **6269** 102–116.

Andrés, M. E., Palamidessi, C., van Rossum, P. and Smith, G. (2010) Computing the leakage of information-hiding systems. In: Proceedings 16th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'10) *Springer Lecture Notes in Computer Science.* **6015** 373–389.

Bérard, B., Mullins, J., and Sassolas, M. (2010) Quantifying opacity. In: Ciardo, G. and Segala, R. (eds.) *Proceedings of the 7th International Conference on Quantitative Evaluation of Systems (QEST'10)*, IEEE Computer Society 263–272.

Bianco, A. and de Alfaro, L. (1995) Model checking of probabilistic and nondeterministic systems. In: Proceedings 15th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'95). *Springer Lecture Notes in Computer Science* **1026** 499–513.

Boreale, M. (2009) Quantifying information leakage in process calculi. *Information and Computation* **207** (6) 699–725.

Boreale, M., Clark, D. and Gorla, D. (2010) A semiring-based trace semantics for processes with applications to information leakage analysis. In: Calude, C. S. and Sassone, V. (eds.) Proceedings of 6th IFIP TC 1/WG 2.2 International Conference, TCS 2010, Held as Part of WCC 2010 (IFIP TCS). *Springer International Federation for Information Processing* **323** 340–354.

Boreale, M., Pampaloni, F. and Paolini, M. (2011a) Asymptotic information leakage under one-try attacks. In: Hofmann, M. (ed.) Foundations of Software Science and Computational Structures - 14th International Conference, FOSSACS 2011, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2011. *Springer Lecture Notes in Computer Science* **6604** 396–410.

Boreale, M., Pampaloni, F. and Paolini, M. (2011b) Quantitative information flow, with a view. In: Atluri, V. and Díaz, C. (eds.) Proceedings of 16th European Symposium on Research in Computer Security (ESORICS 2011). *Springer Lecture Notes in Computer Science* **6879** 588–606.

Bryans, J. W., Koutny, M., Mazaré, L. and Ryan, P. Y. A. (2008) Opacity generalised to transition systems. *International Journal of Information Security* **7** (6) 421–435.

Chatzikokolakis, K., Palamidessi, C. and Panangaden, P. (2008) Anonymity protocols as noisy channels. *Information and Computation* **206** (2-4) 378–401.

Chaum, D. (1988) The dining cryptographers problem: Unconditional sender and recipient untraceability. *Journal of Cryptology* **1** 65–75.

Courcoubetis, C. and Yannakakis, M. (1998) Markov decision processes and regular events. *IEEE Transactions on Automatc Control* **43** (10) 1399–1418.

Dubreil, J., Darondeau, P. and Marchand, H. (2010) Supervisory control for opacity. *IEEE Transactions on Automatic Control* **55** (5) 1089 –1100.

Eftenie, A. (2010) Computing quantitative opacity with TPOT – Available at `http://www.polymtl.ca/crac/TPOT/`.

Giro, S. and D'Argenio, P. R. (2009) On the expressive power of schedulers in distributed probabilistic systems. *Electronic Notes in Theoretical Computer Science* **253** (3) 45–71.

Goguen, J. A. and Meseguer, J. (1982) Security policy and security models. In: *Proceedings of IEEE Symposium on Security and Privacy*, IEEE Computer Society Press 11–20.

Hansson, H. and Jonsson, B. (1994) A logic for reasoning about time and reliability. *Formal Aspects of Computing* **6** (5) 512–535.

Kemeny, J. G. and Snell, J. L. (1976) *Finite Markov Chains*, Undergraduate Texts in Mathemetics, Sringer-Verlag.

Lakhnech, Y. and Mazaré, L. (2005) Probabilistic opacity for a passive adversary and its application to Chaum's voting scheme, *Technical Report* 4, Verimag.

Lin, F. (2011) Opacity of discrete event systems and its applications. *Automatica* **47** (3) 496–503.

Lowe, G. (2004) Defining information flow quantity. *Journal of Computer Security* **12** (3-4) 619–653.

Mantel, H. and Sudbrock, H. (2009) Information-theoretic modeling and analysis of interrupt-related covert channels. In: Degano, P., Guttman, J. and Martinelli, F. (eds.) Proceedings of the Workshop on Formal Aspects in Security and Trust, FAST 2008. *Springer Lecture Notes in Computer Science* **5491**67–81.

Mazaré, L. (2005) Decidability of opacity with non-atomic keys. In: Proceedings 2nd Workshop on Formal Aspects in Security and Trust (FAST'04). *Springer International Federation for Information Processing* **173** 71–84.

McIver, A., Meinicke, L. and Morgan, C. (2010) Compositional closure for bayes risk in probabilistic noninterference. In: Abramsky, S., Gavoille, C., Kirchner, C., auf der Heide, F. M. and Spirakis, P. G. (eds.) Proceeding of 37th International Colloquium on Automata, Languages and Programming (ICALP'10), Part II. *Springer Lecture Notes in Computer Science* **6199** 223–235.

Millen, J. K. (1987) Covert channel capacity. In: *Proceedings of IEEE Symposium on Research in Computer Security and Privacy* 144–161.

Reiter, M. K. and Rubin, A. D. (1998) Crowds: Anonymity for web transactions. *ACM Transactions on Information and System Security* **1** (1) 66–92.

Segala, R. (1995) *Modeling and Verification of Randomized Distributed Real-Time Systems.* Ph.D. thesis, MIT, Department of Electrical Engineering and Computer Science.

Smith, G. (2009) On the foundations of quantitative information flow. In: *Proceedings 12th International Conference on Foundations of Software Science and Computational Structures (FOSSACS'09)*, Springer-Verlag 288–302.

Wittbold, J. T. and Johnson, D. M. (1990) Information flow in nondeterministic systems. In *Proceedings of IEEE Symposium on Research in Computer Security and Privacy* 144–161.