A Process-algebraic Approach for the Analysis of Probabilistic Non-interference [1]

Alessandro Aldini[‡] Mario Bravetti[†] Roberto Gorrieri[†]

[‡] Istituto STI, Università di Urbino, Italy,
aldini@sti.uniurb.it
[†] Dipartimento di Scienze dell'Informazione, Università di Bologna, Italy,
{bravetti, gorrieri}@cs.unibo.it

Abstract

We define several security properties for the analysis of probabilistic non-interference as a conservative extension of a classical, nondeterministic, process-algebraic approach to information flow theory. We show that probabilistic covert channels (which are not observable in the nondeterministic setting) may be revealed through our approach and that probabilistic information can be exploited to give an estimate of the amount of confidential information flowing to unauthorized users. Finally, we present a case study showing that the expressiveness of the calculus we adopt makes it possible to model and analyze real concurrent systems.

1 Introduction

The analysis of information flow among different components of a concurrent computer system is a well established approach used for preventing unauthorized disclosure of confidential information (see, e.g., [8, 4, 31, 28]). Among the different proposals which address such an analysis, we consider the notion of *non-interference* [31] that expresses a privacy property of systems. The central idea of non-interference is that a program is secure whenever the variation of the confidential data does not affect the program behavior as observed by an external attacker. The original notion of non-interference was defined for system programs which are deterministic. Generalized notions of noninterference were then designed to include nondeterministic systems on the basis of different system models (see, e.g., [61, 42, 40]) and here we concentrate on the use of process algebras (see, e.g., [36, 48, 47, 23, 45, 50]). In particular, in this paper we consider the nondeterministic approach by Focardi and Gorrieri [23] and their classification of a set of security properties capturing the idea of information

[1]Preliminary results of this paper appeared in: *A. Aldini, "Probabilistic Information Flow in a Process Algebra", Proc. of 12th Int. Conf. on Concurrency Theory (CONCUR 2001), Springer LNCS 2154:152-168* and *A. Aldini, "On the Extension of Non-interference with Probabilities", 2nd ACM SIGPLAN and IFIP WG 1.7 Workshop on Issues in the Theory of Security (WITS 2002).*

flow and non-interference [25]. In short, they employ an extension of CCS [44], where events are partitioned into two different levels of confidentiality (low level and high level), thus allowing the information flowing between the two different levels to be modeled and controlled. As an example, the most intuitive property they advocate is the Non Deducibility on Compositions (*NDC*), whose intuitive idea is that an attacker should not be able to infer anything about the confidential interactions of the system with the high-level environment. Informally, it can be described as follows:

$$\forall \Pi \in \mathit{High\ Users}.\ E \,|\, \Pi \approx E \ \mathit{w.r.t.\ Low\ Users}$$

where E is the model of the system to be considered, $|$ stands for the parallel composition operator, $\approx$ is the equivalence relation on the behavior as observed by low users. The above formula says that E is *NDC* secure if, from the low-level user standpoint, the behavior of the system is invariant with respect to the composition with any high-level user Π.

Possibilistic noninterference for nondeterministic processes abstracts away from further aspects of the system behavior which could be exploited by the attacker to set up a covert channel. For instance, as we will formally show in an example borrowed from [51] (which in turn is inspired by [42]), consider an attack exploiting the frequency of the possible outcomes (deriving from repeated executions of the system) in order to infer confidential information (communicated by high users) that otherwise is not revealed by considering the nondeterministic behavior only. In particular, assume that a secret 1-bit value can be communicated to the unauthorized user among randomly created low-level noise and that both secret value and random low-level noise belong to the same domain (0 and 1):

$$\text{low input } l := (\text{high output } h \texttt{ or } \text{random value}).$$

The high behavior, expressed by the direct assignment of h to l, does not alter the set of possible low outcomes that is always the same with or without the high-level interaction. However, by observing the frequency of each low result deriving from repeated executions of the system, an attacker could be able to infer (with a certain probability) which one is directly communicated by the high user (see Fig. 1). Therefore, while in a nondeterministic setting the system is secure since the choice between the two assignments is left underspecified, in a richer framework, where the probability distribution of the possible events is known, it is possible to reveal an unwanted information leakage.

Another reason for considering probabilities is that a probabilistic approach permits to overcome the classical, qualitative, binary view according to which high information is or is not leaked to low users. More precisely, by computing the probability associated to the high-level behaviors which can be distinguished by low users, the modeler can estimate the level of security of the system. In practice, an observer may check if the probability of detecting an illegal information flow is beyond a threshold for which he considers the system to be

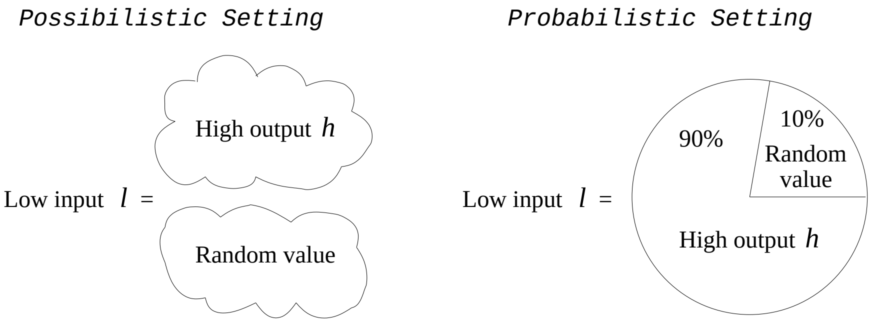


Figure 1: Concealed probabilistic covert channel

secure "enough". For instance, by considering the example above, if the low input l takes the value of the high output h with a negligible probability ε close to 0, then the interference of the high user during the system execution may be considered insufficient to set up a real, effective covert channel from high level to low level. This quantitative approach to information flow analysis is particularly useful for the verification of real systems, where the interplay between high users and low behaviors is more tight than that we expect [49] and the main goal consists of minimizing the illegal interactions, by keeping the inevitable information flows at an acceptable level. As a simple, real example, note that password-based authentication systems are beyond the scope of possibilistic information-flow techniques, which reject programs which do not guarantee absolute secrecy. Indeed, a typical brute-force polynomial-time attack may try to check every possible password to crack the system, but the probability of success, i.e. the potential information leakage, is negligibly small (see, e.g., [60]). As an example, we will formally evaluate the effectiveness of such an attack in case of the authentication system of an automatic teller machine (as done, e.g., in [19]).

In this paper, we apply the approach to information flow theory of [23] to a nondeterministic process algebra and, in the context of a probabilistic extension of such a calculus, we propose a natural, conservative, probabilistic extension of the security properties described in [23]. With respect to [23, 24, 25], which are based on a CCS-like process algebra, here we resort to a CSP-like calculus, that makes it easy and natural the definition of the probabilistic model and is particularly adequate to combine powerful mechanisms (like external/internal choices, multiway synchronization, and asynchronous execution of parallel processes) that are suitable to describe the behavior of real, complex systems (as emphasized, e.g., in [1]). The extension we propose is conservative in the sense that by removing probabilistic information from a system P of our probabilistic calculus we obtain the nondeterministic variant P_{nd} of P, on which the approach to information flow theory based on nondeterminism of [23] may be applied by obtaining results which are fully consistent with those related to the security analysis of P. In particular, if P satisfies a given security property S in our probabilistic framework, then P_{nd} satisfies the corresponding nondeterministic property S_{nd} and, in general, in the probabilistic setting we have the same inclusion relationships among security properties as in the nondeterministic setting. Therefore, as we will show through several examples, our probabilistic security properties capture both information flows which are observable by considering the logical behaviors of the system and information leakage due to the probabilistic behavior of the system only. Moreover, we show how probabilities can be exploited to achieve additional information related to the information flowing through the system, like, e.g., the security level of the system in the sense we specified above.

Finally, all the main features of our probabilistic approach are emphasized in a case study. More precisely, we model the algebraic specification of a router working at the network layer and implementing a probabilistic routing mechanism that handles both confidential and public data. With these assumptions in view, we analyze different versions of such a system that may reveal potential possibilistic and probabilistic information flows from high level to low level.

1.1 Related Work

A significant amount of work has been done in order to extend the relation among potential information flows and different aspects of concurrent systems such as time and probabilities. As an example, programs which execute several concurrent processes introduce the potential for a malicious party to observe the internal timing behaviors of programs, via the effect that the running time of commands might have on the scheduler choice of whenever various concurrent alternatives can be executed (see, e.g., [26, 22, 54]). On the other hand, real systems may exhibit probabilistic covert channels that are not ruled out by standard nondeterministic security models [32, 61]. In particular, Gray [32] proposes in the context of a probabilistic version of Millen's synchronous state machine [43] two definitions of security: (*i*) the Probabilistic Non-interference (*PNI*), which intuitively says that different behaviors of the high part of the environment do not affect the probability distribution of the low events (such an intuition, as we will see, will be rephrased in our process-algebraic setting); (*ii*) the Applied Flow Model (*AFM*), which is an interpretation of the Flow Model of [42], saying that the probability of a low output may depend on previous low events, but not on previous high events. Gray proves that *AFM* is stronger than *PNI* (similarly, we will present a notion of security stronger than our definition of *PNI*). Moreover, Jürjens [37] considers the notion of *PNI* of [32] in the setting of a model of probabilistic dataflow, by showing the compositionality of *PNI* in such a context and that a simpler nonprobabilistic notion of non-interference, called Nondeducibility on Strategies [61], is an instantiation of *PNI*.

Additionally, Sabelfeld and Sands [52] say that "since a scheduler's behavior might be probabilistic, this opens up an opportunity for the attacker to set up a probabilistic covert channel whereby information is obtained by statistical inferences drawn from the relative frequency of outcomes of a repeated computation". Therefore, they formalize the idea of confidentiality for a simple imperative language with dynamic thread creation. More precisely, they develop extensional semantics based specifications of secure information flow for multi-threaded programs. Such specifications capture probabilistic information flows that arise from the scheduling of concurrent threads. The novelty of their approach is the adoption of a probabilistic notion of bisimulation in formalizing security conditions (as previously done in [23] in a nondeterministic setting). In [55], a type system is presented that aims to ensure secure information flow in a simple multi threaded imperative programming language running under a uniform probabilistic scheduler. The same author also proposes a definition of weak probabilistic bisimulation (inspired by [7]) in [56] to justify the correctness of a type system for secure information flow. In the same line of the above discussion, [16] resorts to a possibilistic information flow analysis of a Probabilistic Idealised Algol to check for probabilistic interference, whereas in [18, 19] a probabilistic security analysis is proposed in the framework of a declarative programming language. Finally, in [33, 58] a modal logic is set out for reasoning about multilevel security of probabilistic systems.

However, to the best of our knowledge, no approach has been proposed for the definition of the information flow theory in the context of a probabilistic process algebra. We think that the definition and the analysis of information flow security properties for concurrent systems could benefit by the well-established support given by the known results and formal techniques developed in (probabilistic) process algebras that, as the literature has shown in the nondeterministic case (see, e.g., [45, 24, 50]), offer the adequate tool for a clean and rigorous formalization of a generalized notion of security.

1.2 Outline

The remainder of the paper is organized as follows. We start by introducing a simple nondeterministic process algebra and by providing as a background the definition of three security properties in the same line of the nondeterministic approach of [23] (Sect. 2). Then, we present the probabilistic extension [2, 10, 9] of the nondeterministic calculus of Sect. 2, obtained by adding probabilistic parameters to the algebraic operators (Sect. 3). As far as the equivalence relation we adopt is concerned, we resort to a probabilistic variant of the weak bisimulation (Sect. 4), the main reason being that weak bisimulation has been proved to be particularly suitable to deal with some kind of information flow for nondeterministic processes [23, 24]. Based on such a notion of equivalence, we define the probabilistic version of the properties of Sect. 2 and we make a comparison between the nondeterministic setting and such a probabilistic framework (Sect. 5). Afterwards, we present the case study on probabilistic routing and we analyze the potential information flows from high level to low level that may be revealed by different versions of the same system (Sect. 6). Finally, we report some conclusions (Sect. 7) and the proofs of results (see the appendix).

2 Possibilistic Security Properties

In this section, we rephrase the possibilistic approach to information flow theory of [23, 24, 25], which employ a CCS-like calculus, in the context of a nondeterministic process algebra which is a synthesis of CCS [44] and CSP [35].

2.1 A Nondeterministic Calculus

In our nondeterministic calculus, actions are explicitly partitioned into output actions and input actions and a multiway cooperation discipline is applied in order to allow synchronizations involving (at most) one output action and (possibly several) input actions to be performed. As we will see, these features simplify the definition of the probabilistic model when extending the nondeterministic process algebra with probabilities. As usual in security models, we distinguish between high-level visible actions and low-level visible actions in order to allow the analysis of the information flow between parties with different levels of confidentiality to be performed.

Formally, we denote the set of action types by *AType*, ranged over by a, $b, \ldots$. As usual, *AType* includes the special type τ denoting an internal action. We define a set of input actions by $I = \{a_* \mid a \in \mathit{AType} - \{\tau\}\}$ and a set of output actions by $O = \mathit{AType} - \{\tau\}$. The complete set of actions is denoted by $\mathit{Act} = I \cup O \cup \{\tau\}$, ranged over by π, $\pi', \ldots$. Moreover, we denote two disjoint sets AType_H and AType_L of high-level and low-level action types which form a covering of $\mathit{AType} - \{\tau\}$, such that $a \in O$ and $a_* \in I$ are high (low) actions if $a \in \mathit{AType}_H$ ($a \in \mathit{AType}_L$). Throughout the paper, we consider h, $h', \ldots$ to be high-level types and l, $l', \ldots$ to be low-level types.

Let *Const* be a set of constants, ranged over by $A, B, \ldots$. The set $\mathcal{L}_{nd}$ of process terms of our calculus is ranged over by P_{nd}, $Q_{nd}, \ldots$ (when it is clear from the context we omit the subscript nd) and is generated by the syntax:

$$P ::= \underline{0} \mid \pi.P \mid P + P \mid P \parallel_S P \mid P/L \mid A$$

where $S, L \subseteq \mathit{AType} - \{\tau\}$.

As far as an informal intuition of the operators is concerned, $\underline{0}$ represents the terminated or deadlocked term, $\pi.P$ is the classical prefix operator, $P + Q$ is a CCS-like alternative composition operator expressing a nondeterministic choice between P and Q, and $P \parallel_S Q$ is a CSP-like parallel composition operator, where processes P and Q:

- locally execute their input and output actions of type $a \notin S$;

- are required to synchronize over visible actions of the same type $a \in S$. In particular, a synchronization between two actions of type $a \in S$ may occur only if either they are both input actions a_* (and the result is an input action a_*), or one of them is an output action a and the other one is an input action a_* (and the result is an output action a).

Hence, our parallel operator allows multiway synchronizations composed by all input actions except at most one output action to be performed. In particular, the synchronization policy we consider is *asymmetric*, in that by following a so-called *master-slave cooperation discipline* a process can broadcast a given output action a, while each of the other processes reacts by synchronizing through an input action a_* [10, 12]. Moreover, we point out that the CSP-like parallel operator can be used to express the restriction of actions. In particular, $P \parallel_L \underline{0}$ represents a system executing the local actions which the lefthand process P can perform except for the actions whose type belongs to the synchronization set L, which are not allowed to synchronize because the righthand process is the null term $\underline{0}$. Therefore, for the sake of simplicity we use the abbreviation $P \backslash L$, denoting a system P which prevents the execution of the actions belonging to the type set L, to stand for the expression $P \parallel_L \underline{0}$.

Then, we have the hiding operator P/L which turns the actions of type $a \in L$ into internal actions τ. Finally, constants A are used to specify recursive systems.

In the following, we restrict ourselves to the set $\mathcal{G}_{nd} \subseteq \mathcal{L}_{nd}$ of closed and guarded terms, i.e. we require that every constant has exactly one defining equation (of the form $A \stackrel{\Delta}{=} P$) and every constant occurrence is within the scope of an action prefix operator [44]. We denote by $sort(P)$, with $P \in \mathcal{G}_{nd}$, the set of actions which syntactically occur in the action prefix operators within P, and by $\mathcal{G}_{H_{nd}} = \{P \in \mathcal{G}_{nd} \mid sort(P) \subseteq AType_H\}$ the set of high-level terms (i.e. containing high-level actions only). If $P \xrightarrow{\pi_1} \ldots \xrightarrow{\pi_n} P'$, with $\pi_1, \ldots, \pi_n$ a (possibly empty) sequence of actions in Act, then P' is a *derivative* of P. We denote by $Der(P)$ the set of derivatives of P. Finally, we define a function $t : Act \rightarrow AType$ that maps actions onto the corresponding action types, namely $t(\pi) = \pi$ if $\pi \notin I$, and $t(a_*) = a$ with $a_* \in I$.

The formal semantics of our nondeterministic process algebra is given by the labeled transition system $(\mathcal{L}_{nd}, Act, \rightarrow)$, whose states are terms of the process algebra and the transition relation $\rightarrow \subseteq \mathcal{L}_{nd} \times Act \times \mathcal{L}_{nd}$ is the minimal relation satisfying the operational rules reported in Table 1.

In order to define security properties in the context of our calculus, we need a notion of equivalence relation. In particular, since we concentrate on the observational power of the low user (which is limited to the low-level events, while the high-level activities are considered as unobservable τ actions) we need a notion of equivalence which abstracts from τ actions and focuses only on visible behaviors. Due to this, we employ a weak bisimulation equivalence [44] (instead of, e.g., a strong bisimulation). Moreover, as shown in [24], the reason for resorting to a weak bisimulation equivalence stems from some deficiencies typical of other notions of equivalence, such as the trace equivalence. The notion of bisimulation is finer than trace equivalence and is able to detect a higher number of information flows (e.g., trace equivalence is not able to detect high-level deadlocks). As far as other notions of equivalence are concerned, [24] shows that failure/testing equivalences are not adequate for systems with some high-level loops or with τ loops.

Pre	$\pi.P \xrightarrow{\pi} P$	
Sum	$\dfrac{P \xrightarrow{\pi} P'}{P + Q \xrightarrow{\pi} P'}$	$\dfrac{Q \xrightarrow{\pi} Q'}{P + Q \xrightarrow{\pi} Q'}$
Par	$\dfrac{P \xrightarrow{\pi} P'}{P \parallel_S Q \xrightarrow{\pi} P' \parallel_S Q} \; t(\pi) \notin S$	$\dfrac{Q \xrightarrow{\pi} Q'}{P \parallel_S Q \xrightarrow{\pi} P \parallel_S Q'} \; t(\pi) \notin S$
	$\dfrac{P \xrightarrow{a} P' \; Q \xrightarrow{a_*} Q'}{P \parallel_S Q \xrightarrow{a} P' \parallel_S Q'} \; a \in S$	$\dfrac{P \xrightarrow{a_*} P' \; Q \xrightarrow{a} Q'}{P \parallel_S Q \xrightarrow{a} P' \parallel_S Q'} \; a \in S$
	$\dfrac{P \xrightarrow{a_*} P' \; Q \xrightarrow{a_*} Q'}{P \parallel_S Q \xrightarrow{a_*} P' \parallel_S Q'} \; a \in S$	
Hid	$\dfrac{P \xrightarrow{\pi} P'}{P/L \xrightarrow{\pi} P'/L} \; t(\pi) \notin L$	$\dfrac{P \xrightarrow{\pi} P'}{P/L \xrightarrow{\tau} P'/L} \; t(\pi) \in L$
Con	$\dfrac{P \xrightarrow{\pi} P'}{A \xrightarrow{\pi} P'} \; \text{if } A \stackrel{\Delta}{=} P$	

Table 1: Operational semantics for the nondeterministic calculus

In the following, given $A \subseteq Act^*$, we say that $P \stackrel{A}{\Rightarrow} P'$ if and only if there exists $w \in A$ (with $w = w_1 \ldots w_n$ for some n) and terms P_i (with $1 \leq i \leq n$), such that $P_i \xrightarrow{w_i} P_{i+1}$ and $P \equiv P_1$ and $P' \equiv P_{n+1}$. Moreover, given $\pi \in Act$, we use the abbreviation $P \stackrel{\pi}{\Longrightarrow} P'$ to stand for $P \stackrel{\tau^*\pi}{\Longrightarrow} P'$. In particular, $P \stackrel{\hat{\pi}}{\Longrightarrow} P'$ stands for $P \stackrel{\pi}{\Longrightarrow} P'$ if $\pi \in Act - \{\tau\}$ and for $P \stackrel{\tau^*}{\Longrightarrow} P'$ if $\pi = \tau$. Note that $P \stackrel{\hat{\tau}}{\Longrightarrow} P'$ means that zero or more τ labeled transitions occur, while $\stackrel{\tau}{\Longrightarrow}$ requires at least one τ labeled transition.

We now report the classical definition of weak bisimulation over terms of our nondeterministic calculus.

Definition 2.1 A relation $R \subseteq \mathcal{G}_{nd} \times \mathcal{G}_{nd}$ is a weak bisimulation if $(P, Q) \in R$ implies for all $\pi \in Act$:

- whenever $P \xrightarrow{\pi} P'$, then there exists $Q' \in \mathcal{G}_{nd}$ such that $Q \stackrel{\hat{\pi}}{\Longrightarrow} Q'$ and $(P', Q') \in R$,
- conversely, whenever $Q \xrightarrow{\pi} Q'$, then there exists $P' \in \mathcal{G}_{nd}$ such that $P \stackrel{\hat{\pi}}{\Longrightarrow} P'$ and $(P', Q') \in R$.

Two terms $P, Q \in \mathcal{G}_{nd}$ are weakly bisimulation equivalent, denoted $P \approx_B Q$, if there exists a weak bisimulation R containing the pair (P, Q). It is provable that $\approx_B$ is an equivalence relation [44].

Finally, we use the expression $P \stackrel{\text{iso}}{=} Q$ to say that P and Q are isomorphic, i.e. the labeled transition systems derived from P and Q are structurally equal (there exists a bijection between the two state spaces such that any pair of corresponding states have identically labeled transitions toward any pair of corresponding states).

2.2 Analyzing Possibilistic Security Properties

Based on the above nondeterministic process algebra and the related notion of equivalence, we now define three security properties in the line of [23]. We start with the Bisimulation Strong Nondeterministic Non-interference (*BSNNI*), which informally says that a system is secure if what the low-level part can see does not depend on what the high-level part can do. In our process-algebraic framework it may be expressed as follows.

Definition 2.2 $P \in \mathcal{G}_{nd}$ is *BSNNI* secure if and only if

$$P / AType_H \approx_B P \backslash AType_H .$$

In the above definition $P / AType_H$ represents a system where the high-level actions are hidden, while $P \backslash AType_H$ represents a system where the execution of the high-level actions is prevented. Therefore, if a system P satisfies the *BSNNI* property then, from the low-level user viewpoint, the following condition is preserved: for each behavior in $P \backslash AType_H$ which is observable whenever the high-level user does not interact with the system, we have an equivalent behavior in $P/AType_H$ which is observable whenever the high-level user performs actions not visible by the low view, and viceversa. In essence, a low-level user cannot infer, by observing the low view of the system, that some high-level activity has occurred.

Example 2.3 Let us consider the system $P \stackrel{\Delta}{=} l.\underline{0} + h.l'.\underline{0}$ that, from the low-level standpoint, nondeterministically performs either an action l or an action l' if the high-level output h has previously occurred. When analyzing the security properties of P, we make the following assumptions. Since we employ a calculus where communications are synchronous, system P interacts with the external environment by offering high (low) outputs which have to be accepted by the high (low) user and by accepting high (low) inputs which come from the high (low) user. For instance, in the initial state of our example, system P can either execute the output action l if the low user is available to accept it, or execute the output action h if the high user is available to accept it. Hence, we check if the interplay between the high user and the high part of the system can affect the view of the system as observed by a low user. Then, we also assume that the low user knows how the system P is defined, and we check if, in spite of this, he is not able to infer the behavior of the high user by observing the low view of the system at run time. With these assumptions in view, we can verify that P is clearly not secure, because depending on which output action between l and l' is executed by the system, a low user can infer that the high user has (has not) accepted the output action h. Formally, we have that $P/AType_H \stackrel{\text{iso}}{=} l.\underline{0} + \tau.l'.\underline{0} \not\approx_B l.\underline{0} \stackrel{\text{iso}}{=} P \backslash AType_H$.

Another example of an unwanted information flow is given by the system $P \stackrel{\Delta}{=} l.P + h.\underline{0}$, where a possible deadlock can be caused by the execution of the high-level action h. Aware of such a potential behavior, a low user knows that no high-level action has been performed each time he observes a low-level action l. This is more than enough to set up an information flow from high level to low level. Formally, we have that $P/AType_H \stackrel{\text{iso}}{=} l.P + \tau.\underline{0} \not\approx_B l.P \stackrel{\text{iso}}{=} P \backslash AType_H$. Note that the same system turns out to be *BSNNI* secure if the equivalence relation we employ is the classical trace equivalence that, hence, is not adequate to reveal high-level deadlocks.

Finally, an example of a system which is *BSNNI* secure is given by $P \stackrel{\Delta}{=} l.P + h.l.P$. Indeed, the low-level view of P consists of the repeated execution of action l, that is not enough to infer anything about the high-level behavior. Formally, $P/AType_H \stackrel{\text{iso}}{=} l.P + \tau.l.P \approx_B l.P \stackrel{\text{iso}}{=} P \backslash AType_H$. ■

Other properties have been suggested in order to capture different behaviors of the system that have to be considered insecure. For instance, as already seen in the introduction, one of the most established security properties is the Non Deducibility on Composition, which says that the low view of a system P in isolation is not to be altered when considering each potential interaction of P

with the high users of the external environment. Here, we consider a notion of the *NDC* property based on the bisimulation equivalence, called Bisimulation *NDC* (*BNDC*). A formalization of such a property in the context of our calculus may be as follows.

Definition 2.4 $P \in \mathcal{G}_{nd}$ is *BNDC* secure if and only if

$$P \backslash AType_H \approx_B ((P \,\|_S\, \Pi)/S) \backslash AType_H, \forall \Pi \in \mathcal{G}_{H_{nd}}, \forall S \subseteq AType_H.$$

The left-hand term represents the low view of the system in isolation, while the right-hand term expresses the low view of P interacting with the high environment (note that the activities resulting from such interactions are made invisible). Such a formalization is compliant to the definition of *BNDC* given in [25, 24], where the only difference is the adoption of a CCS-like parallel operator. In particular, due to the CSP-like parallel operator of our calculus, we first hide the high-level actions belonging to S which succeed in synchronizing in $P \,\|_S\, \Pi$ and then we purge the system of the remaining high-level actions.

Example 2.5 Term $P \stackrel{\Delta}{=} l.\underline{0} + h.h.l.\underline{0}$ is *BSNNI* secure. However, the high-level part can alter the low view of the system in case the high-level user first executes the first high-level action h and then blocks the execution of the second high-level action h. In such a case a deadlock is caused by the high user that prevents the low-level action l from being observed by the low user. This information flow is captured by the *BNDC* property if we take $\Pi \stackrel{\Delta}{=} h_*.\underline{0}$ and $S = \{h\}$. Indeed, in such a case we have $P \backslash AType_H \stackrel{\text{iso}}{=} l.\underline{0} \not\approx_B l.\underline{0} + \tau.\underline{0} \approx_B (((l.\underline{0} + h.h.l.\underline{0}) \,\|_S (h_*.\underline{0}))/S) \backslash AType_H$. ■

Because of the presence of the universal quantification on all possible high-level processes, the *BNDC* property could be difficult to be used in practice. Therefore, stronger formulations of such a property have been proposed that avoid the adoption of universal quantification on all possible high-level processes and reveal additional potential information flows that some authors in the literature consider to be insecure. Here, we decide to consider the Strong *BNDC* (*SBNDC*), introduced in [23], that does not involve composition with each possible high-level process Π and is the strongest security property defined by the authors of [23]. In [29, 46] the same property under the name of *strong local non-interference* (*SLNI*) has been proposed for the analysis of non-interference in the context of CSP [35]. Like the *BNDC* property, the *SBNDC* checks the condition which says that whatever the high-level part does, the set of nondeterministic behaviors which are observable by the low user should not change. Moreover, as a stricter requirement, it also checks the condition which says that whenever the low user observes the system behavior, he should not distinguish which, if any, high-level event has occurred at some point in the past. We will clarify the difference between such requirements through some examples. The formal definition of the *SBNDC* property in the context of our calculus is as follows.

Definition 2.6 $P \in \mathcal{G}_{nd}$ is *SBNDC* secure if and only if $\forall P' \in Der(P)$ and $\forall P''$ such that $P' \xrightarrow{\pi} P''$, $t(\pi) \in AType_H$, then

$$P' \backslash AType_H \approx_B P'' \backslash AType_H.$$

With respect to the definition of *BNDC*, where we have to check the property for an infinite set of terms, i.e. $\forall\, \Pi \in \mathcal{G}_{H_{nd}}$, the condition needed by the *SBNDC* requires to check a finite number of terms, i.e. each $P' \in Der(P)$, whenever P is associated to a finite state transition system.

Example 2.7 Let us consider term $P \stackrel{\Delta}{=} \tau.(\tau.\underline{0} + \tau.l.\underline{0}) + \tau.h.l.\underline{0}$ from which the labeled transition system of Fig. 2(a) is derived. P is *BSNNI* secure. Formally, $P/AType_H \stackrel{\text{iso}}{=} \tau.(\tau.\underline{0}+\tau.l.\underline{0})+\tau.\tau.l.\underline{0}$ and, by following the τ-laws of [44], we have $\tau.(\tau.\underline{0} + \tau.l.\underline{0}) + \tau.\tau.l.\underline{0} \approx_B \tau.(\tau.\underline{0} + \tau.l.\underline{0}) \approx_B \tau.(\tau.\underline{0} + \tau.l.\underline{0}) + \tau.\underline{0} \stackrel{\text{iso}}{=} P \backslash AType_H$. Moreover, it is provable that P is *BNDC* secure too. Informally, with or without the interaction of high users which accept the output action h, the system evolves either into a deadlock state or into a state where l, which is the only action observable by a low user, is executed. However, P does not satisfy the *SBNDC* property, because the low view in state $P' \stackrel{\Delta}{=} h.l.\underline{0}$, reachable from P through the execution of an internal action, depends on whether the high-level output h is executed. More precisely, a transition labeled with the action h leads from P' to $P'' \stackrel{\Delta}{=} l.\underline{0}$ and $P' \backslash AType_H \stackrel{\text{iso}}{=} \underline{0} \not\approx_B l.\underline{0} \stackrel{\text{iso}}{=} P'' \backslash AType_H$. Intuitively, it can be debatable whether we should classify P as an insecure system. For instance, one may argue for this being an information flow because, once the internal nondeterminism is resolved, the system can evolve into a state where a high user can block (or delay) the execution of the low-level action l, thus creating a covert channel. On the other hand, it is worth noting that the high-level part has no control over the resolution of the nondeterminism in term P. ■

Example 2.8 As another example, in Fig. 2(b) we show the labeled transition system associated to term $Q \stackrel{\Delta}{=} h.l.\underline{0} + \tau.l.\underline{0} + l'.\underline{0}$, which is *BSNNI* secure ($Q \backslash AType_H \stackrel{\text{iso}}{=} \tau.l.\underline{0} + l'.\underline{0} \approx_B \tau.l.\underline{0} + \tau.l.\underline{0} + l'.\underline{0} \stackrel{\text{iso}}{=} Q/AType_H$) and *BNDC* secure too. In particular, for any high user which interacts with Q by accepting the high-level output h, the low view of the system consists of a nondeterministic choice between l' and an internal action which leads to the execution of action l. On the other hand, for any high user which does not allow the output action h to be executed, the low view of the system still remains the same. However, we cannot consider Q as a secure system, because a low user which observes the action l' can deduce that the high user has not performed the action h. In other words, the low view of the system reveals the past behavior of the high user.

Such an information flow is captured by the *SBNDC* property. Formally, the execution of the high-level action h leads from Q to $Q' \triangleq l.\underline{0}$, but $Q \backslash AType_H \stackrel{\text{iso}}{=} \tau.l.\underline{0} + l'.\underline{0} \not\approx_B l.\underline{0} \stackrel{\text{iso}}{=} Q' \backslash AType_H$. ■

Example 2.9 As a final example, which has been proposed in [46], let us consider the following system $R \triangleq \tau.l_0.R + h_{0_*}.l_0.R + \tau.l_1.R + h_{1_*}.l_1.R$, where the high-level environment repeatedly communicates a value (0 or 1) to the low-level part amongst randomly created noise (see the underlying labeled transition system in Fig. 2(c)). One can believe that R has no information flow if we consider the possible behaviors of the system only. Indeed R is both *BSNNI* ($R \backslash AType_H \approx_B \tau.l_0.R + \tau.l_1.R \approx_B R/AType_H$) and, as it is easy to see, *BNDC* secure. Anyway, each time the action l_0 (l_1) is executed a low user infers that the high-level input h_{1_*} (h_{0_*}) has not taken place. Moreover, in [46] it is suggested that the low user should be able to infer the high behavior whenever the high user always offers the input h_{0_*} with respect to the input h_{1_*}. Formally, the system R does not satisfy the *SBNDC* property. For instance, R can execute the high-level action h_{0_*} thus reaching the state $R' \triangleq l_0.R$, but $R \backslash AType_H \stackrel{\text{iso}}{=} \tau.l_0.R + \tau.l_1.R \not\approx_B l_0.R \stackrel{\text{iso}}{=} R' \backslash AType_H$. ■

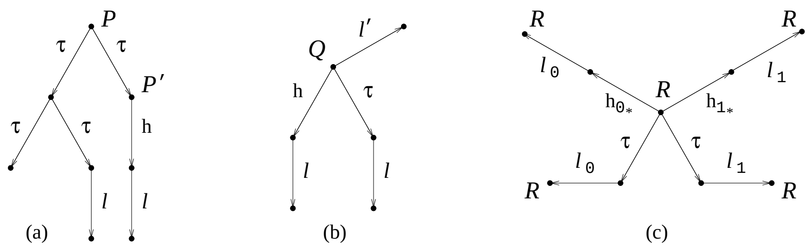


Figure 2: Examples of systems that are *BNDC* secure, but not *SBNDC* secure

We conclude this introductory section by observing that the inclusion relationships for the bisimulation-based security properties depicted in Fig. 3 hold.

Proposition 2.10 *SBNDC* $\subset$ *BNDC* $\subset$ *BSNNI*.

In this section we proposed the nondeterministic non-interference property, which is the interpretation in a process-algebraic setting of the non-interference notion of [31], the nondeducibility on composition property, which we consider the most intuitive property proposed in [23], and a strong (and easier to be verified) version of such a property [23]. We conclude by observing that many other properties exist and they all can be rephrased in our nondeterministic process algebra by preserving the inclusion relationship shown in [23, 25].

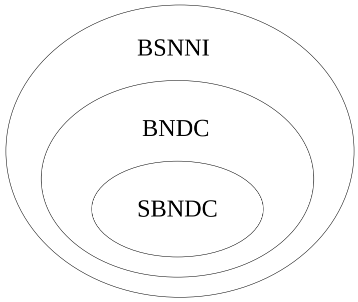


Figure 3: Inclusion relationship among nondeterministic properties

3 A Calculus with Probabilities

In this section, we consider a probabilistic extension of the nondeterministic process algebra of Sect. 2.1. Such a calculus is a slight variant of a process algebra proposed in [2, 10, 9] and expresses both nondeterministic behaviors and probabilistic information. This is a novelty with respect to the existing models used to formalize a probabilistic notion of non-interference, which do not model nondeterminism but just probabilistic behaviors.

As we will see, the most natural and intuitive way to define such an extension consists of:

1. adding probabilistic information to the algebraic operators;
2. adopting a model of probabilities, that turns out to be a mixture of the classical generative and reactive models of [30].

In particular, we assume that output actions behave as *generative* actions: a process which behaves generatively autonomously decides, on the basis of a probability distribution, which action will be performed. Moreover, we assume that input actions behave as *reactive* actions: a process which behaves reactively just reacts internally to the action chosen by a generative component of its environment (say a) on the basis of a probability distribution associated with the reactive actions of type a it can perform. Therefore, since the choice of the action type is completely nondeterministic, as it depends on the environment, we see the reactive actions as *incomplete* actions that must synchronize with generative actions of another system component in order to form a closed system (i.e. a system not including nondeterministic choices that have to be resolved by the environment). Then, a system is considered to be *fully specified* when it gives rise to a probabilistic transition system that is purely generative, in the sense that it executes complete actions only. A fully specified system is therefore a fully probabilistic system from which a Markov Chain, which can be analyzed to get performance measures of the system, can be easily derived by discarding actions from transition labels.

In the remainder of this section, we introduce the models, called *generative-reactive transition systems*, associated to terms of the calculus by the operational semantics, and then we present the syntax and the semantics. Finally, we equip the calculus with a weak probabilistic bisimulation equivalence in the line of [7].

3.1 Generative-reactive Transition Systems

Generative-reactive transition systems are composed of transitions labeled with an action, which can be either output or input, and a probability. According to our choice of considering input actions as reactive actions and output actions as generative actions, we denote the set of reactive actions by $RAct = I$ and the set of generative actions by $GAct = O \cup \{\tau\}$. Note that the action τ is generative, because it expresses an internal move autonomously executed by the process without interactions with the environment. Moreover, it turns out that $Act = RAct \cup GAct$.

Transitions leaving a state are grouped into several bundles. We have a single generative bundle composed of all the transitions labeled with a generative action, and several reactive bundles, each one referring to a different action type a and composed of all the transitions labeled with a_*. A bundle of transitions expresses a probabilistic choice. On the contrary, the choice among bundles is performed non-deterministically.

Definition 3.1 A generative-reactive transition system ($GRTS$, for short) is a triple $(S, AType, T)$ where S is a set of states, $AType$ is a set of action types, and $T \in \mathcal{M}(S \times Act \times]0,1] \times S)$ is a multiset[2] of probabilistic transitions, such that:

1. $\forall s \in S, \forall a_* \in RAct,\ \sum \{\!| p \mid \exists t \in S : (s, a_*, p, t) \in T |\!\} \in \{0, 1\}$
2. $\forall s \in S,\ \sum \{\!| p \mid \exists a \in GAct, t \in S : (s, a, p, t) \in T |\!\} \in \{0, 1\}$

A rooted $GRTS$ is a tuple $(S, AType, T, s_0)$, with $s_0 \in S$ the initial state and $(S, AType, T)$ a $GRTS$.

Requirement 1 defines the reactive bundles (one for each action type) and requirement 2 defines the unique generative bundle. Both requirements say that for each state the probabilities of the transitions composing a bundle, if there are any, sum up to 1 (otherwise the summation over empty multisets is defined equal to 0). As an example, in Fig. 4 we show a $GRTS$ composed of a generative bundle enabling three transitions a, b, and c, a reactive bundle of type e enabling two transitions e_*, and a reactive bundle of type d enabling a single transition d_*. Graphically, transitions of the same bundle are grouped by an arc, and

[2] We use "$\{\!|$" and "$|\!\}$" as brackets for multisets and $\mathcal{M}(S)$ to denote the collection of multisets over S. We use multisets to handle the particular case in which from a state s several transitions (labeled with the same action π and probability p) lead to a state t.

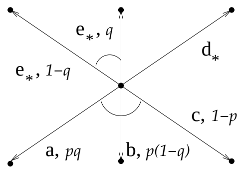


Figure 4: Example of a generative-reactive transition system

the probability of a transition is omitted when equal to 1. We point out that a fully specified system (i.e. a system that does not include reactive bundles) is *performance closed*, in the sense that it gives rise to a fully probabilistic transition system that does not include nondeterministic choices.

Finally, it is easy to see that from a generative-reactive transition system we can derive a labeled transition system $(\mathcal{L}_{nd}, Act, \rightarrow)$ as those described in Sect. 2.1 by simply removing the probabilities from the labels of the transitions.

3.2 Syntax and Informal Semantics

The set $\mathcal{L}$ of process terms of our calculus, ranged over by P, $Q, \ldots$, is generated by the syntax:

$$P ::= \underline{0} \mid \pi.P \mid P +^p P \mid P \parallel^p_S P \mid P/^p_a \mid A$$

where $S \subseteq \mathit{AType} - \{\tau\}$, $a \in \mathit{AType} - \{\tau\}$, and $p \in]0,1[$. We denote by $\mathcal{G}$ the set of guarded and closed terms (i.e. the set of processes of $\mathcal{L}$) and by $\mathcal{G}_H = \{P \in \mathcal{G} \mid \mathit{sort}(P) \subseteq \mathit{AType}_H\}$ the set of high-level processes, i.e. including high-level actions only.

An informal overview of the operators is very similar to that given in Sect. 2.1, except for the fact that here we have probabilistic parameters attached to the operators that we employ to resolve the choices in each process term. Indeed, it is worth noting that from any term $P \in \mathcal{L}$ we can derive a corresponding term $P_{nd} \in \mathcal{L}_{nd}$ by removing the probabilistic parameters from the operators in P. As an example, process $P \stackrel{\Delta}{=} l.\underline{0} +^p h.h.l.\underline{0}$ is a term in $\mathcal{L}$ and its nondeterministic counterpart is process P_{nd} of Example 2.5. Now, we show the intuitive meaning of each operator.

$\underline{0}$ represents a terminated or deadlocked term having no transitions.

The prefix operator $\pi.P$ performs the action π with probability 1 and then behaves like P.[3]

The alternative composition operator $P +^p Q$ expresses a probabilistic choice between the generative actions of P and Q and between the reactive actions of P and Q of the same type. In particular, we have the following cases:

[3]Usually, we abbreviate terms $\pi.\underline{0}$ by omitting the final $\underline{0}$.

1. if both P and Q can execute generative actions, then $P +^p Q$ chooses a generative action of P with probability p and a generative action of Q with probability $1 - p$. See, e.g., Fig. 5(a);

2. if one process P or Q cannot execute generative actions, $P +^p Q$ chooses a generative action of the other process with probability 1 (similarly as in [6]);

3. as far as reactive actions of a given type a are concerned, $P +^p Q$ chooses between the reactive actions a_* of P and Q according to probability p, by following the same mechanism seen for generative actions. See, e.g., Fig. 5(b);

4. the choice among generative and reactive actions or among reactive actions of different types is just nondeterministic. See, e.g., Fig. 5(c) which shows purely nondeterministic choices and Fig. 5(d) which shows a mixed nondeterministic/probabilistic choice.

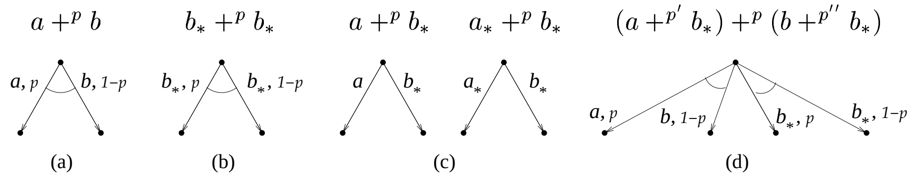


Figure 5: Examples of *GRTS*s derived from alternative composition

The parallel composition operator $P \,\|_S^p\, Q$ performs the actions of its components P and Q by following the synchronization policy described in Sect. 2.1 and, for the actions of P and Q not in S, the same probabilistic mechanism seen for alternative composition. Moreover, it follows these additional rules:

1. since the execution of some generative action of P (Q) can be prevented in $P \,\|_S^p\, Q$, the probabilities of executing the remaining generative actions of P (Q) are proportionally redistributed so that their overall probability sums up to 1 (as standard when restricting actions in the generative model [30]);

2. in case of synchronizing generative actions a of P (Q), their probability is distributed among the multiple actions a obtained by synchronizing with reactive actions a_* executable by Q (P), according to the probability the actions a_* are chosen in Q (P);

3. in case of synchronizing reactive actions of a given type $a \in S$, if both P and Q can execute some action a_*, the choice of the two actions a_* of P

and Q forming the actions a_* executable by $P \,||_S^p\, Q$ is made according to the probability that they are *independently* chosen by P and Q.

Example 3.2 In Fig. 6(a) we report the *GRTS* derived from term $((a +^q b) +^{q'} c_*) \,||_{\{c\}}^p\, c$ that may execute the two non-synchronizing generative actions a and b of the lefthand process and the synchronizing generative action c of the righthand process (note that the execution of c is allowed by the reactive action c_* of the lefthand process). As in case of alternative composition, since both processes may execute some generative actions, we perform a generative action of the lefthand process with probability p and a generative action of the righthand process with probability $1-p$. Since the generative actions executable by $(a +^q b) +^{q'} c_*$ are a with probability q and b with probability $1-q$, the composed system executes action a with probability $p \cdot q$, action b with probability $p \cdot (1-q)$ and action c with probability $1-p$.

As another example, in Fig. 6(b) we show the *GRTS* derived from the term $(a +^r b) \,||_{\{a,b\}}^p\, (b_*.c +^q b_*.d)$, where only the lefthand process may execute some generative action (hence parameter p is not considered). In particular, since the righthand process does not allow the action a to synchronize, we execute only the generative action b with probability 1 (hence parameter r is not considered). Such an action synchronizes with one of the two reactive actions b_* of the righthand process chosen according to probability q. Therefore, we execute the action b leading to $\underline{0} \,||_{\{b\}}^p\, c$ with probability q and the action b leading to $\underline{0} \,||_{\{b\}}^p\, d$ with probability $1-q$. ■

Figure 6: Examples of *GRTS*s derived from parallel composition

The hiding operator $P/_a^p$ turns generative and reactive actions of type a into generative actions τ, by changing the probabilities according to these rules:

- as far as generative actions executable by $P/_a^p$ are concerned, we distinguish the following cases:

    1. if P enables both some generative action and some reactive action a_*, then $P/_a^p$ chooses a generative action τ (obtained by hiding an action a_* of P) with probability p and a generative action previously enabled in P with probability $1-p$. Such a rule guarantees that the hiding operator does not introduce nondeterminism among generative actions by preserving the requirement 2 of definition 3.1;
    2. if either P does not enable generative actions, or P does not enable reactive actions a_*, then in $P/_a^p$ parameter p is not considered;

- as far as reactive actions are concerned, $P/^{p}_{a}$ enables the same reactive actions (with the same probability distribution) of type $b \neq a$ enabled in P.

Example 3.3 Let us consider the process $P \stackrel{\Delta}{=} a_* +^{q'} (b +^q c)$, where the choice among a_* and the generative actions b and c is purely nondeterministic (parameter q' is not considered). The semantics of $P/^{p}_{a} \stackrel{\text{iso}}{=} \tau +^p (b +^q c)$, where the action of type a of P is turned into a τ action, is a probabilistic choice between τ, executed with probability p, and the actions b and c, executed with probability $(1-p) \cdot q$ and $(1-p) \cdot (1-q)$, respectively. Hence, parameter p expresses the probability that action τ obtained by hiding the reactive action a_* of P is executed with respect to the generative actions previously enabled by P. ■

The reason for turning reactive actions a_* into generative internal actions τ depends on the fact that one of the goals of the hiding operator consists of obtaining fully specified systems from *open systems* (i.e. systems enabling reactive choices). In particular, when a system is fully specified, the nondeterministic choices due to the possible interactions with the environment have to be resolved, and parameter p turns such choices into probabilistic choices. Formally, the effect of hiding the reactive action a_* in process P corresponds to the execution of a synchronization between a_* and an external generative action a that gives rise to an internal action τ whose probability distribution depends on parameter p. When analyzing security properties which employ the hiding operator, we will show that the low behavior of a secure system does not depend on the choice of parameter p.

Constants A are used to specify recursive systems. When defining an algebraic specification, we assume a set of constants defining equations of the form $A \stackrel{\Delta}{=} P$ (with P a guarded term) to be given.

For the sake of simplicity, we assume:

- the parameter p to be equal to $\frac{1}{2}$ whenever it is omitted from any probabilistic operator;
- the synchronization set S to be the empty set whenever it is omitted from the parallel operator;
- the abbreviation $P/^{p_1 \dots p_n}_{a_1 \dots a_n}$ to denote the expression $P/^{p_1}_{a_1} \dots /^{p_n}_{a_n}$ which hides actions of several types;
- the abbreviation P/S, with $a_1 \dots a_k$ a sequence of types in $AType - \{\tau\}$ and $S = \{a_1, \dots, a_k\}$ the related set of types, to stand for $P/_{a_1 \dots a_k}$ when each term in $Der(P)$ does not enable reactive actions of type in S. It is easy to see that, in such a case, the probabilistic parameters of the hiding operators are not meaningful and, in particular, $P/_{a_1 \dots a_k} \stackrel{\text{iso}}{=} P/^{p_1 \dots p_k}_{a_1 \dots a_k}$, $\forall p_1, \dots, p_k \in]0, 1[$.

As in the nondeterministic case, we employ the parallel operator to express the restriction of actions. Hence, we use the abbreviation $P \backslash L$ to stand for the process $P \,\|_L\, \underline{0}$ denoting the restriction of the actions with type in the set L. Since the null term cannot perform any move, the probabilistic parameter in the parallel operator is not considered, i.e. $P \,\|_L\, \underline{0} \stackrel{\text{iso}}{=} P \,\|^p_L\, \underline{0} \;\forall p \in]0,1[$.

In order to avoid ambiguities, we introduce the following operator precedence relation: prefix $>$ restriction $=$ hiding $>$ alternative composition $>$ parallel composition. Finally, we point out that parameter p of the probabilistic operators cannot be equal to the limiting values 0 and 1, since in our calculus a probabilistic choice does not exclude the execution of a possible behavior of the system (instead, extremal probabilities can be used to express priorities [57]).

3.3 Operational Semantics

The formal semantics of our calculus is given by the generative-reactive transition system $(\mathcal{G}, AType, T)$, whose states are terms of the calculus and the transition relation T is the least multiset satisfying the operational rules reported in Table 2 and in Table 3, where in addition to rules which refer to the local moves of the lefthand process P, we also consider the symmetrical rules taking into account the local moves of the righthand process Q, obtained by exchanging the roles of terms P and Q in the premises and by replacing p with $1-p$ in the label of the derived transitions.

As far as the notation is concerned, we use $P \xrightarrow{\pi} P'$ to stand for $\exists\, p : P \xrightarrow{\pi, p} P'$, meaning that P can execute action π with probability p and then behave as P', $P \xrightarrow{\pi}$ to stand for $\exists\, P' : P \xrightarrow{\pi} P'$, and $P \xrightarrow{G}$ to stand for $\exists\, a \in G \subseteq AType : P \xrightarrow{a}$, meaning that P can execute a generative action of a type belonging to set G. Here, we detail the rules of Table 3 concerning the generative transitions executable by $P \,\|^p_S\, Q$, while it is easy to see that all other rules reflect the informal description of the operators given in Sect. 3.2.

Since we consider a restricted set of executable actions, like in [30] we redistribute the probabilities of the generative transitions of P executable by $P \,\|^p_S\, Q$ so that their overall probability sums up to 1 (we can symmetrically argue for Q). To this aim, in semantics rules we employ the function $\nu_P(G) : \mathcal{P}(AType) \longrightarrow]0,1]$, with $P \in \mathcal{G}$, defined as $\nu_P(G) = \sum\{\!|\, p \mid \exists P',\, a \in G : P \xrightarrow{a, p} P' \,|\!\}$ that computes the sum of the probabilities of the generative transitions executable by P whose type belongs to set G. Note that the set of types of the generative transitions of P executable by $P \,\|^p_S\, Q$ is composed by the action types not belonging to the synchronization set S and the action types belonging to S for which a synchronization between a generative action of P and a reactive action of Q is allowed to be performed. Therefore, in $\nu_P(G)$ if we take as set G the set $G_{S,Q} \subseteq AType$, with $S \subseteq AType - \{\tau\}$ and $Q \in \mathcal{G}$, to be defined as $G_{S,Q} = \{a \in AType \mid a \notin S \vee (a \in S \wedge Q \xrightarrow{a_*})\}$, then we have that $\nu_P(G_{S,Q})$

computes the aggregate probability of the generative transitions of P executable by $P \,\|^p_S\, Q$ and can be used to normalize the probabilities of the generative transitions of P. Moreover, note that the probability of the generative transitions a (with $a \in S$) of P are further distributed among the reactive transitions of type a of Q. Finally, parameter p is used to redistribute the probabilities of the normalized generative transitions of P and Q executable by $P \,\|^p_S\, Q$.

As shown in [2, 10] we have that the operational semantics of a process $P \in \mathcal{G}$ is a *GRTS*.

$$\pi.P \xrightarrow{\pi,1} P$$

$$\frac{P \xrightarrow{a_*,q} P' \quad Q \xrightarrow{a_*}}{P +^p Q \xrightarrow{a_*,p\cdot q} P'} \qquad \frac{P \xrightarrow{a_*,q} P' \quad Q \not\xrightarrow{a_*}}{P +^p Q \xrightarrow{a_*,q} P'}$$

$$\frac{P \xrightarrow{a,q} P' \quad Q \xrightarrow{GAct}}{P +^p Q \xrightarrow{a,p\cdot q} P'} \qquad \frac{P \xrightarrow{a,q} P' \quad Q \not\xrightarrow{GAct}}{P +^p Q \xrightarrow{a,q} P'}$$

$$\frac{P \xrightarrow{a_*,q} P' \quad P \xrightarrow{GAct}}{P/^p_a \xrightarrow{\tau,p\cdot q} P'/^p_a} \qquad \frac{P \xrightarrow{a_*,q} P' \quad P \not\xrightarrow{GAct}}{P/^p_a \xrightarrow{\tau,q} P'/^p_a}$$

$$\frac{P \xrightarrow{b_*,q} P'}{P/^p_a \xrightarrow{b_*,q} P'/^p_a} \quad a \neq b$$

$$\frac{P \xrightarrow{b,q} P' \quad P \xrightarrow{a_*}}{P/^p_a \xrightarrow{b,(1-p)\cdot q} P'/^p_a} \quad a \neq b \qquad \frac{P \xrightarrow{a,q} P' \quad P \xrightarrow{a_*}}{P/^p_a \xrightarrow{\tau,(1-p)\cdot q} P'/^p_a}$$

$$\frac{P \xrightarrow{b,q} P' \quad P \not\xrightarrow{a_*}}{P/^p_a \xrightarrow{b,q} P'/^p_a} \quad a \neq b \qquad \frac{P \xrightarrow{a,q} P' \quad P \not\xrightarrow{a_*}}{P/^p_a \xrightarrow{\tau,q} P'/^p_a}$$

$$\frac{P \xrightarrow{\pi,q} P'}{A \xrightarrow{\pi,q} P'} \quad \text{if } A \triangleq P$$

Table 2: Operational semantics (part I)

$$\frac{P \xrightarrow{a_*,q} P' \quad Q \xrightarrow{a_*}}{P \parallel^p_S Q \xrightarrow{a_*,p\cdot q} P' \parallel^p_S Q} \; a \notin S \qquad \frac{P \xrightarrow{a_*,q} P' \quad Q \not\xrightarrow{a_*}}{P \parallel^p_S Q \xrightarrow{a_*,q} P' \parallel^p_S Q} \; a \notin S$$

$$\frac{P \xrightarrow{a_*,q} P' \quad Q \xrightarrow{a_*,q'} Q'}{P \parallel^p_S Q \xrightarrow{a_*,q\cdot q'} P' \parallel^p_S Q'} \; a \in S$$

$$\frac{P \xrightarrow{a,q} P' \quad Q \xrightarrow{G_{S,P}}}{P \parallel^p_S Q \xrightarrow{a,p\cdot q/\nu_P(G_{S,Q})} P' \parallel^p_S Q} \; a \notin S$$

$$\frac{P \xrightarrow{a,q} P' \quad Q \not\xrightarrow{G_{S,P}}}{P \parallel^p_S Q \xrightarrow{a,q/\nu_P(G_{S,Q})} P' \parallel^p_S Q} \; a \notin S$$

$$\frac{P \xrightarrow{a,q} P' \quad Q \xrightarrow{a_*,q'} Q' \quad Q \xrightarrow{G_{S,P}}}{P \parallel^p_S Q \xrightarrow{a,p\cdot q'\cdot q/\nu_P(G_{S,Q})} P' \parallel^p_S Q'} \; a \in S$$

$$\frac{P \xrightarrow{a,q} P' \quad Q \xrightarrow{a_*,q'} Q' \quad Q \not\xrightarrow{G_{S,P}}}{P \parallel^p_S Q \xrightarrow{a,q'\cdot q/\nu_P(G_{S,Q})} P' \parallel^p_S Q'} \; a \in S$$

Table 3: Operational semantics (part II)

4 Equivalence

In this section we define the weak probabilistic bisimulation equivalence, proposed in [7], in the context of generative-reactive transition systems. The basic idea consists of replacing the classical weak transitions $s \stackrel{\hat{\pi}}{\Longrightarrow} s'$ used in Definition 2.1 by the probability to reach the class of states equivalent to s' from s via a sequence of transitions labeled with strings in $\tau^*\hat{\pi}\tau^*$. Before introducing such a notion of equivalence, we recall some basic notation related to probability theory (see, e.g., [34]).

Probability theory begins with the concept of probability space, which is a triple $(\Omega, \mathcal{F}, P)$, where:

- the *sample space* Ω is a nonempty set of elements, called *sample points*;
- the *sigma field* $\mathcal{F}$ is a nonempty collection of subsets of Ω, called *events*,

which is closed under complementation and countable union;

- the *probability measure* P is an assignment of a real number $P(F)$ to every event F of the sigma field such that (i) $P(F) \geq 0$ for all $F \in \mathcal{F}$, (ii) $P(\Omega) = 1$, (iii) if F_i (with $i = 1, 2, \ldots, n$) are disjoint, then $P(\bigcup_{i=1}^{n} F_i) = \sum_{i=1}^{n} P(F_i)$, (iv) if F_i (with $i = 1, 2, \ldots$) are disjoint, then $P(\bigcup_{i=1}^{\infty} F_i) = \sum_{i=1}^{\infty} P(F_i)$.

Now, let us consider the rooted *GRTS* $(S, AType, T, s_0)$. We define function $Pr : S \times Act \times S \rightarrow [0,1]$ by $Pr(s, \pi, s') = \sum\{\mid p \mid (s, \pi, p, s') \in T \mid\}$ and, for $C \subseteq S$, $Pr(s, \pi, C) = \sum_{s' \in C} Pr(s, \pi, s')$. An execution fragment is a finite sequence $\sigma = s_0 \xrightarrow{\pi_1, p_1} s_1 \xrightarrow{\pi_2, p_2} \ldots s_k$, such that $(s_{i-1}, \pi_i, p_i, s_i) \in T$ for each $i \in \{1 \ldots k\}$. We denote by $tr(\sigma)$ the sequence $\pi_1, \ldots, \pi_k$ and we say that σ is *maximal* if and only if the last state s_k of σ is terminal. Moreover, we denote by $Pr(\sigma) = Pr(s_0, \pi_1, s_1) \cdot \ldots \cdot Pr(s_{k-1}, \pi_k, s_k)$ the probability of the execution fragment σ. An execution is either a maximal execution fragment or an infinite sequence $\sigma' = s_0 \xrightarrow{\pi_1, p_1} s_1 \xrightarrow{\pi_2, p_2} \ldots$ where $(s_{i-1}, \pi_i, p_i, s_i) \in T$ for each $i \in \mathbb{N}$. In particular, we denote by $tr_k(\sigma')$ the sequence $\pi_1, \ldots, \pi_k$. Finally, let $\sigma \uparrow$ denote the set of executions $\{\mid \sigma' \mid \sigma \leq_{prefix} \sigma' \mid\}$ where $\leq_{prefix}$ is the usual prefix relation on sequences.

Now, given an initial state $s \in S$, we define a probability space on the executions which start in s. Let $Exec(s)$ be the set of executions starting in s and $ExecFrag(s)$ be the set of execution fragments starting in s. We denote by $\Sigma(s)$ the smallest sigma field on $Exec(s)$ which contains all $\sigma \uparrow$ where $\sigma \in ExecFrag(s)$. The probability measure *Prob* is the unique measure on $\Sigma(s)$ such that $Prob(\sigma \uparrow) = Pr(\sigma)$.

By assuming $C \subseteq \mathcal{G}$, $P \in \mathcal{G}$, and $A \subseteq Act^*$, we denote by $Exec(A, C)$ the set of executions that lead to a state in C from their initial state through a sequence in A, i.e. $\sigma' \in Exec(A, C)$, with $\sigma' = s_0 \xrightarrow{\pi_1, p_1} s_1 \xrightarrow{\pi_2, p_2} \ldots$, if and only if there exists k such that $tr_k(\sigma')$ belongs to the set of sequences in A and $s_k \in C$. Let $Exec(P, A, C) \stackrel{\text{def}}{=} Exec(A, C) \cap Exec(P)$. It is easy to see that $Exec(P, A, C)$ is measurable in $\Sigma(P)$, as $Exec(P, A, C) = \bigcup_{\sigma} \sigma \uparrow$, where σ ranges over all execution fragments starting in P whose last state is in C and $tr(\sigma)$ is a sequence in A.

In the following, $\tau^* \hat{a}$ stands for the set of sequences $\tau^* a$ if $a \in GAct - \{\tau\}$ and for the set of sequences τ^* if $a = \tau$.

Theorem 4.1 Denoted with $Prob(P, \tau^* \hat{a}, C)$ the probability of going from P to a term in C via sequences of the form $\tau^* \hat{a}$, the equation $Prob(P, \tau^* \hat{a}, C) =$

$$\begin{cases} 1 & \text{if } a = \tau \wedge P \in C \\ \sum_{Q \in \mathcal{G}} Prob(P, \tau, Q) \cdot Prob(Q, \tau^*, C) & \text{if } a = \tau \wedge P \notin C \\ \sum_{Q \in \mathcal{G}} Prob(P, \tau, Q) \cdot Prob(Q, \tau^* a, C) + Prob(P, a, C) & \text{if } a \neq \tau \end{cases}$$

has a least solution and is exactly $Prob(Exec(P, \tau^* \hat{a}, C))$.

The definition of weak probabilistic bisimulation for *GRTS*s derives from the definition of weak probabilistic bisimulation for fully probabilistic systems of [7], where the weak transition relation $\stackrel{\hat{a}}{\Rightarrow}$ of Milner [44] is replaced by the probability $Prob(P, \tau^*\hat{a}, C)$. It is worth noting that the authors of [7] define both weak and delay bisimulation [44] for fully probabilistic transition systems and show that these two relations coincide in the probabilistic setting (hence we can use $\tau^*\hat{a}$ instead of $\tau^*\hat{a}\tau^*$ in the following definition).

Definition 4.2 An equivalence relation $R \subseteq \mathcal{G} \times \mathcal{G}$ is a weak probabilistic bisimulation if and only if, whenever $(P, Q) \in R$, then for all $C \in \mathcal{G}/R$:

- $Prob(P, \tau^*\hat{a}, C) = Prob(Q, \tau^*\hat{a}, C)$ for all $a \in GAct$
- $Prob(P, a_*, C) = Prob(Q, a_*, C)$ for all $a_* \in RAct$

Two terms $P, Q \in \mathcal{G}$ are weakly probabilistically bisimulation equivalent, denoted $P \approx_{PB} Q$, if there exists a probabilistic weak bisimulation R containing the pair (P, Q).

Note that since the choice between reactive actions and internal (generative) actions is nondeterministic, then the definition above requires two equivalent terms to be strongly equivalent in case of reactive actions.

Example 4.3 The two terms $P \stackrel{\Delta}{=} (a_*.\underline{0} + a.\underline{0})$ and $Q \stackrel{\Delta}{=} (a_*.\underline{0} + \tau.P)$ are weakly probabilistically bisimulation equivalent. From an external observer standpoint, both terms nondeterministically choose between a reactive action a_* and a generative action a. Let R be the equivalence relation on $S = \{P, Q, \underline{0}\}$ such that $S/R = \{C_1, C_2\}$ with $C_1 = \{P, Q\}$ and $C_2 = \{\underline{0}\}$. Therefore, for each S' in C_1 we have $Prob(S', \tau^*a, C_2) = 1$ and $Prob(S', a_*, C_2) = 1$. Note that, if we erroneously try to relax the strong condition in case of reactive actions, we would obtain $Prob(Q, \tau^*a_*, C_2) = 2$, since Q has at his disposal two possible paths for the execution of a_*, but the choice between these two paths is completely nondeterministic. ■

Example 4.4 Let us consider the two *GRTS*s of Fig. 7, which turn out to be weakly probabilistically bisimulation equivalent. To show such a result, let R be the equivalence relation on $S = \{S_0, S_1, S_2, S'_0, S'_1, S'_2\}$ such that $S/R = \{C_1, C_2\}$ where $C_1 = \{S_0, S'_0\}$ is the equivalence class of the initial states and $C_2 = \{S_1, S_2, S'_1, S'_2\}$ is the equivalence class of the remaining states. For each $S' \in C_1$ we have $Prob(S', \tau^*, C_1) = 1$ and $Prob(S', \tau^*\pi, C_2) = 1/2$ where $\pi \in \{a, b\}$. On the other hand, for each $S' \in C_2$ we have $Prob(S', \tau^*, C_2) = 1$ and $Prob(S', \tau^*\pi, C_1) = 0$ where $\pi \in \{a, b\}$. Hence, R is a weak probabilistic bisimulation. It is worth noting that from state $S'_0 \in C_1$ we reach state $S'_1 \in C_2$ ($S'_2 \in C_2$) via an action a (b) by executing an arbitrary number of times the action τ. In order to evaluate the probability $Prob(S'_0, \tau^*a, C_2)$ ($Prob(S'_0, \tau^*b, C_2)$) we have to redistribute the probability $\frac{1}{3}$ associated with the outgoing internal transition of S_0' among the other outgoing transitions of S_0'. By applying the formula defined in this section, we have $Prob(S_0', \tau^* a, C_2) = \frac{1}{3} \cdot Prob(S_0', \tau^* a, C_2) + \frac{1}{3}$ from which we obtain $Prob(S_0', \tau^* a, C_2) = \frac{1}{2}$ (similarly for b). ■

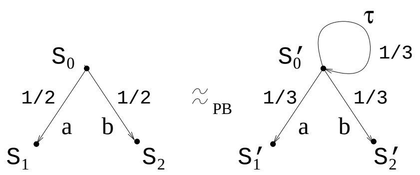


Figure 7: Bisimulation equivalent probabilistic systems

As far as the equivalence check is concerned, [7] describes an algorithm that computes weak probabilistic bisimulation equivalence classes in time $\mathcal{O}(n^3)$ and space $\mathcal{O}(n^2)$, where n is the number of states. We now show that if two terms are weakly probabilistically bisimulation equivalent, then their nondeterministic counterparts are weakly bisimulation equivalent. Formally, the following lemma shows that, given two terms $P, Q \in \mathcal{G}$ such that $P \approx_{PB} Q$, then the two terms $P_{nd}, Q_{nd} \in \mathcal{G}_{nd}$ obtained from P and Q by removing the probabilities from the algebraic operators are weakly bisimulation equivalent.

Lemma 4.5 Given $P, Q \in \mathcal{G}$, $P \approx_{PB} Q \Rightarrow P_{nd} \approx_B Q_{nd}$.

In particular, as we will see in the next section, such a result is used to prove that the security properties defined in our probabilistic setting capture all information flows that arise by analyzing the nondeterministic behavior of a system.

We conclude this section by giving a notion of weak probabilistic bisimulation up to $\approx_{PB}$ in the line of [13], that will be used in the next section in the proof of Theorem 5.11. In the following definition, given a relation R, we denote by R^{-1} the inverse relation of R, and by $(R)^+$ the transitive closure of R.

Definition 4.6 A relation $R \subseteq \mathcal{G} \times \mathcal{G}$ is a weak probabilistic bisimulation up to $\approx_{PB}$ if and only if, whenever $(P, Q) \in R$, then for all $C \in \mathcal{G}/(\approx_{PB} \cup R \cup R^{-1})^+$:

- $Prob(P, \tau^* \hat{a}, C) = Prob(Q, \tau^* \hat{a}, C)$ for all $a \in GAct$
- $Prob(P, a_*, C) = Prob(Q, a_*, C)$ for all $a_* \in RAct$.

5 Security Properties for Probabilistic Processes

In this section, we present three probabilistic security properties, by extending the non-interference theory described in Sect. 2 to the probabilistic framework of Sect. 3 and 4. First, we need to formally characterize the set of processes which we consider for the definition of the probabilistic properties. In particular, we want to exclude those transition systems where the probability of observing a visible event a (preceded by a sequence of invisible actions) may tend to 0. This is possible, e.g., if the transition system enables execution fragments σ of the form $\tau^* a$, where the sequence of visited states is acyclic and $Pr(\sigma)$ tends to 0 as the length of the sequence τ^* tends to ∞. If this is the case, we cannot quantify neither the probability of observing the visible event a, since its execution probability may tend to 0, nor the effect of such a potential event on the (low-level) observable behavior of the system. In order to avoid such a situation and to simplify the technical machinery needed in the rest of the paper, we restrict ourselves to finite state processes. Extensions which include infinite processes are possible, but would complicate our definitions and theorems. In the following, we denote by $\mathcal{G}^f$ the set of finite state processes and, by taking into consideration such a family, we introduce the probabilistic version of the non-interference properties of Sect. 2.2. Then, through several examples, we discuss such properties by emphasizing their adequacy to reveal indirect information flows that are not captured by classical properties based on nondeterminism. Afterwards, we show that our probabilistic approach is a conservative extension of the nondeterministic approach to information flow theory described in Sect. 2. Finally, we show that by modeling the probabilistic behavior of the system a quantitative estimate of the information flow can be given, in the sense that we can evaluate the probability of observing each (potentially insecure) behavior.

5.1 Probabilistic Non-interference

We start our analysis by defining a probabilistic extension of the *BSNNI* property, that we call Bisimulation Strong Probabilistic Non-interference (*BSPNI*). In the following, given $P \in \mathcal{G}^f$, we denote by $H_P \subseteq AType_H$ the set containing the types of the high-level actions which syntactically occur in the action prefix operators within P, by h_P the sequence (in alphabetic order) of the types in H_P, and by $|H_P|$ the cardinality of H_P. Moreover, we denote by Seq_D^k the set of k-length sequences with domain D.

Definition 5.1 $P \in \mathcal{G}^f$ is *BSPNI* secure if and only if

$$P/^{\mathbf{p}}_{h_P} \approx_{PB} P \backslash AType_H \ \forall \mathbf{p} \in Seq^{|H_P|}_{]0,1[}.$$

The intuition under such a definition is that a system P is secure if and only if the probability distribution of the events which are observable by a low user is independent of the behavior of the high user. In other words, the *BSPNI* property extends the condition checked by the *BSNNI* property by taking into consideration also the probabilistic information. Note that in term $P/^{\mathbf{p}}_{h_P}$, the parameters $p_1 \ldots p_{|H_P|}$ forming the sequence $\mathbf{p}$ are used to solve the nondeterminism in P due to the potential interaction of any reactive high-level action of P with the external environment. In practice, they model the probabilistic behavior of the high user that executes generative high-level actions which synchronize with the corresponding reactive high-level actions of P to form internal actions τ. One can think that the use of the parameters p_i when hiding the system can be debatable, since it is not clear if the choice of such parameters can affect the result of the analysis. Therefore, we want the *BSPNI* property to be satisfied for any choice of such parameters, i.e. system P is to be secure for any probability distribution (chosen by the high user) of the hidden reactive high-level actions. This means that we check if in any way the nondeterministic choices in P are resolved by the high user the probabilistic behavior of P as observed by a low user does not change.

A sufficient condition which permits to verify the *BSPNI* property in spite of the universal quantification consists of checking (when we compute weak bisimulation equivalence classes of $P/^{\mathbf{p}}_{h_P}$) if the class partitioning we obtain depends on some of the values p_i forming the sequence $\mathbf{p}$. If this is not the case, i.e. if the probability associated with any transition leading from one class to another one does not depend on any parameter p_i, and if $P/^{\mathbf{p}}_{h_P}$ is equivalent to $P \backslash \mathit{AType}_H$, then we have that the high probabilistic behavior does not affect the low view of the system. On the other hand, if some p_i affects the probability of going from one class to another one, then $P \backslash \mathit{AType}_H$ cannot be equivalent to $P/^{\mathbf{p}}_{h_P}$ $\forall \mathbf{p} \in \mathit{Seq}^{|H_P|}_{]0,1[}$.

Example 5.2 As an example of a secure system, let us consider term $P \stackrel{\Delta}{=} l.P +^q h.l.P$, representing a process that can perform a low-level action preceded (or not) by a high-level action. The high level does not interfere with the low level, because, from the low user standpoint, the action l is executed with probability 1 with no information about the high behavior. Indeed, we have $P/_h \stackrel{\mathrm{iso}}{=} l.P +^q \tau.l.P \approx_{PB} l.P \stackrel{\mathrm{iso}}{=} P \backslash \{h\}$ (note that the probabilistic parameter in $P/_h$ is not meaningful as we hide generative actions only). We point out that the same example obtained by replacing h with h_*, i.e. if we consider term $P' \stackrel{\Delta}{=} l.P' + h_*.l.P'$, is still secure, as $P'/^p_h \stackrel{\mathrm{iso}}{=} l.P' +^{1-p} \tau.l.P' \approx_{PB} l.P' \stackrel{\mathrm{iso}}{=} P' \backslash \{h\}$ for any choice of parameter p. This means that any potential interaction of P' with the environment by means of a synchronization via a high-level action h cannot alter the probability distribution of the event l (which is always executed with probability 1) and, as a consequence, the behavior of the system observable by a low user. Therefore, not depending on how the nondeterministic choice between l and h_* will be resolved by the high environment, the system remains secure and does not reveal any high information to the low view. ■

5.1.1 Capturing Probabilistic Information Flows

In this subsection, we study the kind of information leakage that can be ruled out by the *BSPNI* property by employing the probabilistic information, and the relation with the purely possibilistic framework.

Example 5.3 Let us consider the system $P_{nd} \stackrel{\Delta}{=} l.\underline{0} + l.l'.\underline{0} + l.h.l'.\underline{0}$ which represents the behavior of a nondeterministic process that allows a low user to observe either a single event l or the sequence of events $l.l'$. P_{nd} is *BSNNI* secure, since $P_{nd}/AType_H \stackrel{\text{iso}}{=} l.\underline{0} + l.l'.\underline{0} + l.\tau.l'.\underline{0} \approx_B l.\underline{0} + l.l'.\underline{0} \approx_B l.\underline{0} + l.l'.\underline{0} + l.\underline{0} \stackrel{\text{iso}}{=} P_{nd}\backslash AType_H$. However, process P_{nd} has an undesirable information flow that is captured by considering a property stronger than the *BSNNI*. For instance, by employing the *SBNDC* property we can observe that after the occurrence of the action l leading to state $h.l'.\underline{0}$, the high user can delay the execution of the action l', thus affecting the low view of the system. Formally, $P \xrightarrow{l} h.l'.\underline{0} \xrightarrow{h} l'.\underline{0}$, and $(h.l'.\underline{0})\backslash AType_H \not\approx_B (l'.\underline{0})\backslash AType_H$.

Now, let us detail P_{nd} with probabilistic information, by obtaining the system $P \stackrel{\Delta}{=} (l.\underline{0} +^p l.l'.\underline{0}) +^q l.h.l'.\underline{0}$ (see the corresponding *GRTS* in Fig. 8). As we have seen, such a system has no information flow if we only consider the possible behaviors of the system and the *BSNNI* property, but the execution of the sequence $l.l'$ (with respect to the execution of a single l) will reveal information about h when considering statistical inferences derived from the relative frequency of outcomes of repeated executions. In other words, the high user affects the low view as observed by a low user that considers the execution frequency of the low events in several runs of the system. Formally, in $P\backslash AType_H$ a low user observes either the single event l with probability $q \cdot p + (1-q)$ or the sequence $l.l'$ with probability $q \cdot (1-p)$. On the other hand, in $P/AType_H$ a low user observes either the single event l with probability $q \cdot p$ or the sequence $l.l'$ with probability $1 - q \cdot p$. As a consequence $P\backslash AType_H \not\approx_{PB} P/AType_H$ and, therefore, P is not *BSPNI* secure. ■

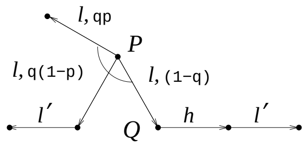


Figure 8: Example of probabilistic information flow

Example 5.4 Let us consider the nondeterministic process:

$$
\begin{aligned}
P_{nd} &\stackrel{\Delta}{=} \tau.Q_{nd} + \tau.l_2.\underline{0} \\
Q_{nd} &\stackrel{\Delta}{=} \tau.l_1.\underline{0} + h_*.l_2.\underline{0}
\end{aligned}
$$

which is *BSNNI* secure. Indeed, by the τ-laws of [44] we have $P_{nd}\backslash AType_H \stackrel{\text{iso}}{=} \tau.\tau.l_1.\underline{0}+\tau.l_2.\underline{0} \approx_B \tau.l_1.\underline{0}+\tau.l_2.\underline{0}$ and $P_{nd}/AType_H \stackrel{\text{iso}}{=} \tau.(\tau.l_1.\underline{0}+\tau.l_2.\underline{0})+\tau.l_2.\underline{0} \approx_B \tau.(\tau.l_1.\underline{0} + \tau.l_2.\underline{0})$, from which we derive $P_{nd}\backslash AType_H \approx_B P_{nd}/AType_H$. Moreover, P_{nd} satisfies also the *BNDC* property. Intuitively, the low view of P is given by a nondeterministic choice between the actions l_1 and l_2. The system may be considered insecure because whether the state Q_{nd} is reached, a high user can determine, by communicating an action h via a synchronization with h_*, whether the low user will observe the action l_1 or l_2. Therefore, a low user which observes an action l_1 can deduce that no high-level event has taken place. Such a behavior is captured by the *SBNDC* property.

Formally, $Q_{nd} \xrightarrow{h_*} l_2.\underline{0}$ and $l_2.\underline{0}\backslash AType_H \not\approx_B \tau.l_1.\underline{0} \stackrel{\text{iso}}{=} Q_{nd}\backslash AType_H$. Now, let us consider the following probabilistic extension of process P_{nd}:

$$\begin{array}{rcl} P & \stackrel{\Delta}{=} & \tau.Q +^{\frac{1}{2}} \tau.l_2.\underline{0} \\ Q & \stackrel{\Delta}{=} & \tau.l_1.\underline{0} + h_*.l_2.\underline{0} \end{array}$$

that gives rise to the *GRTS* depicted in Fig. 9. In our probabilistic setting the low-level view of the system is given by a probabilistic choice between the actions l_1 and l_2. If state Q is reached, a high user can interfere with the system by resolving the nondeterministic choice (between the internal action τ and the reactive action h_*) in such a way that the probability of observing an action l_1 with respect to an action l_2 changes, thus altering the low view of the system when considering the relative frequency of outcomes of repeated executions. More formally, P is not *BSPNI* secure, because $P\backslash AType_H \stackrel{\text{iso}}{=} \tau.\tau.l_1.\underline{0} +^{\frac{1}{2}} \tau.l_2.\underline{0} \not\approx_{PB} \tau.(\tau.l_1.\underline{0} +^{1-q} \tau.l_2.\underline{0}) +^{\frac{1}{2}} \tau.l_2.\underline{0} \stackrel{\text{iso}}{=} P/^q_h$ for any choice of parameter $q \in]0, 1[$. In particular, in $P\backslash AType_H$ a low user observes the action l_1 with probability $\frac{1}{2}$, while in $P/^q_h$ he observes the same action l_1 with probability $(1 - q) \cdot \frac{1}{2}$. ■

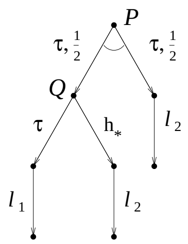


Figure 9: The nondeterministic version of P is *BSNNI* and *BNDC*, while P is not *BSPNI*

The two examples above show that from a nondeterministic process P_{nd} which is *BSNNI* secure, we can derive a probabilistic process P obtained by adding probabilistic information to the system, such that P is not *BSPNI* secure because it reveals a probabilistic covert channel. On the other hand, in the following example we show that the *BSPNI* property is more than enough to capture probabilistic covert channels of systems that, without probabilistic information, are classified as secure systems even by the strongest nondeterministic security properties of Sect. 2.2 (like, e.g., the *SBNDC*).

Example 5.5 Let us consider the following pseudocode which is inspired by an example proposed by Sabelfeld and Sands in [51, 52]:

$$l := h \; +^p \; l := rand(1)$$

where h is a high-level variable that has been previously set either to 0 or to 1 by the high user (e.g., according to a given probability distribution), and $rand(n)$ is a function which randomly samples a natural number in the set $\{0, \ldots, n\}$. Then, the low-level output (i.e. the value of variable l) either is directly read by the high-level output with probability p or is randomly sampled from the possible values 0 and 1 with probability $1-p$. The behavior of such a system is graphically represented in Fig. 10. It is easy to see that the above program has no information flow if we only consider the possible behaviors of the system, but the final value of l will reveal h if the low user observes the relative frequency of outcomes of repeated executions.

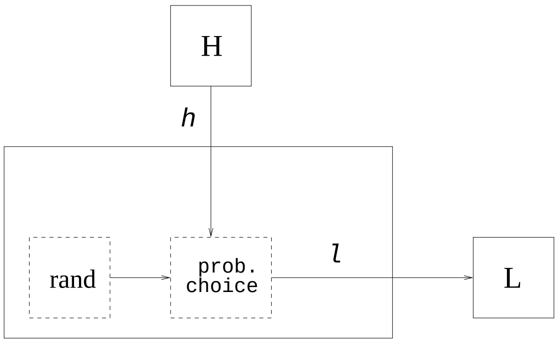


Figure 10: Example of a system concealing a probabilistic covert channel

Similarly, let us consider the following process:

$$P \stackrel{\Delta}{=} (l_0.P +^{\frac{1}{2}} l_1.P) +^p h.(l_0.P +^q l_1.P)$$

where a low user observes either an action l_0 or an action l_1 with a probability distribution that depends on the high environment behavior. In particular, if no external communication can be completed via a synchronization with the high action h, the system chooses between the two low-level actions l_0 and l_1 with the same probability $\frac{1}{2}$ (such a choice models the assignment $l := rand(1)$). On the other hand, if the system interacts with the high user by executing the action h, then the probability distribution of the two low-level events is guided by parameter q (such a component is the counterpart of the assignment $l := h$).

Intuitively, q is the parameter of the probability distribution of the two possible values (0 and 1) that h can assume and transmit to the low part, while p is the parameter which guides the probabilistic choice between the two alternative ways of producing a low-level output. The nondeterministic version of process P is secure, because the high behavior does not alter the low view of the system, that is represented by a nondeterministic choice between the actions l_0 and l_1. However, in the probabilistic setting this is not the case. Indeed, the probability of observing an action l_0 with respect to an action l_1 changes depending on the behavior of the high user. In particular, from $P \backslash AType_H \stackrel{\text{iso}}{=} l_0.P +^{\frac{1}{2}} l_1.P$ and $P / AType_H \stackrel{\text{iso}}{=} (l_0.P +^{\frac{1}{2}} l_1.P) +^p \tau.(l_0.P +^q l_1.P)$ we derive $P \backslash AType_H \not\approx_{PB} P / AType_H$, except for the case $q = \frac{1}{2}$ (see Fig. 11). In practice, we have that P is *BSPNI* secure if and only if $q = \frac{1}{2}$, because in such a case the high user simulates the behavior of the function $rand(1)$ used to determine the low output, so that a low user cannot infer the high behavior by observing the probability distribution of the two low-level actions. ■

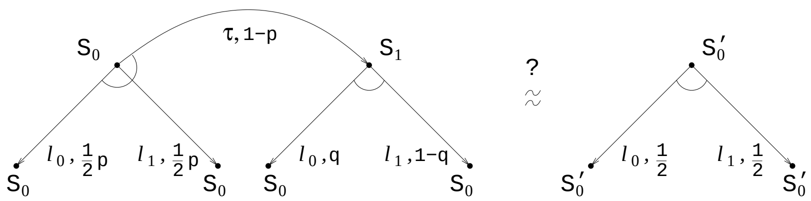


Figure 11: $S_0 \approx_B S_1 \approx_B S_0'$, but $S_0 \not\approx_{PB} S_1 \not\approx_{PB} S_0'$, except for the case $q = \frac{1}{2}$.

The information flow in Example 5.5 is not captured by any security property defined in [25], since the potential interaction between the system and the high user does not change the possibilistic behavior that is observable by a low user. In other words, the covert channel of Example 5.5 is merely probabilistic, in that it is revealed by observing the low-level probabilistic behavior only.

5.2 Probabilistic Nondeducibility On Composition

Sometimes the non-interference property is not enough to capture all the potential insecure behaviors of a system. For this reason, other properties have been suggested in order to overcome the lacks of such a property. Among the different proposals, here we consider the *BNDC* that, in our probabilistic setting, we call Probabilistic *BNDC* (*PBNDC*, for short).

Definition 5.6 $P \in \mathcal{G}^f$ is *PBNDC* secure if and only if

$P \backslash AType_H \approx_{PB} ((P \,\|^p_{\{h_1,\ldots,h_k\}}\, \Pi)/^{\mathbf{p}}_{h_1\ldots h_k}) \backslash AType_H$, for each sequence $h_1 \ldots h_k$ of high-level types, $\forall \Pi \in \mathcal{G}^f_H$, $\forall p \in]0,1[$ and $\forall \mathbf{p} \in Seq^k_{]0,1[}$.

Such a definition says that the probabilistic low view of P is invariant with respect to the execution of P in parallel with any high-level process Π. The *PBNDC* does not depend on the probability distribution of the hidden high-level actions which derive from the synchronizations between P and Π, because it is to be satisfied:

- for any choice of process Π which guides the probability distribution of the generative high-level actions which are offered by the high user and synchronize with the corresponding reactive high-level actions of P,
- for any choice of parameters $p_1, \ldots, p_k$ forming the sequence $\mathbf{p}$, which are chosen by the high user to solve the nondeterminism due to the reactive high-level actions executable by $P \,||^{p}_{\{h_1,\ldots,h_k\}}\, \Pi$ (deriving from synchronizations among reactive high-level actions of the same type of P and Π) that are turned into internal τ actions,
- for any choice of parameter p, which guides the probabilistic parallel execution of processes P and Π.

The *PBNDC* property expresses an intuition similar as that underlying the notion of probabilistic non-interference given by Gray in [32]. In particular, Gray defines the notions of *high environment behavior* $\mathcal{H}$ and *low environment behavior* $\mathcal{L}$, which give the probability of the external environment producing high (low) inputs given that the previous history of high (low) input and output events has taken place. Then, he checks if, by fixing $\mathcal{L}$ and by modifying $\mathcal{H}$, the probability distribution of the low-level events of the system changes or not depending on the environment behavior described by $\mathcal{L}$ and $\mathcal{H}$. In our process-algebraic framework, $\mathcal{L}$ is fixed (note that P is constrained to synchronize on high-level actions only, meaning that the low environment behavior does not alter the probability distribution of the low-level actions of P), while $\mathcal{H}$ is explicitly modeled by the high-level process Π, that interacts with the reactive high-level (input) actions and the generative high-level (output) actions of P. Then, we check if any high user is able to affect the probabilistic low view of P, or, by using the terminology borrowed from Gray, we check if the high environment behavior alters the probability distribution of the low-level events.

As in the nondeterministic framework, *PBNDC* is at least as strong as *BSPNI*.

Proposition 5.7 *PBNDC* $\subset$ *BSPNI*.

The example reported in the proof of Proposition 5.7 ($P \triangleq l.\underline{0} +^{p} h.h.l.\underline{0}$) shows that the *BSPNI* property is not able to detect some potential deadlock due to high-level activities, exactly as shown in [24] in the nondeterministic case. Similarly, the next example reveals that the *PBNDC* is more adequate than the *BSPNI* to capture probabilistic covert channels.

Example 5.8 Let us consider the process:

$$P \stackrel{\Delta}{=} (\tau.(l.\underline{0} +^p h_1.\underline{0}) +^{1-p} \tau.(l.\underline{0} +^p \tau.\underline{0})) +^p h_2.l.\underline{0},$$

whose corresponding *GRTS* is shown in Fig. 12(a). It is provable that $P \in$ *BSPNI*. Indeed, we have that $P/ATypе_H \stackrel{\text{iso}}{=} (\tau.(l.\underline{0}+^p\tau.\underline{0})+^{1-p}\tau.(l.\underline{0}+^p\tau.\underline{0}))+^p \tau.l.\underline{0} \approx_{PB} \tau.(l.\underline{0} +^p \tau.\underline{0}) +^p \tau.l.\underline{0} \approx_{PB} \tau.l.\underline{0} +^{1-p} \tau.(l.\underline{0} +^p \tau.\underline{0}) \approx_{PB} (\tau.(l.\underline{0} +^p h_1.\underline{0}) +^{1-p} \tau.(l.\underline{0} +^p \tau.\underline{0}))\backslash AType_H \approx_{PB} ((\tau.(l.\underline{0} +^p h_1.\underline{0}) +^{1-p} \tau.(l.\underline{0} +^p \tau.\underline{0})) +^p h_2.l.\underline{0})\backslash AType_H \stackrel{\text{iso}}{=} P\backslash AType_H$.

Intuitively, it is easy to see that if a low user observes the action l, then he knows that the high-level action h_1 has not taken place. More precisely, to show the potential covert channel which may be set up from high level to low level, we observe that a high user may (i) prevent the execution of action h_2 of the component $h_2.l.\underline{0}$, (ii) wait for the probabilistic choice between the two internal actions τ, and then (iii) if the first component $l.\underline{0}+^p h_1.\underline{0}$ is chosen with probability $1-p$, he can affect the low view of the system by communicating (or not) the action h_1. We can prove that this insecure behavior is captured by the *PBNDC* property. To this end, consider the process $\Pi \stackrel{\Delta}{=} h_{1_*}.\underline{0}$ and the synchronization set $S = \{h_1, h_2\}$. The *GRTS* derived from $((P \parallel_S \Pi)/S)\backslash AType_H$ is depicted in Fig. 12(b). In particular, the behavior of such a composed system consists of a probabilistic choice between the action l, performed with probability p, and the internal action τ, performed with probability $1-p$. The *GRTS* derived from $P/AType_H$ is that of Fig. 12(c), which is clearly not equivalent to that of Fig. 12(b). Therefore, $((P \parallel_S \Pi)/S)\backslash AType_H \not\approx_{PB} P/AType_H$ and, as a consequence, P is not *PBNDC* secure. ■

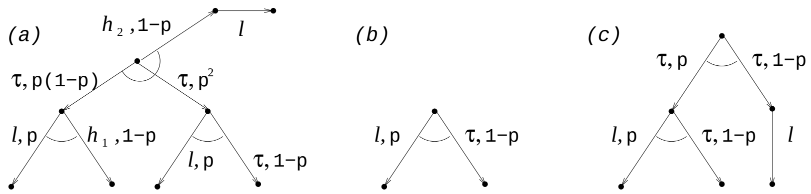


Figure 12: Example of a system that is *BSPNI* secure but not *PBNDC* secure

Finally, it is worth noting that, similarly as reported in [24, 26], the above definition of *PBNDC* is difficult to use because of the universal quantification on high-level processes. For this reason, in the following section we propose the probabilistic version of the *SBNDC* property that solves such a problem.

5.3 Strong Probabilistic *BNDC*

In this section, we introduce a probabilistic security property stronger than the *PBNDC*, which extends to the probabilistic setting the *SBNDC* of Sect. 2.2.

Example 5.9 The process $P \stackrel{\Delta}{=} \tau.(\tau.l_1.\underline{0} +^p h.l_2.\underline{0}) +^p \tau.(h.l_1.\underline{0} +^p \tau.l_2.\underline{0})$ probabilistically evolves either into a state which performs action l_1 with probability 1 or into a state which performs action l_2 with probability 1. The corresponding *GRTS* is shown in Fig. 13. Such a system is *BSPNI* secure, because $P/AType_H \approx_{PB} \tau.l_1.\underline{0} +^p \tau.l_2.\underline{0} \approx_{PB} P \backslash AType_H$. Moreover, it is provable that P is *PBNDC* secure too. As a sketch of such a proof, we observe that any composed term $((P \,\|^q_S\, \Pi)/S) \backslash AType_H$ is weakly probabilistically bisimulation equivalent to:

- $P/AType_H$ in the case $h \in S$ and Π enables h,
- $P \backslash AType_H$ otherwise.

Should we consider P as a secure system? Intuitively, if we assume that the high user is allowed to know whether from the initial state P the system evolves either into P_1 or into P_2 (see Fig. 13), then the system is certainly insecure. Indeed, the relative frequency of the low-level action l_1 (l_2) as observed in a repeated execution of system P in isolation is p $(1-p)$, but such a probability distribution can be easily altered by a high user who, aware of how the internal probabilistic choice in term P is resolved, can decide to block (to communicate) the action h. Such a behavior is not captured by the *PBNDC*, therefore we need a stronger notion of security, and to this end we introduce the strong version of the *PBNDC*. However, we point out that the assumption above on the knowledge of the high user which is necessary to set up the insecure information flow is, in our view, rather questionable. ■

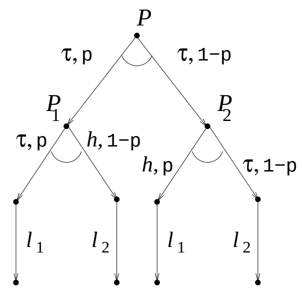


Figure 13: Example of a covert channel in a system that is *PBNDC* secure

Now, we define the probabilistic version of the *SBNDC* that we call Strong *PBNDC* (*SPBNDC*). Such a definition completes the probabilistic extension of the nondeterministic approach presented in Sect. 2.

Definition 5.10 $P \in \mathcal{G}^f$ is *SPBNDC* secure if and only if $\forall P' \in Der(P)$ and $\forall P''$ such that $P' \xrightarrow{\pi,p} P''$, $t(\pi) \in AType_H$, $p \in]0,1]$, then

$$P'\backslash AType_H \approx_{PB} P''\backslash AType_H.$$

In general, by following a basic intuition similar to that given for *AFM*s [32], an interpretation of the *SPBNDC* property is that in each system state the probability of the low events is independent of the previous high events. Indeed, supposed that the probability distribution of a low event l in a given state of a system P depends on a high action h previously executed, we should also have that there exist P' and P'', reachable from P, such that $P' \xrightarrow{h} P''$ and the probability distribution of l in P' is not equal to the probability distribution of l in P'', thus violating the *SPBNDC* property. Therefore, the strong condition verified by the *SPBNDC* is that in any state of the system the low user cannot make inferences about high information previously communicated by (to) the high user. The several examples we reported in the previous sections suggest that, if we are interested in guaranteeing that what the low part can see is independent of what the high part can do, then the *PBNDC* is the appropriate definition with respect to protection against probabilistic covert channels. On the other hand, the *SPBNDC* is easier to be verified than the *PBNDC* and, as we can see in the following Theorem 5.11, is also stronger than the *PBNDC*. Hence, it can be used as a verification condition for *PBNDC*.

Theorem 5.11 *SPBNDC* $\subset$ *PBNDC*.

Here, we just give an intuition of the reason why the inclusion above holds. If a system P is *SPBNDC*, then $P' \xrightarrow{\pi} P''$, with P' and P'' derivatives of P and $t(\pi) \in AType_H$, implies that $P'\backslash AType_H$ and $P''\backslash AType_H$ are weakly probabilistically bisimulation equivalent. Therefore, the probability distribution of the high-level actions executable by $P' \in Der(P)$ does not affect the probability distribution of the low view of the system, or, equivalently, any high-level process interacting with P' cannot alter the probability distribution of the low-level actions as observed by a low user.

From the proof of Theorem 5.11, we derive that if $P \in SPBNDC$ then $\forall P' \in Der(P).\, P'\backslash AType_H \approx_{PB} ((P' \|^{p}_{\{h_1,\ldots,h_k\}} \Pi)/^{\mathbf{q}}_{h_1\ldots h_k})\backslash AType_H$, for each sequence $h_1 \ldots h_k$ of high-level types, $\forall \Pi \in \mathcal{G}^f_H$, $\forall p \in]0,1[$, and $\forall \mathbf{q} \in Seq^k_{]0,1[}$. Therefore, $\forall P' \in Der(P)$ we have that P' is *PBNDC* (and also *BSPNI*), which is a condition stricter than that we wanted to prove.

5.4 Conservative Extension and Comparison

In this section, we show that the security properties defined in our probabilistic framework are the natural, conservative extension of the possibilistic security properties based on the nondeterministic setting described in Sect. 2. Intuitively, the integration of probabilities as a further aspect of the system to be modeled adds new information that extends what is already known in the nondeterministic case. In the previous sections, we have shown through several examples that if a nondeterministic model P_{nd} satisfies a security property, say SP_{nd}, then its probabilistic version P, obtained by modeling also the probabilistic behavior of the system, may not satisfy the security property SP, which is the probabilistic counterpart of SP_{nd}. Now, we want to show that given an arbitrary security property SP which has to be checked for a term $P \in \mathcal{G}^f$, if P satisfies SP then P_{nd} satisfies SP_{nd}. In order to prove such a relation, we now show what happens, from the security property viewpoint, when passing from the probabilistic framework to the nondeterministic one.

The following theorem shows that the probabilistic security properties extend in the expected way the corresponding nondeterministic counterparts, in the sense that if a probabilistic system turns out to be SP–secure, then its nondeterministic version obtained by replacing probabilities by nondeterministic choices is SP_{nd}–secure. We assume $SP \in \{BSPNI, PBNDC, SPBNDC\}$, and we denote by SP_{nd} the nondeterministic counterpart of the probabilistic property SP, i.e. $SP_{nd} \in \{BSNNI, BNDC, SBNDC\}$.

Theorem 5.12 (Conservative Extension)
Given $P \in \mathcal{G}^f$, P is SP secure $\Rightarrow P_{nd}$ is SP_{nd} secure.

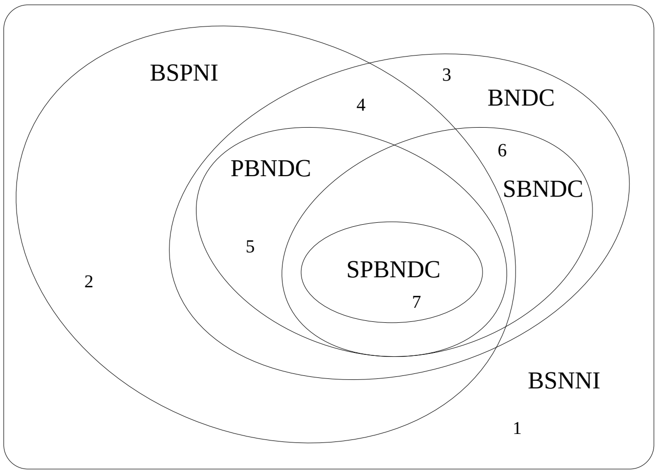
By Proposition 5.7 and Theorem 5.11 it follows that the probabilistic extension of the security properties preserves the same inclusion relationships seen in the nondeterministic case in Proposition 2.10. Further potential relations among nondeterministic and probabilistic security properties are discussed through the following example.

Example 5.13 The nondeterministic system $P_{nd} \stackrel{\Delta}{=} \tau.l.\underline{0} + h_1.l.\underline{0} + \tau.\underline{0} + h_2.\underline{0}$ is $BSNNI$ ($P_{nd}/AType_H \approx_B \tau.l.\underline{0} + \tau.\underline{0} \stackrel{\text{iso}}{=} P_{nd} \backslash AType_H$) and, as it is easy to verify, $BNDC$ secure. However, the execution of the low-level action l reveals that the high user has not performed an action of type h_2. Such an information flow is captured by the $SBNDC$ property only.

The probabilistic extension $P \stackrel{\Delta}{=} (\tau.l.\underline{0} +^p h_1.l.\underline{0}) +^{1-p} (\tau.\underline{0} +^p h_2.\underline{0})$ is $BSPNI$ secure. Indeed, we have $P/AType_H \approx_{PB} \tau.l.\underline{0} +^{1-p} \tau.\underline{0} \approx_{PB} P \backslash AType_H$. For the same reason reported in case of P_{nd}, term P should be considered insecure. With respect to the nondeterministic case, where the $SBNDC$ is needed to reveal the insecure behavior, in our probabilistic setting the $PBNDC$ is enough to capture an information flow in P. Indeed, if we consider the process $\Pi \stackrel{\Delta}{=} h_{2_*}.\underline{0}$ and the synchronization set $S = \{h_1, h_2\}$, it follows that $P/AType_H \not\approx_{PB} ((P \parallel_S \Pi)/S) \backslash AType_H$. ■

The example above shows that the $BSPNI$ property is not stricter than the $SBNDC$ property, i.e. even if a system P_{nd} is not $SBNDC$ secure, its probabilistic extension P may be $BSPNI$ secure. Moreover, we have seen that system

$P \stackrel{\Delta}{=} l.\underline{0} +^p h.h.l.\underline{0}$ is *BSPNI* secure (see the proof of Proposition 5.7), while its nondeterministic version $P_{nd} \stackrel{\Delta}{=} l.\underline{0} + h.h.l.\underline{0}$ does not satisfy the *BNDC* property (see Example 2.5). Therefore, the *BSPNI* property is not stricter than the *BNDC* property. In addition, Example 5.9 shows a process P which is *PBNDC* secure, but not *SPBNDC* secure. As it is easy to verify, its nondeterministic counterpart P_{nd} is *BNDC* secure, but not *SBNDC* secure. Hence, the *PBNDC* property is not stricter than the *SBNDC* property. This concludes the discussion on the inclusion relations among security properties. In Table 4 we graphically report the relations between the nondeterministic setting and the probabilistic one. The figure shows, e.g., that if $P \in \mathcal{G}^f$ is *BSPNI* secure, then its nondeterministic version $P_{nd} \in \mathcal{G}_{nd}$ may be not *BNDC* secure, since the *BSPNI* labeled graph is not included in the *BNDC* labeled graph. Table 4 reports also a legend with some examples related to the different regions of the figure. Note that we are not aware of a process P such that P is *BSPNI* secure and P_{nd} is *SBNDC* secure, but P is not *PBNDC* (*SPBNDC*) secure. If such a process does not exist, then we would have an alternative, easily verifiable characterization of *PBNDC*.



Some examples:

1 : $(\tau.(l.\underline{0} +^{0.5} h_1.\underline{0}) +^{0.5} \tau.(l.\underline{0} +^{0.5} \tau.\underline{0})) +^{0.3} h_2.l.\underline{0}$
2 : example 5.8
3 : example 5.4
4 : example 5.13
5 : example 5.9
6 : example 5.5
7 : example 5.2

Table 4: Inclusion relationship among security properties

Finally, we point out the following remark. In the nondeterministic case the strong version of the *BNDC* turns out to be important to reveal potential behaviors which are clearly insecure (see, e.g., Examples 2.8 and 2.9). On the other hand, in our view in the probabilistic case it is not easy to find a realistic example showing that the *SPBNDC* is stronger than the *PBNDC*, which seems to be the adequate notion of security in most cases. For instance, Example 5.9 assumes a very strong knowledge of the high user, who should be able to see the result of an internal probabilistic choice within the system to set up a covert channel which is captured by the *SPBNDC*, but not by the *PBNDC*.

5.5 Interplay between Security and Probability

By modeling the probabilistic behavior of systems, new aspects of potential information flows from high level to low level can be analyzed. In particular, while in a nondeterministic, qualitative view we can just deduce that a system is or is not secure, in the probabilistic setting we can add that the system reveals an insecure information flow with a certain probability. Actually, the capacity of covert channels may be measured in a nondeterministic setting in the sense that the amount of leaked information changes according to the number of different behaviors of the high user that can be distinguished by a low user (see, e.g., [39]). Here, our claim is that probabilistic information offers the means for evaluating the real effectiveness of each of such unwanted behaviors.

From a practical standpoint, a quantitative, probabilistic approach to information flow analysis can be used to verify the security level of systems for which probabilities play an important role. For instance, many problems can be solved by using deterministic algorithms which turn out to be secure and require exponential time. On the other hand, probabilistic algorithms are often implemented that solve the same problems in polynomial time (see, e.g., [15, 41]). In such a case, the price to pay for a computational gain is the possibility for the observer of detecting an undesirable information flow. Because of such a possibility, by following an approach to information flow theory for nondeterministic processes, the probabilistic algorithms turn out to be insecure. Instead, by employing a probabilistic approach, we could formally prove that the same algorithms have an insecure behavior which is executed with probability 0 (or close to 0). Hence, a formal quantitative estimate of the unwanted behaviors is decisive to evaluate the security level of probabilistic systems. To this end, answers to questions like "What is the probability that an information flow from high level to low level has taken place?" cannot be provided if the verification of the security properties only depends on classical behavioral equivalences like the weak probabilistic bisimulation considered in Sect. 4. Based on the considerations above, here we introduce a quantitative notion of behavioral equivalence for deciding if two probabilistic processes behave almost (up to small fluctuations) the same or, more formally, for measuring the distance between probabilistic transition systems.

Along this line, in [14, 17] different pseudometrics are proposed that quantify the similarity of the behavior of probabilistic transition systems which are not bisimilar. In particular, in [17] the authors consider a metric on partial labeled Markov chains that are a generalization of our fully specified *GRTS*s (i.e. purely generative systems) in that for each state the sum of the probabilities of the outgoing transitions, if there are any, is less than (or equal to) 1, while in case of purely generative *GRTS*s such a sum sums up to 1. The need for replacing the notion of indistinguishability between processes by a notion of similarity has been emphasized also in [18, 19, 20] in the context of a probabilistic declarative language called *PCCP*. In particular, similarity among *PCCP* processes is checked by observing the behavior of the processes in all contexts, and by looking at how much the observed behaviors differ.

With respect to these proposals, here we decide to consider an enriched notion of weak probabilistic bisimulation which is able to tolerate fluctuations that make the security condition less restrictive. In particular, as we will see, such a relaxed definition of bisimulation allows us to relate systems which can have largely different possible behaviors under the condition that such behaviors are observable with a negligible probability.

Formally, the following definition of weak probabilistic bisimulation with ε-precision is used to compare processes for which the difference between the probabilistic behaviors is within a small distance.

Definition 5.14 A relation $R \subseteq \mathcal{G} \times \mathcal{G}$ is a weak probabilistic bisimulation with ε-precision, where $\varepsilon \in]0,1[$, if and only if, whenever $(P,Q) \in R$, then for all $C \in \mathcal{G}/R$:

- $|\, Prob(P, \tau^* \hat{a}, C) - Prob(Q, \tau^* \hat{a}, C)\,| \leq \varepsilon \quad \forall a \in GAct$
- $|\, Prob(P, a_*, C) - Prob(Q, a_*, C)\,| \leq \varepsilon \quad \forall a_* \in RAct$

We use the abbreviation $P \approx_{PB_\varepsilon} Q$ to denote that there exists a weak probabilistic bisimulation with ε-precision R containing the pair (P,Q); alternatively, we say that P (Q) is a ε-perturbation of Q (P). Note that since $\approx_{PB_\varepsilon}$ is an approximation relation, then it is not a transitive relation and, therefore, it cannot be an equivalence relation.

Example 5.15 Let us consider the system:

$$\begin{array}{rcl} P & \stackrel{\Delta}{=} & R +^p \tau.Q \\ Q & \stackrel{\Delta}{=} & l.\underline{0} +^q h.l.\underline{0} \\ R & \stackrel{\Delta}{=} & h.l'.\underline{0} \end{array}$$

represented by the *GRTS* of Fig. 14. As we have already seen in Example 5.2, the component Q is secure. On the contrary, the component R is clearly not secure, because the execution of the high-level action h allows a low user to observe the subsequent low-level action l' that otherwise is not enabled. The probabilistic information given in the operators is not needed to capture such an information flow, that is easily revealed in the nondeterministic setting by the *BSNNI* property. In other words, the probabilistic information given in P is not responsible for making the system insecure. On the other hand, the frequency of the possible outcomes of repeated executions of the system reveals that the behavior of P is secure with probability $1-p$ and discloses an unwanted information flow with probability p. In particular, in case p is a value close to 0, the low user can deduce the high-level interaction with a negligible probability, so that P could be considered as a good approximation of a secure system.

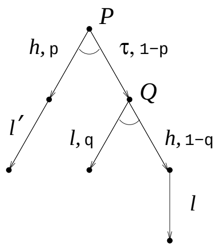


Figure 14: Example of probabilistic information flow

Formally, $P\backslash AType_H \not\approx_{PB} P/AType_H$, but $P\backslash AType_H$ is a p-perturbation of $P/AType_H$, as $P\backslash AType_H \stackrel{\text{iso}}{=} \tau.l.\underline{0} \approx_{PB} \tau.(l.\underline{0} +^q \tau.l.\underline{0}) \approx_{PB_p} \tau.l'.\underline{0} +^p \tau.(l.\underline{0} +^q \tau.l.\underline{0}) \stackrel{\text{iso}}{=} P/AType_H$. Hence, the system can be considered secure enough as p tends to the value 0. Note that, according to the definition of weak probabilistic bisimulation with ε-precision, if p is less than the threshold ε, then the subsystem P' reached from $P/AType_H$ by executing the hidden high-level action h is simply disregarded, since it expresses a behavior of the system reachable with a negligible probability. Therefore P' is not to be related with any corresponding behavior of the system $P\backslash AType_H$. ∎

Example 5.16 Let us consider the process:

$$P \triangleq h_*.(l'.\underline{0} +^p l.\underline{0}) + l.\underline{0}.$$

In the absence of the high user, the low view of the system is represented by the output action l, while the interaction of the high user affects the low behavior by adding the output action l' to the potential low interactions of the system. The nondeterministic version of P does not satisfy the *BSNNI* property and, as a consequence, P is not *BSPNI* secure. However, it is worth noting that in any way the high user resolves the nondeterminism in P, we have that $P\backslash AType_H$ is a p-perturbation of $P/^q_h$ $\forall q \in]0,1[$. In particular, supposed C the class of the null term $\underline{0}$, we have $Prob(P\backslash AType_H, \tau^* l, C) = 1$ and $Prob(P\backslash AType_H, \tau^* l', C) = 0$, while $Prob(P/^q_h, \tau^* l, C) = (1-q)+q\cdot(1-p) = 1-q\cdot p$ and $Prob(P/^q_h, \tau^* l', C) = q\cdot p$. Hence, $|Prob(P\backslash ATypeH, \tau^* l, C) - Prob(P/^q_h, \tau^* l, C)| = q\cdot p < p$ and $|Prob(P\backslash ATypeH, \tau^* l', C) - Prob(P/^q_h, \tau^* l', C)| = q\cdot p < p$. Therefore, for all $q \in]0,1[$, $P\backslash ATypeH \approx_{PB_p} P/^q_h$. This result is particularly meaningful since it says that if p is a small value, then the illegal information flow is negligible and P is an acceptable approximation of a secure system. ■

Example 5.17 As another example, let us consider the probabilistic process of Example 5.3:

$$P \triangleq (l.\underline{0} +^p l.l'.\underline{0}) +^q l.h.l'.\underline{0}.$$

As we have seen, P is not *BSPNI* secure. Now, let us denote by C_1 the equivalence class of the null term $\underline{0}$ and by C_2 the equivalence class of term $l'.\underline{0}$. On the one hand, we have $Prob(P\backslash ATypeH, \tau^* l, C_1) = q\cdot p + 1 - q$ and $Prob(P\backslash ATypeH, \tau^* l, C_2) = q\cdot(1-p)$. On the other hand, we have $Prob(P/ATypeH, \tau^* l, C_1) = q\cdot p$ and $Prob(P/ATypeH, \tau^* l, C_2) = 1 - q\cdot p$. Therefore, $|Prob(P\backslash ATypeH, \tau^* l, C_1) - Prob(P/ATypeH, \tau^* l, C_1)| = 1 - q = |Prob(P\backslash ATypeH, \tau^* l, C_2) - Prob(P/ATypeH, \tau^* l, C_2)|$, from which we derive $P\backslash ATypeH \approx_{PB_\varepsilon} P/ATypeH$ if $q > 1-\varepsilon$. Intuitively, if q is close to 1 then the insecure behavior in P is enabled with a negligible probability, i.e. the low view of P, with or without the interaction with the high user, changes according to a small ε-fluctuation. This means that by observing the outcome of repeated executions of the system a low user does not notice the behavior of the high user, since the high interferences can alter the frequency associated with each possible low outcome according to small, negligible fluctuations. ■

To conclude this section, we propose a simple but real example (inspired by [19]) showing the value of a quantitative notion of information flow.

Example 5.18 Consider an Automatic Teller Machine (ATM) which gives cash if and only if the client inserts the unique, correct PIN number i (of m possible PINs) within a fixed number, say n, of attempts, after which the ATM retires the card:

$$ATM_k \triangleq insPIN_{i_*}.cash.ATM_1 + \sum_{j=1,j\neq i}^{m} insPIN_{j_*}.fail.ATM_{k+1} \quad 0 < k < n$$
$$ATM_n \triangleq insPIN_{i_*}.cash.ATM_1 + \sum_{j=1,j\neq i}^{m} insPIN_{j_*}.alarm.ATM_1$$

An attacker which is in possession of the card (but not of the PIN) may try to illegally withdraw the cash:

$$Spy \triangleq insPIN_1.Spy' +^{p_1} (insPIN_2.Spy' +^{p_2} \ldots)$$
$$Spy' \triangleq cash_*.spend.\underline{0} + fail_*.Spy + alarm_*.flee.\underline{0}$$

We assume that *cash* is a low-level action, since it expresses the tangible proof that a dishonest spy withdrew the cash, while all the other activities are high-level actions. If we consider the composed system

$$ATMSys \triangleq ATM_1 \parallel_{\{cash,alarm,fail,insPIN_i,\, i=1,\ldots,m\}} Spy$$

and check a nondeterministic property, say the *BSNNI*, we obtain that the system is not secure, since if we hide the high-level actions then the event *cash* is observable, while if we purge the system of the high-level actions then the action *cash* is not executable. Obviously, the nondeterministic property captures the fact that an illegal behavior can be observed in case the spy guesses the right PIN. In practice, such an event is quite negligible. For instance, if for any attempt the spy randomly samples a PIN value according to a uniform distribution and if we take two realistic parameters, i.e. $m = 100000$ and $n = 3$, then, by assuming C the equivalence class of the null term, we obtain $Prob(ATMSys/AType_H, \tau^* cash, C) \approx 0.00003$. Formally, if we employ the weak probabilistic bisimulation with ε-precision (and $\varepsilon = 0.00003$) then the system turns out to be *BSPNI* secure since the probability of observing the illegal cash leakage is considered to be negligible. ■

6 A Case Study: Probabilistic Routing

In this section, we present a case study showing that our approach can be exploited to analyze security properties of real systems. In particular, we consider a routing mechanism of the OSI network layer [59], and we model an internetworking node, termed Interface Message Processor (IMP), that is responsible for deciding, at the network layer, on which output link an arriving packet should be sent, depending on the destination of that packet (see [11] for an algebraic description of an IMP). In this paper, we consider a version of the IMP with two possible destinations (one is considered to be a high-level link and one is considered to be a low-level link) and a single path for each of them, and we show how to analyze the information flows that may occur.

6.1 Algebraic Specification of the IMP

The algebraic specification of the IMP is presented in Table 5 in case of two possible destinations, a and b, where a is dedicated to the transmission of public data (the actions handling packets destined to a represent the low-level view of the system), while b is reserved to the transmission of confidential data (the actions handling packets destined to b represent the high-level view of the system). Therefore the actions related to packets destined to a (*receivea*, *waita*, *accepta*, *senda*, *busya*, *transma*, and *avail_cha*) are low-level actions, while all the other actions, related to packets destined to b, are high-level actions.

The system *IMP* consists of the parallel composition of three components: a term *Arrivals* modeling the incoming traffic, a term *Router* modeling the core of the IMP, and a term *Channels* modeling the outgoing channels. The parameters which probabilistically guide the choice of the component to be executed are p_1 and p_2, i.e. in each system state the term executing the next move is *Arrivals* with probability p_1, *Router* with probability $p_2 \cdot (1 - p_1)$, and *Channels* with probability $(1-p_2)\cdot(1-p_1)$. Now let us describe in detail the behavior of each component of the system.

$$
\begin{array}{rcl}
IMP & \triangleq & Arrivals \parallel_S^{p_1} (Router \parallel_C^{p_2} Channels) \\
S & = & \{receivea,\ receiveb\} \\
C & = & \{avail\_cha,\ avail\_chb,\ transma,\ transmb\} \\
 & & \\
Arrivals & \triangleq & Arrivala \parallel^{\frac{1}{2}} Arrivalb \\
Channels & \triangleq & Channela \parallel^{v} Channelb \\
Router & \triangleq & (Queues \parallel_{S_1 \cup I} Switch) \parallel_I Idle \\
S_1 & = & \{accepta,\ acceptb\} \qquad I \ = \ \{idle\} \\
 & & \\
Queues & \triangleq & Queuea \parallel_I Queueb \\
Switch & \triangleq & (Managera \parallel_{S'_2} Routinga) \ \parallel_I^p \ (Managerb \parallel_{S''_2} Routingb) \\
S'_2 & = & \{senda,\ busya\} \qquad S''_2 \ = \ \{sendb,\ busyb\} \\
 & & \\
Arrivala & \triangleq & receivea.Arrivala +^{ra} waita.Arrivala \\
Arrivalb & \triangleq & receiveb.Arrivalb +^{rb} waitb.Arrivalb \\
Queuea & \triangleq & receivea_*.accepta_*.Queuea + idle_*.Queuea \\
Queueb & \triangleq & receiveb_*.acceptb_*.Queueb + idle_*.Queueb \\
Managera & \triangleq & accepta.Manager'a + idle_*.Managera \\
Manager'a & \triangleq & senda.Managera + busya.Manager'a \\
Managerb & \triangleq & acceptb.Manager'b + idle_*.Managerb \\
Manager'b & \triangleq & sendb.Managerb + busyb.Manager'b \\
Routinga & \triangleq & senda_*.Routing'a + avail\_cha_*.Routinga \\
Routing'a & \triangleq & transma_*.Routinga + busya_*.Routing'a \\
Routingb & \triangleq & sendb_*.Routing'b + avail\_chb_*.Routingb \\
Routing'b & \triangleq & transmb_*.Routingb + busyb_*.Routing'b \\
Idle & \triangleq & idle.Idle \\
Channela & \triangleq & avail\_cha.Channela + transma.Channela \\
Channelb & \triangleq & avail\_chb.Channelb + transmb.Channelb
\end{array}
$$

Table 5: IMP Model

Term *Arrivals* is composed of two processes *Arrivala* and *Arrivalb*, that model the incoming traffic directed to destinations a and b, respectively. In case of process *Arrivala* (we can argue similarly for process *Arrivalb*), the parameter ra expresses the probability of observing the arrival of a packet destined to a. Such an event is modeled by the action *receivea* that synchronizes with the corresponding reactive action in the queue for packets a of term *Router*. In case such a queue is full the action *receivea* is not enabled and the arriving packets are lost with probability 1 through the execution of the generative action *waita*.

Term *Router*, which is the core of the IMP, manages the packets directed to the two destinations via a probabilistic scheduler. In particular, term *Router* is composed of a term *Queues* collecting the arriving packets, a term *Switch* delivering the packets to the outgoing channels, and a term *Idle* modeling the phases of router inactivity. They are defined as follows. Term *Queues* consists of two terms, one for each kind of packet, that behave reactively. They receive packets destined to a (b) through reactive actions of type *receivea* (*receiveb*) and pass them to the *Switch* term through reactive actions of type *accepta* (*acceptb*). For the sake of simplicity, we assume both queues to be of size 1. Term *Switch* executes two different terms, each one managing packets with a certain destination (a or b) via a probabilistic scheduler with parameter p. The term delivering packets to destination a is composed of a term *Managera* and a term *Routinga* (similarly for the term delivering packets destined to b). Term *Managera* accepts packets destined to a from the dedicated queue through action *accepta* and afterwards either immediately passes them to the *Routinga* term through action *senda*, or waits until the channel is available for transmission by performing action *busya*. The *Routinga* term, which behaves reactively, accepts packets through a reactive action of type *senda* and transmits them through the corresponding channel via a reactive action of type *transma*. Whenever the *Routinga* term is busy, the transmission of packets destined to a is not possible and this is signalled to the *Managera* term through a synchronization by enabling the reactive action of type *busya*. Term *Idle* executes an action *idle*, expressing the inactivity periods of the IMP, whenever term *Router* has nothing else to do. More precisely, action *idle* is executed through a multiway synchronization with all the other *Router* components if and only if the input queues (term *Queues*) are empty and the core of the IMP (term *Switch*) is not waiting for delivering a packet to the channel. It is worth noting that in our model the action *idle* is considered to be a high-level action. The motivation for this will be considered when analyzing the security properties of the router.

Term *Channels* is composed of two processes modeling the possible channels a and b. The process dedicated to destination a (b) can either transmit a packet when the generative action *transma* (*transmb*) synchronizes with the corresponding reactive action of term *Routinga* (*Routingb*) managing that channel, or be available for transmission when the generative action *avail_cha* (*avail_chb*) is synchronized with the corresponding reactive action of term *Routinga* (*Routingb*).

6.2 Security Analysis of the IMP

Now, we are ready to analyze any information flow in the IMP model. The system described in Sect. 6.1 is secure, because no interaction is allowed among the high-level and the low-level components of the router. Formally, system *IMP* is *SPBNDC* secure. Intuitively, we observe that a low-level packet destined to a is received via a low-level ingoing channel (term *Arrivala*), handled by the low-level components of the core of the IMP (term *Router*), and sent on a low-level channel. All the high-level components of the router are not involved in (and cannot affect) the execution of the steps of such a pipeline. The same happens when a high-level packet passes through the router, in that no low-level component is affected by the execution of the actions handling such a packet. In general, if we purge the system of the high-level actions or if hide them, then the probability distribution of the low-level activities we obtain is alway the same. The only difference between the high-level user and the low-level one is that the high-level part of the router is allowed to know (via the execution of the action *idle*) the idleness periods of the router. On the other hand, if we make such a situation observable by the low-level part (by turning *idle* into a low-level action), then a low user can deduce the inactivity phases of the high-level components by observing each occurrence of the action *idle*. For instance, while a high-level packet is waiting for the transmission (e.g., in term $Routing'b$) the action *idle* is not executable and the low user is aware of this condition. This information leakage is easily captured by the *BSNNI* (*BSPNI*).

Now, let us consider the effect on the system behavior of some changes that introduce unwanted interactions between the low level and the high level. In a first variant, we allow the high-level packets (those destined to b) to be transmitted over the low-level channel destined to a. This means that the router sends on a public channel the data received from a confidential channel. This scenario can be described by changing term $Routing'b$, which models the passage of a packet over the channel destined to b, into the following one:

$$Routing'b \stackrel{\Delta}{=} (transmb_*.Routingb + transma_*.Routingb) + busyb_*.Routing'b$$

By enabling both actions $transma_*$ and $transmb_*$ in term $Routing'b$, we obtain the effect that each high-level packet can be passed either to the low-level channel or to the high-level channel, and such a choice is probabilistically guided by parameter v which models the probabilistic parallel execution of the two processes *Channela* and *Channelb* that execute the synchronizing generative actions *transma* and *transmb*, respectively. Whenever high-level packets pass through the router, a low user observes more packets sent over the low-level outgoing channel than packets arrived at the ingoing low-level channel. On the other hand, whenever the high-level part does not interact with the system, a low user observes a transmission over the outgoing low-level channel for each ingoing low-level packet. Formally, this direct information flow is captured by the *BSPNI* (*BSNNI*).

Now, we propose an example showing an information flow that is captured by the *BSPNI* and is not revealed in the nondeterministic case. Let us assume that a malfunctioning in the input ports of the router is responsible for loosing (delaying) some arriving low-level packets each time a high-level packet is handled by the related input queue. In this way, by observing the probability distribution of the low-level arrivals handled by the router, a low user is able to infer information about the high-level traffic. In practice, by monitoring the low-level channels, an attacker knows what happens in the high-level channels. We can model such a behavior by altering the parameter ra that models the incoming low-level traffic. To this aim, we change the *IMP* model as follows:

$$\begin{aligned}
Arrivals &\stackrel{\Delta}{=} Arrivala \,\|_A^{0.5}\, Arrivalb \\
A &= \{receiveb\} \\
Arrivala &\stackrel{\Delta}{=} (receivea.Arrivala +^{ra} waita.Arrivala) + receiveb_*.Arrivala' \\
Arrivala' &\stackrel{\Delta}{=} (receivea.Arrivala +^{ra'} waita.Arrivala) + receiveb_*.Arrivala'
\end{aligned}$$

The generative action *receiveb* of process *Arrivalb* synchronizes with the corresponding reactive action of process *Arrivala*. The effect of such a synchronization in process *Arrivala* is that the probability distribution of the low-level actions *receivea* and *waita* is altered, since it is guided by parameter ra' instead of parameter ra. If, e.g., $ra' < ra$, then the probability of observing the arrival of a low-level packet decreases because of the arrival of a high-level packet. In the nondeterministic setting, this indirect information flow cannot be captured, because the set of possible behaviors of the low-level part is not affected by the high-level components. On the other hand, in the probabilistic setting such a behavior is revealed by the *BSPNI*, since the probability distribution of the two low-level actions *receivea* and *waita* is affected by the behavior of the high-level components. Hence, the information flow described above is purely probabilistic (similarly as seen in Example 5.5) in the sense that the security properties for nondeterministic processes cannot reveal it by analyzing the possibilistic behavior of the system.

7 Conclusion

In this paper, we have considered the problem of extending the non-interference theory of [23, 24] to the probabilistic case. The motivation behind such an extension is that from the security viewpoint the analysis of the probabilistic behavior of systems can contribute to reveal those covert channels that are solely probabilistic and, therefore, not observable in a purely nondeterministic setting.

We have also emphasized that by considering the probabilistic aspect of the behavior of a system, a quantitative estimate of the information flow can be given, allowing us to overcome the classical, qualitative view based on which confidential information is or is not leaked to the low-level part. Approximating non-interference (through the introduction of a tolerance) can be also useful to mitigate the fact that often the modeler of a computer system is not provided with exact probabilistic information. On the other hand, it would be valuable to have probability-independent security notions which would imply security guarantees regardless particular probabilistic behaviors (see, e.g., [52]). As an interesting future work, we could define a probability-independent non-interference property, call it *PIN*, which says, e.g., that a nondeterministic process is *PIN* secure if it satisfies the *BSNNI* property and, for any probability distribution chosen to solve the nondeterministic choices, the probabilistic process we obtain is *BSPNI* secure.

Another aspect concerning approximated non-interference is that we can deal with real case studies varying from complex cryptographic protocols (see, e.g., [15]) to probabilistic algorithms used in secure systems (see, e.g., [41] which we have already studied in [3]). To this aim, we intend to extend the calculus of Sect. 3 with cryptographic primitives (see, e.g., [38, 21]), thus obtaining a formal paradigm including all the main ingredients needed to model computational aspects which are essential for reasoning about systems using real cryptographic operations (see, e.g., [5], where a general definition of probabilistic non-interference is proposed in the context of a model of asynchronous probabilistic reactive systems, and [38] that adopts an asymptotic notion of probabilistic equivalence).

Finally, it could be interesting to employ in our setting the general approach of [27], where non-interference properties like the *BNDC* are used to analyze different requirements of communication protocols, like secrecy, authenticity, and non-repudiation.

Acknowledgement

We thank the anonymous referees for their useful comments. This research has been funded by Progetto MURST "Metodi Formali per la Sicurezza e il Tempo" (MEFISTO).

References

[1] A. Aldini, M. Bernardo, R. Gorrieri, M. Roccetti, *"Comparing the QoS of Internet Audio Mechanisms via Formal Methods"*, ACM Transactions on Modeling and Computer Simulation 11(1):1-42, 2001

[2] A. Aldini, M. Bravetti, *"An Asynchronous Calculus for Generative-Reactive Probabilistic Systems"*, Proc. of 8th Int. Workshop on Process Algebra and Performance Modeling, Carleton Scientific, pp. 591-605, 2000

[3] A. Aldini, R. Gorrieri, *"Security Analysis of a Probabilistic Non-repudiation Protocol"*, Proc. of 2nd Joint Int. Workshop on Process Algebra and Performance Modelling, Probabilistic Methods in Verification, Springer LNCS 2399:17-36, 2002

[4] G. R. Andrews, R. P. Reitman, *"An Axiomatic Approach to Information Flow in Programs"*, ACM Transactions on Programming Languages and Systems, 2(1):56-76, 1980

[5] M. Backes, B. Pfitzmann, *"Computational Probabilistic Non-interference"*, in Proc. of 7th European Symposium on Research in Computer Security, Springer LNCS 2502:1-23, 2002

[6] J.C.M. Baeten, J.A. Bergstra, S.A. Smolka, *"Axiomatizing Probabilistic Processes: ACP with Generative Probabilities"*, Information and Computation 121:234-255, 1995

[7] C. Baier, H. Hermanns, *"Weak Bisimulation for Fully Probabilistic Processes"*, Proc. of 9th Int. Conf. on Computer Aided Verification, Springer LNCS 1254:119-130, 1997

[8] D. E. Bell, L. J. La Padula, *"Secure Computer Systems: Mathematical Foundations and Model"*, Technical Report M74-244, The MITRE Corporation, Bedford, MA, 1973

[9] M. Bravetti, *"Specification and Analysis of Stochastic Real-Time Systems"*, Ph.D. Thesis, University of Bologna (Italy), 2002, `ftp://ftp.cs.unibo.it/pub/techreports/`

[10] M. Bravetti, A. Aldini, *"Discrete Time Generative-reactive Probabilistic Processes with Different Advancing Speeds"*, Theoretical Computer Science 290(1):355-406, 2003

[11] M. Bravetti, A. Aldini, *"Expressing Processes with Different Action Durations through Probabilities"*, Proc. of 1st Joint Int. Workshop on Process Algebra and Performance Modelling, Probabilistic Methods in Verification, Springer LNCS 2165:168-183, 2001

[12] M. Bravetti, M. Bernardo, *"Compositional Asymmetric Cooperations for Process Algebras with Probabilities, Priorities, and Time"*, Proc. of 1st Workshop on Models for Time-Critical Systems, ENTCS 39(3), 2000

[13] M. Bravetti, M. Bernardo, R. Gorrieri, *"A Note on the Congruence Proof for Recursion in Markovian Bisimulation Equivalence"*, Proc. of 6th Int. Workshop on Process Algebras and Performance Modeling, C. Priami ed., pp. 153-164, 1998

[14] F. van Breugel, J. Worrell, *"Towards Quantitative Verification of Probabilistic Systems (extended abstract)"*, Proc. of 28th International Colloquium on Automata, Languages and Programming, Springer LNCS 2076:421-432, 2001

[15] C. Cachin, K. Kursawe, V. Shoup, *"Random oracles in constantipole: practical asynchronous Byzantine agreement using cryptography (extended abstract)"*, Proc. of 19th Symposium on Principles of Distributed Computing, pp. 123-132, ACM Press, 2000

[16] D. Clark, C. Hankin, S. Hunt, R. Nagarajan, *"Possibilistic Information Flow is safe for Probabilistic Non-Interference"*, 1st Workshop on Issues in the Theory of Security, 2000

[17] J. Desharnais, V. Gupta, R. Jagadeesan, P. Panangaden, *"The Metric Analogue of Weak Bisimulation for Probabilistic Processes"*, Proc. of 17th Symposium on Logic in Computer Science, IEEE Computer Society Press, 2002

[18] A. Di Pierro, C. Hankin, H. Wiklicky, *"Probabilistic Security Analysis in a Declarative Framework"*, 1st Workshop on Issues in the Theory of Security, 2000

[19] A. Di Pierro, C. Hankin, H. Wiklicky, *"Approximate Non-Interference"*, Proc. of 15th Computer Security Foundations Workshop, IEEE CS Press, pp. 1-17, 2002

[20] A. Di Pierro, C. Hankin, H. Wiklicky, *"Analysing Approximate Confinement under Uniform Attacks"*, Proc. of 9th Symposium on Static Analysis, Springer LNCS 2477:310-325, 2002

[21] A. Durante, R. Focardi, R. Gorrieri, *"A Compiler for Analysing Cryptographic Protocols Using Non-Interference"*, ACM Transactions on Software Engineering and Methodology, special issue on Software Engineering & Security 9(4):489-530, 2000

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

[23] R. Focardi, R. Gorrieri, *"A Classification of Security Properties"*, Journal of Computer Security 3(1):5-33, 1995

[24] R. Focardi, R. Gorrieri, *"The Compositional Security Checker: A Tool for the Verification of Information Flow Security Properties"*, IEEE Trans. Sof. Eng. 27:550-571, 1997

[25] R. Focardi, R. Gorrieri, *"Classification of Security Properties (Part I: Information Flow)"*, Foundations of Security Analysis and Design - Tutorial Lectures (R. Focardi and R. Gorrieri, Eds.), Springer LNCS 2171:331-396, 2001

[26] R. Focardi, R. Gorrieri, F. Martinelli, *"Information Flow Analysis in a Discrete-Time Process Algebra"*, Proc. of 13th IEEE Computer Security Foundations Workshop, pp. 170-184, IEEE CS Press, 2000

[27] R. Focardi, R. Gorrieri, F. Martinelli, *"Non Interference for the Analysis of Cryptographic Protocols"*, Proc. of 27th International Colloquium on Automata, Languages and Programming, Springer LNCS 1853:354-372, 2000

[28] S. N. Foley, *"A Taxonomy for Information Flow Policies and Models"*, Proc. of the IEEE Symposium on Security and Privacy, pp. 98-108, 1991

[29] R. Forster, *"Non Interference Properties for Nondeterministic Processes"*, Ph.D. Thesis, Oxford University, UK, 1999

[30] R.J. van Glabbeek, S.A. Smolka, B. Steffen, *"Reactive, Generative and Stratified Models of Probabilistic Processes"*, Information and Computation 121:59-80, 1995

[31] J.A. Goguen, J. Meseguer, *"Security Policy and Security Models"*, Proc. of Symposium on Security and Privacy, IEEE CS Press, pp. 11-20, 1982

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

[33] J. W. Gray III, P. F. Syverson, *"A Logical Approach to Multilevel Security of Probabilistic Systems"*, Proc. of IEEE Computer Society Symposium on Research in Security and Privacy, pp. 164-176, 1992

[34] P.R. Halmos, *"Measure Theory"*, Springer-Verlag, 1950

[35] C.A.R. Hoare, *"Communicating Sequential Processes"*, Prentice Hall, 1985

[36] J. Jacob, *"Security Specifications"*, Proc. of IEEE Symposium on Security and Privacy, IEEE CS Press, pp. 14-23, 1988

[37] J. Jürjens, *"Secure information flow for concurrent processes"*, Proc. of 11th Int. Conf. on Concurrency Theory, Springer LNCS 1877:395-409, 2000

[38] P. Lincoln, J. C. Mitchell, M. Mitchell, A. Scedrov *"A Probabilistic Poly-Time Framework for Protocol Analysis"*, Proc. of 5th ACM Conf. on Computer and Communications Security, ACM Press, pp. 112-121, 1998

[39] G. Lowe, *"Quantifying Information Flow"*, Proc. of 15th Computer Security Foundations Workshop, IEEE CS Press, pp. 18-31, 2002

[40] H. Mantel, *"Possibilistic Definitions of Security - An Assembly Kit"*, Proc. of 13th IEEE Computer Security Foundations Workshop, IEEE CS Press, pp. 185-199, 2000

[41] O. Markowitch, Y. Roggeman, *" Probabilistic Non-Repudiation without Trusted Third Party"*, 2nd Conf. on Security in Communication Networks, 1999

[42] J. McLean, *"Security Models and Information Flow"*, Proc. of IEEE Symposium on Research in Security and Privacy, pp. 180-187, 1990

[43] J. K. Millen, *"Hookup Security for Synchronous Machines"*, Proc. of 3rd Computer Security Foundations Workshop, IEEE CS Press, pp. 84-90, 1990

[44] R. Milner, *"Communication and Concurrency"*, Prentice Hall, 1989

[45] A.W. Roscoe, *"CSP and Determinism in Security Modelling"*, Proc. of IEEE Symposium on Security and Privacy, IEEE CS Press, pp. 114-127, 1995

[46] A. W. Roscoe, G. M. Reed, R. Forster, *"The successes and failures of behavioural models"*, Millenial Perspectives in Computer Science, Davies, Roscoe, and Woodcock (eds), Palgrave, 2000

[47] A. W. Roscoe, J.C.P. Woodcock, L. Wulf, *"Noninterference through Determinism"*, Proc. of European Symposium Research Computer Security, Springer LNCS 875:33-53, 1994

[48] P.Y.A. Ryan, *"A CSP Formulation of Non-Interference"*, Cipher, IEEE CS Press, pp. 19-27, 1991

[49] P.Y.A. Ryan, J. McLean, J. Millen, V. Gligor, *"Non-interference: who needs it?"*, Proc. of 14th Computer Security Foundations Workshop, IEEE CS Press, pp. 237-238, 2001

[50] P.Y.A. Ryan, S. Schneider, *"Process Algebra and Noninterference"*, Proc. of 12th Computer Security Foundations Workshop, IEEE CS Press, pp. 214-227, 1999

[51] A. Sabelfeld, D. Sands, *"A Per Model of Secure Information Flow in Sequential Programs"*, Proc. of 8th European Symposium on Programming, Springer LNCS 1576:40-58, 1999

[52] A. Sabelfeld, D. Sands, *"Probabilistic Noninterference for Multi-threaded Programs"*, Proc. of 13th Computer Security Foundations Workshop, IEEE CS Press, pp. 200-215, 2000

[53] D. Sangiorgi, R. Milner, *"The Problem of Weak Bisimulation up to"*, Proc. of 3rd Int. Conf. on Concurrency Theory, Springer LNCS 630:32-46, 1992

[54] S. Schneider, *"Concurrent and Real-Time Systems: the CSP Approach"*, J. Wiley & sons, Inc., 1999

[55] G. Smith, *"A new Type System for Secure Information Flow"*, Proc. of 14th Computer Security Foundations Workshop, IEEE CS Press, pp. 115-125, 2001

[56] G. Smith, *"Weak Probabilistic Bisimulation for Secure Information Flow"*, 2nd ACM SIGPLAN and IFIP WG 1.7 Workshop on Issues in the Theory of Security, 2002

[57] S. A. Smolka, B. Steffen, *"Priority as Extremal Probability"*, Formal Aspects of Computing 8(5):585-606, 1996

[58] P. Syverson, J. W. Gray III, *"The Epistemic Representation of Information Flow Security in Probabilistic Systems"*, Proc. of 8th Computer Security Foundations Workshop, IEEE CS Press, pp. 152-166, 1995

[59] A.S. Tanenbaum, *"Computer Networks"*, Prentice Hall, 1996

[60] D. Volpano, G. Smith, *"Verifying Secrets and Relative Secrecy"*, Proc. of ACM Symposium on Principles of Programming Languages, ACM Press, pp. 268-276, 2000

[61] J. T. Wittbold, D. M. Johnson, *"Information Flow in Nondeterministic Systems"*, Proc. of 1990 Symposium on Research in Security and Privacy, IEEE CS Press, pp. 144-161, 1990

A Proof of Results

A.1 Proposition 2.10

To show that $BNDC \subset BSNNI$, consider process $Input \stackrel{\Delta}{=} \sum_{h \in AType_H} h_*.Input$ that can repeatedly perform each high-level input action h_* with $h \in AType_H$. Hence, for each process $P \in \mathcal{G}_{nd}$, $P \,\|_{AType_H} Input$ enables all (and only) the actions of P, i.e. P and $P \,\|_{AType_H} Input$ are isomorphic. By $BNDC$ definition, it follows $P \backslash AType_H \approx_B ((P \,\|_{AType_H} Input)/AType_H) \backslash AType_H \approx_B P/AType_H$. Example 2.5 proves that the inclusion above is strict.

Now, we show that $SBNDC \subset BNDC$. To this end, let us consider $P \in \mathcal{G}_{nd}$, $\Pi \in \mathcal{G}_{H_{nd}}$, $S \subseteq AType_H$, and $R_{P,\Pi,S}$ a relation defined as follows: $R_{P,\Pi,S} = \{(P' \backslash AType_H, ((P' \,\|_S \Pi')/S) \backslash AType_H) \mid P' \in Der(P), \Pi' \in Der(\Pi)\}$. We now prove that, if P is $SBNDC$, then $R_{P,\Pi,S}$ is a bisimulation up to $\approx_B$ [53]. By inspection of possible cases, it follows:

- if $P' \backslash AType_H \xrightarrow{\pi} P'' \backslash AType_H$ with $t(\pi) \in AType_L$, then we have that $((P' \,\|_S \Pi')/S) \backslash AType_H \xrightarrow{\pi} ((P'' \,\|_S \Pi')/S) \backslash AType_H$, and viceversa (we recall that P' must synchronize on the high-level actions of the set $S \subseteq AType_H$ only);

- similarly, if $P'\backslash AType_H \xrightarrow{\tau} P''\backslash AType_H$ then $((P' \parallel_S \Pi')/S)\backslash AType_H \xrightarrow{\tau} ((P'' \parallel_S \Pi')/S)\backslash AType_H$, and viceversa;

- the most interesting case arises whenever $((P' \parallel_S \Pi')/S)\backslash AType_H \xrightarrow{\tau} ((P'' \parallel_S \Pi'')/S)\backslash AType_H$, due to the fact that processes P' and Π' synchronize on a high action π with $t(\pi) \in S$. In such a case, by hypothesis $P' \xrightarrow{\pi} P''$ and $P'\backslash AType_H \approx_B P''\backslash AType_H$.

Therefore, $R_{P,\Pi,S}$ is a bisimulation up to $\approx_B$ and P is *BNDC*. We also observe that since $P'\backslash AType_H \approx_B ((P' \parallel_S \Pi)/S)\backslash AType_H)$ for all $P' \in Der(P)$, then P' is *BNDC* (and also *BSNNI*), which is a condition stricter than that we wanted to prove. Finally, the counterexamples given in Examples 2.7, 2.8, and 2.9 prove that the inclusion above is strict. ■

From the proof of Proposition 2.10 we derive an alternative definition for the *BNDC* property, as specified in the following lemma.

Lemma A.1 $P \in BNDC \Leftrightarrow P/AType_H \approx_B ((P \parallel_S \Pi)/S)\backslash AType_H, \forall \Pi \in \mathcal{G}_{H_{nd}}$, $\forall S \subseteq AType_H$.

Proof If $P \in BNDC$ then $P\backslash AType_H \approx_B ((P \parallel_S \Pi)/S)\backslash AType_H$, $\forall \Pi \in \mathcal{G}_{H_{nd}}$, $\forall S \subseteq AType_H$. By Proposition 2.10, it follows $P \in BSNNI$, i.e. $P\backslash AType_H \approx_B P/AType_H$, from which we derive $P/AType_H \approx_B ((P \parallel_S \Pi)/S)\backslash AType_H$, $\forall \Pi \in \mathcal{G}_{H_{nd}}$, $\forall S \subseteq AType_H$.

On the other hand, if $P/AType_H \approx_B ((P \parallel_S \Pi)/S)\backslash AType_H, \forall \Pi \in \mathcal{G}_{H_{nd}}, S \subseteq AType_H$, then, by taking the empty process $\Pi \triangleq \underline{0}$ and as the synchronization set S the empty set $\emptyset$, we have that $P/AType_H \approx_B ((P \parallel_\emptyset \underline{0})/\emptyset)\backslash AType_H \approx_B P\backslash AType_H$ and, as a consequence, $P\backslash AType_H \approx_B ((P \parallel_S \Pi)/S)\backslash AType_H, \forall \Pi \in \mathcal{G}_{H_{nd}}, S \subseteq AType_H$, i.e. P is *BNDC*. ■

A.2 Theorem 4.1

In order to show that the following equation

$$\begin{cases} 1 & \text{if } a = \tau \wedge P \in C \\ \sum_{Q \in \mathcal{G}} Prob(P, \tau, Q) \cdot Prob(Q, \tau^*, C) & \text{if } a = \tau \wedge P \notin C \\ \sum_{Q \in \mathcal{G}} Prob(P, \tau, Q) \cdot Prob(Q, \tau^* a, C) + Prob(P, a, C) & \text{if } a \neq \tau \end{cases}$$

has a least solution and is exactly $Prob(Exec(P, \tau^* \hat{a}, C))$, we interpret such an equation as a transformation from infinite vectors over $\mathbb{R}^+ \cup \{0\}$ (whose elements are the probabilities $Prob(P, \tau^* \hat{a}, C)$) to infinite vectors over $\mathbb{R}^+ \cup \{0\}$. The partial ordered set (poset) defined on such vectors[4] is a chain-complete poset (ccpo) with least element the infinite vector containing all zero values.

[4]We assume an ordering $\sqsubseteq$ among vectors such that if v and v' are two infinite vectors and $v \sqsubseteq v'$, then v is element-wise less than v'.

Moreover, the equation defined above is continuous (the transformation applied to the least upper bound of any chain coincides with the least upper bound of the transformations applied to the chain). Therefore, the statement trivially derives by applying the fixed point theorem, i.e. by repeatedly applying the transformation by starting from the least element of the ccpo (the vector with all zero elements). ■

A.3 Lemma 4.5

Lemma 4.5 says that if $P, Q \in \mathcal{G}$ and $P \approx_{PB} Q$ then $P_{nd} \approx_B Q_{nd}$.

To stem the proof we have to show that there exists a weak bisimulation R including the pair (P_{nd}, Q_{nd}). By inspection of possible cases, it follows:

- since $P \approx_{PB} Q$ it follows $Prob(P, \tau^*\hat{a}, C) = Prob(Q, \tau^*\hat{a}, C)$ for all $a \in GAct$. Now, assume that $P_{nd} \xrightarrow{a} P'_{nd}$, where P'_{nd} is the nondeterministic version of a term $P' \in C$. Then, by hypothesis there exists Q'_{nd} derived from $Q' \in C$ such that $Q_{nd} \stackrel{\hat{a}}{\Rightarrow} Q'_{nd}$, and $(P'_{nd}, Q'_{nd}) \in R$ because both P' and Q' belong to the same equivalence class C (the same if we exchange the roles of P and Q);
- since $P \approx_{PB} Q$ it follows $Prob(P, a_*, C) = Prob(Q, a_*, C)$ for all $a_* \in RAct$. Now, assume that $P_{nd} \xrightarrow{a_*} P'_{nd}$, where P'_{nd} is the nondeterministic version of a term $P' \in C$. Then, by hypothesis there exists Q'_{nd} derived from $Q' \in C$ such that $Q_{nd} \xrightarrow{a_*} Q'_{nd}$, and $(P'_{nd}, Q'_{nd}) \in R$ because both P' and Q' belong to the same equivalence class C (the same if we exchange the roles of P and Q). ■

A.4 Proposition 5.7

We show that $PBNDC \subset BSPNI$ by following the same idea formalized in the nondeterministic case. Given $P \in PBNDC$ and the set H_P of the high-level types which syntactically occur in P to be the set $\{h_1, \ldots, h_n\}$, we define $RH_P \stackrel{\Delta}{=} h_{1_*}.RH_P + (h_{2_*}.RH_P + (\ldots + (h_{n-1_*}.RH_P + h_{n_*}.RH_P)\ldots)$ as the process which repeatedly performs a purely nondeterministic choice among the high-level input actions with type in H_P (note that in such a process the probabilistic parameters in the alternative choice operators never come into play). By the semantics rules of the probabilistic parallel operator, it is easy to see that P and $P \,\|^p_{H_P}\, RH_P$ are isomorphic $\forall p \in]0,1[$. Due to this and by $PBNDC$ definition, we have that $P/^{\mathbf{p}}_{h_P} \approx_{PB} ((P \,\|^p_{H_P}\, RH_P)/^{\mathbf{p}}_{h_P}) \approx_{PB} ((P \,\|^p_{H_P}\, RH_P)/^{\mathbf{p}}_{h_P}) \backslash AType_H \approx_{PB} P \backslash AType_H$, $\forall\, p \in]0,1[$, $\mathbf{p} \in Seq^{|H_P|}_{]0,1[}$.

Now, we prove that the inclusion is strict by employing a probabilistic variant of the system described in Example 2.5. Let us consider the process $P \stackrel{\Delta}{=} l.\underline{0} +^p h.h.l.\underline{0}$, that is $BSPNI$ secure, since $P/AType_H \stackrel{\text{iso}}{=} l.\underline{0} +^p$

$\tau.\tau.l.\underline{0} \approx_{PB} l.\underline{0} \stackrel{\text{iso}}{=} P\backslash AType_H$. But, if we consider the process $\Pi \stackrel{\Delta}{=} h_*.\underline{0}$ we get $(P \|^q_{\{h\}} \Pi)/\{h\})\backslash AType_H \approx_{PB} l.\underline{0} +^p \tau.\underline{0} \not\approx_{PB} l.\underline{0}$. ■

Similarly as seen in the nondeterministic case, an expected result which follows from the proof of Proposition 5.7 is the following alternative definition for the *PBNDC* property.

Lemma A.2 $P \in PBNDC \Leftrightarrow P/^{\mathbf{p}}_{h_P} \approx_{PB} ((P \|^p_{\{h_1,\ldots,h_k\}} \Pi)/^{\mathbf{q}}_{h_1\ldots h_k})\backslash AType_H$, for each sequence $h_1 \ldots h_k$ of high-level types, $\forall \Pi \in \mathcal{G}^f_H$, $\forall p \in]0,1[$, $\forall \mathbf{p} \in Seq^{|H_P|}_{]0,1[}$, and $\forall \mathbf{q} \in Seq^k_{]0,1[}$.

Proof On the one hand, by Proposition 5.7, if $P \in PBNDC$ then $P \in BSPNI$. Hence, $P/^{\mathbf{p}}_{h_P} \approx_{PB} P\backslash AType_H \approx_{PB} ((P \|^p_{\{h_1,\ldots,h_k\}} \Pi)/^{\mathbf{q}}_{h_1\ldots h_k})\backslash AType_H$, for each sequence $h_1 \ldots h_k$ of high-level types, $\Pi \in \mathcal{G}^f_H$, $p \in]0,1[$, $\mathbf{p} \in Seq^{|H_P|}_{]0,1[}$, and $\mathbf{q} \in Seq^k_{]0,1[}$.

On the other hand, if $P/^{\mathbf{p}}_{h_P} \approx_{PB} ((P \|^p_{\{h_1,\ldots,h_k\}} \Pi)/^{\mathbf{q}}_{h_1\ldots h_k})\backslash AType_H$, for each sequence $h_1 \ldots h_k$ of high-level types, $\Pi \in \mathcal{G}^f_H$, $p \in]0,1[$, $\mathbf{p} \in Seq^{|H_P|}_{]0,1[}$, and $\mathbf{q} \in Seq^k_{]0,1[}$, then, by taking the empty process $\Pi \stackrel{\Delta}{=} \underline{0}$ and as the synchronization set the empty set $\emptyset$, we have $P/^{\mathbf{p}}_{h_P} \approx_{PB} ((P \|^p_{\emptyset} \underline{0})/_{\emptyset})\backslash AType_H \approx_{PB} P\backslash AType_H, \forall\, p \in]0,1[$, $\mathbf{p} \in Seq^{|H_P|}_{]0,1[}$. ■

A.5 Theorem 5.11

Let us assume $P \in \mathcal{G}^f$, $\Pi \in \mathcal{G}^f_H$, $\mathbf{h} = h_1 \ldots h_k$ a sequence of high-level types, S the set of types containing $h_1 \ldots h_k$, $p \in]0,1[$, and $\mathbf{q} \in Seq^k_{]0,1[}$.

Let $R_{P,\Pi,p,\mathbf{h},\mathbf{q}}$ be a relation defined as:

$$\{(P'\backslash AType_H, ((P' \|^p_S \Pi')/^{\mathbf{q}}_{\mathbf{h}})\backslash AType_H) \mid P' \in Der(P),\ \Pi' \in Der(\Pi)\}.$$

We want to prove that if P is *SPBNDC*, then $R_{P,\Pi,p,\mathbf{h},\mathbf{q}}$ is a weak probabilistic bisimulation up to $\approx_{PB}$. By inspection of the possible transitions, we proceed as follows.

$$\begin{array}{l} Prob(P'\backslash AType_H, a_*, C) = Prob(((P' \|^p_S \Pi')/^{\mathbf{q}}_{\mathbf{h}})\backslash AType_H, a_*, C) \\ \forall C \in \mathcal{G}^f/(\approx_{PB} \cup R_{P,\Pi,p,\mathbf{h},\mathbf{q}} \cup R^{-1}_{P,\Pi,p,\mathbf{h},\mathbf{q}})^+, \forall a \in RAct. \end{array}$$

If $a \in AType_H$ then both probabilities are equal to 0, and if $a \in AType_L$ then the two probabilities have the same value. Indeed, the low-level transitions of $P'\backslash AType_H$ are the same as those of $((P' \|^p_S \Pi')/^{\mathbf{q}}_{\mathbf{h}})\backslash AType_H$ and in both terms the probability distribution of the reactive bundle of type a is always the same as that of process P'.

$$\begin{array}{l} Prob(P'\backslash AType_H, \tau^*\hat{a}, C) = Prob(((P' \|^p_S \Pi')/^{\mathbf{q}}_{\mathbf{h}})\backslash AType_H, \tau^*\hat{a}, C) \\ \forall C \in \mathcal{G}^f/(\approx_{PB} \cup R_{P,\Pi,p,\mathbf{h},\mathbf{q}} \cup R^{-1}_{P,\Pi,p,\mathbf{h},\mathbf{q}})^+, \forall a \in GAct. \end{array}$$

In the following, we enumerate by $R_1, \ldots, R_n, \ldots, R_{max}$ all terms of the form $((P' \parallel^p_S \Pi')/^{\mathbf{q}}_{\mathbf{h}})\backslash AType_H$, with $P' \in Der(P)$ and $\Pi' \in Der(\Pi)$. Note that, since P and Π are in $\mathcal{G}^f$, then each R_n is in $\mathcal{G}^f$. Moreover, for terms R_n we syntactically distinguish among the internal actions originated from hiding (of high-level actions deriving from a synchronization between a high action of P' and one of Π'), denoted by τ^σ, and the internal actions inherited by the local τ actions of P', denoted by τ^λ. Then, we define $I^\sigma_n = \{m \mid \exists q : R_n \xrightarrow{\tau^\sigma, q} R_m\}$ ($I^\lambda_n = \{m \mid \exists q : R_n \xrightarrow{\tau^\lambda, q} R_m\}$) that denotes the set of terms reachable from R_n by executing an action τ^σ (τ^λ), and $p^\sigma_{n,m} = \sum \{\!| q \mid R_n \xrightarrow{\tau^\sigma, q} R_m |\!\}$ ($p^\lambda_{n,m} = \sum \{\!| q \mid R_n \xrightarrow{\tau^\lambda, q} R_m |\!\}$) denoting the probability of reaching R_m from R_n by executing an action τ^σ (τ^λ). We denote by $\mathcal{V} = \{P \in \mathcal{G}^f \mid P \not\stackrel{a}{\Rightarrow} \ \forall a \in GAct - \{\tau\}\}$ the set of processes whose derivatives do not enable visible generative actions. Finally, we assume the abbreviations $X_n = Prob(R_n, \tau^* \hat{a}, C)$ and $Z_n = Prob(R_n, \hat{a}, C)$. Due to the recursive definition of $Prob$, in the following we will deal with a finite number of equations over a finite number of variables X_n whose solutions are vectors over $\mathbb{R}^+ \cup \{0\}$. According to the notation above, we now prove the statement in the special case $a \in GAct - \{\tau\}$.

By definition of $Prob(R_n, \tau^* a, C)$ given in Sect. 4, and by following the notation introduced above, we observe that

$$X_n = \begin{cases} \sum_{m \in I^\sigma_n} p^\sigma_{n,m} \cdot X_m + \sum_{m \in I^\lambda_n} p^\lambda_{n,m} \cdot X_m + Z_n & \text{if } R_n \notin \mathcal{V} \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

Note that if $R_n \in \mathcal{V}$, i.e. if each derivative of R_n does not enable visible generative actions, then $Prob(R_n, \tau^* a, C) = 0$.

For terms R_n we now consider the probabilities $Prob(R^{-1}(R_n), \tau^* a, C)$. Given a term $R_n = ((P' \parallel^p_S \Pi')/^{\mathbf{q}}_{\mathbf{h}})\backslash AType_H$ (with $R^{-1}(R_n) = P'\backslash AType_H$), we define p_{R_n} representing the aggregate probability of the generative transitions of $R^{-1}(R_n)$ when considered in R_n. Denoted:

- $HidP = \sum \{\!| q \mid \exists h : P' \xrightarrow{h,q} \wedge \Pi' \xrightarrow{h_*} \wedge h \in S |\!\}$, denoting the sum of the probabilities of the high-level generative transitions of P' which will be hidden in R_n;

- $RisP = \sum \{\!| q \mid \exists h : P' \xrightarrow{h,q} \wedge (h \notin S \vee (h \in S \wedge \Pi' \not\xrightarrow{h_*})) |\!\}$, denoting the sum of the probabilities of the high-level generative transitions of P' which will be restricted in R_n;

- $Ris\Pi = \sum \{\!| q \mid \exists h : \Pi' \xrightarrow{h,q} \wedge (h \notin S \vee (h \in S \wedge P' \not\xrightarrow{h_*})) |\!\}$, denoting the sum of the probabilities of the high-level generative transitions of Π' which will be restricted in R_n;

- $Hid_* = \prod_{1 \leq i \leq k}(1 - \tilde{q}_i)$, where

$$\tilde{q}_i = \begin{cases} q_i & \text{if } P' \xrightarrow{h_{i_*}} \wedge \Pi' \xrightarrow{h_{i_*}}, \ h_i \in \mathbf{h} \wedge q_i \in \mathbf{q} \\ 0 & \text{otherwise,} \end{cases}$$

expressing the aggregate probability of the generative transitions of $P' \parallel_S^p \Pi'$ when considered in $(P' \parallel_S^p \Pi')/_{\mathbf{h}}^{\mathbf{q}}$;

the aggregate probability p_{R_n} of the generative transitions of $R^{-1}(R_n)$ when considered in R_n is:

$$p_{R_n} = \frac{Hid_* \cdot p \cdot (1 - (HidP + RisP))}{1 - (Hid_* \cdot p \cdot RisP + Hid_* \cdot (1-p) \cdot Ris\Pi)}$$

Hence, $1 - p_{R_n}$ represents the aggregate probability of the τ^σ transitions enabled by R_n, i.e. $1 - p_{R_n} = \sum_{m \in I_n^\sigma} p_{n,m}^\sigma$.

Now, supposed $X_n^r = Prob(R^{-1}(R_n), \tau^* a, C)$, we have

$$X_n^r = \begin{cases} \sum_{m \in I_n^\lambda} \frac{p_{n,m}^\lambda}{p_{R_n}} \cdot X_m^r + \frac{Z_n}{p_{R_n}} & \text{if } R_n \notin \mathcal{V} \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Note that by *SPBNDC* hypothesis, $R_n \in \mathcal{V}$ if and only if $R^{-1}(R_n) \in \mathcal{V}$ and, therefore, $X_n = 0$ if and only if $X_n^r = 0$. Moreover, for both equations 1 and 2, the unique solution is obtained by solving a system over a finite number of variables, each one defined by a linear equation. Hence, to show that equations 1 and 2 have the same solution, we replace in equation 1 each occurrence of variable X_n by variable X_n^r[5]:

$$X_n^r = \sum_{m \in I_n^\sigma} p_{n,m}^\sigma \cdot X_m^r + \sum_{m \in I_n^\lambda} p_{n,m}^\lambda \cdot X_m^r + Z_n$$

By *SPBNDC* hypothesis we have that $X_n^r = X_m^r \ \forall R_m$ reachable from R_n by executing a sequence of transitions τ^σ, therefore we can rewrite X_n^r as follows:

$$X_n^r = \sum_{m \in I_n^\sigma} p_{n,m}^\sigma \cdot X_n^r + \sum_{m \in I_n^\lambda} p_{n,m}^\lambda \cdot X_m^r + Z_n$$

and then

$$X_n^r = X_n^r \cdot (1 - p_{R_n}) + \sum_{m \in I_n^\lambda} p_{n,m}^\lambda \cdot X_m^r + Z_n.$$

Since $p_{R_n} > 0$ (by *SPBNDC* hypothesis, if $p_{R_n} = 0$ then $R_n \in \mathcal{V}$) it follows

$$X_n^r = X_n^r \cdot (1 - p_{R_n}) + p_{R_n} \cdot (\sum_{m \in I_n^\lambda} \frac{p_{n,m}^\lambda}{p_{R_n}} \cdot X_m^r + \frac{Z_n}{p_{R_n}})$$

[5] For the sake of brevity, we just report the case $R_n \notin \mathcal{V}$.

and

$$X_n^r = (1 - p_{R_n}) \cdot X_n^r + p_{R_n} \cdot X_n^r = X_n^r$$

By following the same arguments, it is easy to prove the statement in the special case $a = \tau$, by taking into consideration that in such a case the equations to be considered are

$$X_n = \begin{cases} \sum_{m \in I_n^\sigma} p_{n,m}^\sigma \cdot X_m + \sum_{m \in I_n^\lambda} p_{n,m}^\lambda \cdot X_m & \text{if } R_n \notin C \wedge R_n \notin \mathcal{V} \\ 1 & \text{if } R_n \in C \\ 0 & \text{otherwise} \end{cases}$$

and

$$X_n^r = \begin{cases} \sum_{m \in I_n^\lambda} \frac{p_{n,m}^\lambda}{p_{R_n}} \cdot X_m^r & \text{if } R_n \notin C \wedge R_n \notin \mathcal{V} \\ 1 & \text{if } R_n \in C \\ 0 & \text{otherwise} \end{cases}$$

In conclusion, $R_{P,\Pi,p,\mathbf{h},\mathbf{q}}$ is a bisimulation up to $\approx_{PB}$, hence P is *PBNDC*. The counterexample which shows that *SPBNDC* is stricter than *PBNDC* is given in Example 5.9. In particular, given $P \stackrel{\Delta}{=} \tau.(\tau.l_1.\underline{0} +^p h.l_2.\underline{0}) +^p \tau.(h.l_1.\underline{0} +^p \tau.l_2.\underline{0})$, we have $P \xrightarrow{\tau,p} P' \stackrel{\Delta}{=} \tau.l_1.\underline{0} +^p h.l_2.\underline{0}$ and $P' \xrightarrow{h,1-p} P'' \stackrel{\Delta}{=} l_2.\underline{0}$, but $P'' \backslash AType_H \stackrel{\text{iso}}{=} l_2.\underline{0} \not\approx_{PB} \tau.l_1.\underline{0} \stackrel{\text{iso}}{=} P' \backslash AType_H$. Therefore, P is not *SPBNDC* secure.

A.6 Conservative Extension Theorem

The conservative extension theorem says that if $P \in \mathcal{G}^f$ and P is *SP* secure, then P_{nd} is SP_{nd} secure.

We prove the statement in the case *SP* = *BSPNI* (SP_{nd} = *BSNNI*).

Formally, if P is *BSPNI* secure then $P/^{\mathbf{p}}_{h_P} \approx_{PB} P \backslash AType_H \; \forall \mathbf{p} \in Seq^{|H_P|}_{]0,1[}$. By Lemma 4.5 it follows that $P_{nd}/AType_H$ and $P_{nd} \backslash AType_H$, which are the nondeterministic counterparts of terms $P/^{\mathbf{p}}_{h_P}$, $\forall \mathbf{p} \in Seq^{|H_P|}_{]0,1[}$ and of term $P \backslash AType_H$, respectively, are weakly bisimulation equivalent.

On the other hand, the verification of the *BSNNI* for P_{nd} consists of checking the weak bisimulation equivalence between $P_{nd}/AType_H$ and $P_{nd} \backslash AType_H$, which are the same terms as those derived from $P/^{\mathbf{p}}_{h_P}$, $\forall \mathbf{p} \in Seq^{|H_P|}_{]0,1[}$ and $P \backslash AType_H$ by removing the probabilistic parameters from the operators.

We can argue similarly for the *PBNDC* and the *SPBNDC* to stem the complete proof. ■