Conditional Automata: a Tool for Safe Removal of Negligible Events

Roberto Segala and Andrea Turrini

Dipartimento di Informatica
Università di Verona - Italy

Abstract. Polynomially accurate simulations [19] are relations for Probabilistic Automata that require transitions to be matched up to negligible sets provided that computation lengths are polynomially bounded. They are proposed for verification of cryptographic protocols. In this paper we introduce a general construction on probabilistic automata, called Conditional Automata, that allows us to remove safely events that occur with negligible probability. The construction is justified in terms of polynomially accurate simulations. This, combined with the hierarchical and compositional verification style that underlies simulation relations, permits one to abstract one cryptographic component at a time in a complex system. We illustrate our construction through a simple example based on nonce generation, where we remove the event of repeated nonces.

1 Introduction

Hierarchical and compositional verification is one of the most successful byproducts of concurrency theory: compositional verification allows one to decompose complex systems into several sub-components that can be analyzed separately, while hierarchical verification permits one to analyze the interaction between several components on appropriate abstractions that do not include any irrelevant detail, thus reducing the overall complexity of a verification task.

Recently there has been increasing interest in hierarchical and/or compositional verification methods for security protocols [1, 4, 5, 7, 14, 16]. Indeed, a generic security protocol runs in an arbitrarily large network of agents and involves the interaction of several instances of functionalities like encryption schemes, signature schemes, nonce generators, etc. The resulting systems are rather complex to analyze, and therefore the ability to analyze components separately turns out to be a significant relief in the verification task and may allow us to reuse old proofs in other contexts. This approach, although of independent interest, turns out to be useful in automatic verification as well since it allows us to limit the well known state explosion problem.

In [19] we have used Probabilistic Automata [17] and proposed *polynomially accurate simulations* (PASs) as a tool for verifying hierarchically cryptographic protocols. The main idea is to adapt the simulation method of [11], already used in the context of distributed systems, to the area of security. Specifically, rather than requiring transitions of a concrete implementation to be matched exactly

by the abstract system, we allow for errors of negligible probability. This allows us to combine abstraction steps that rely on cryptographic assumptions, justified in terms of PASs, with abstraction steps that do not rely on any cryptographic assumption and that can be justified in terms of classical concurrency theory.

Polynomially accurate simulations are used successfully in [21] for the analysis of protocols, including a complete formal proof of the Dolev-Yao soundness result of [9]. In the case studies of [21] we start with a concrete description of the protocol under analysis, we abstract away the undesirable behaviors that occur with negligible probability, and then we proceed with other abstraction steps to prove that the resulting system always satisfies the protocol specification.

In this paper we focus on a general technique, used in [21], that allows us to remove negligible events from a concrete system while preserving all properties that hold with overwhelming probability in any context. For instance, if we have a complex system that obtains nonces from a nonce generator that draws random numbers, then we know that the probability of obtaining repeated nonces is negligible. It is then reasonable to analyze the complex system assuming that repeated nonces never occur. What is needed, however, is a theorem that supports our reasonable proposal. Let $\mathcal{A}$ be a probabilistic automaton describing the nonce generator that draws random numbers. Then repeated nonces may occur with negligible probability. Let G be the set of *good* states of $\mathcal{A}$ where no nonce is repeated. We introduce the concept of *conditional automaton* $\mathcal{A}|G$, obtained from $\mathcal{A}$ by replacing the target measure μ of each transition by the measure $\mu|G$, that is, the measure μ conditional on reaching a state from G. Our main theorem states that the identity relation is a PAS from $\mathcal{A}$ to $\mathcal{A}|G$ as well as from $\mathcal{A}|G$ to $\mathcal{A}$. The theorem holds for any set of states G whose complement is negligible. As a consequence, we can analyze systems hierarchically and remove safely any negligible event from any component by replacing it with its conditional counterpart.

Of course, although the final statement of he theorem turns out to be simple, we need to define precisely what we mean by negligible event in a probabilistic automaton, how we identify the set G of good states, and how PASs allow us to relate computations of a concrete system to computations of its abstraction. This is indeed the most complex part of the whole treatment. Informally, the notion of negligible event is defined by parameterizing probabilistic automata by a security parameter k and requiring bad states to have a negligible probability whenever computations are of polynomial length, G is identified by recording the past history in the states of an automaton, while correspondence between computations is stated by an *execution correspondence theorem* that turns out to be a generalization of the mapping lemma of [13].

One problem of the PASs of [19] is that, although they admit hierarchical verification, they are not compositional, that is, their existence is not preserved by parallel composition of probabilistic automata. On the other hand, a typical security protocol does not rely only on nonce generators, but rather it relies on encryption and/or signature schemes as well. Since also for encryption and signature schemes there are negligible events that we would like to remove (e.g.,

generation of repeated keys or generation of repeated ciphertexts in an IND-CCA schema), and since it is reasonable to apply conditional automata on each single primitive individually, compositionality of PASs becomes essential. For this reason, in the first part of the paper we introduce a new notion of PAS (state PAS or sPAS) that is stronger than the one of [19] and that is compositional as well. Roughly speaking, the PASs of [19] require matching of hyper-transitions up to negligible errors, while our new simulations require that the set of pairs of concrete and abstract states where the abstract states can match the transitions of the concrete state up to a negligible error has an overwhelming probability.

Related Work The simulation method is used already in the security literature. For example in [4] bisimulation relations are used to prove correctness of implementations according to the notion of reactive simulatability [16]. Although the definition of bisimulation is not worked out in full details, the idea is clear: transitions should be matched up to some "error sets", where an error set is a set of parts of transitions (e.g., messages, states) that have no corresponding piece in the abstract system; then, a separate argument shows that the global probability of the error sets is negligible. In our approach we impose conditions on the probabilities of the error sets directly in the step condition with the aim of bounding the global probability of the error sets to be negligible. Simulation relations are used also in [7] in the context of the Universally Composable framework [5]. In this case simulation relations are exact and the computational arguments are carried out with respect to a notion of approximated probabilistic language inclusion [6] based on the trace distribution semantics of [17]. Also in [14] there is a use of exact probabilistic bisimulations in the context of a probabilistic polynomial time process calculus. In this case the computational aspects are captured directly in the definition of the calculus. Another proposal of approximated probabilistic simulation relations appears in [15]. In this case a distance between probability of measures is defined based on the ability to produce similar trace distributions. Then an ε simulation matches steps from ε-distant measures by preserving ε-distance. Our definition is based on a different distance that may grow by a negligible amount at each step.

Though we are not aware of any formal treatment of the idea of conditional automaton in the literature, informal arguments along the lines of our results are common in the security community. In this context our result provides additional foundations and rigor to an idea that is common belief within the area of security. In [20] there is an idea of representing a correctness proof as a sequence of related games, where games are representations of attacks against protocols that are described at different levels of abstraction, and where two games are related if the difference of the probabilities of successful attacks is negligible. A general result of [20] is the *Difference Lemma* that implicitly gives us a way to transform a game G to a polynomially-equivalent one where some negligible failure event does not occur any more. Game transformations can be seen as an alternative hierarchical way of analyzing protocols. They are missing compositionality, though.

Of course almost all proofs of Dolev-Yao soundness (see, e.g., [3, 13] and related literature) rely on some form of abstraction from negligible events. The aim of this paper is to provide a tool based on probabilistic automata and simulation relations that allows us to safely remove negligible events from generic automata in order to obtain basic building blocks that can be used to recast Dolev-Yao soundness proofs following a modular approach (see, e.g., our case study in [21]). For instance, conditional automata can be used to abstract away events like signature forgery and key guessing. The main advantage of conditional automata proposed in this paper is that our construction is general and the removal of negligible events is justified in terms of approximate simulations that support hierarchical and compositional verification. This allows us to split events depending on properties of cryptographic primitives from events arising from the interaction of adversary and participants of the protocol, to study an event at a time independently from the others, and to exploit proof techniques and results of concurrency theory in other fields like verification of cryptographic protocols.

The rest of the paper is structured as follows: Section 2 recalls Probabilistic Automata and gives some basic notions on nonce generation; Section 3 introduces state polynomially accurate simulations as well as the execution correspondence theorem; Section 4 defines conditional automata and states the conditional automaton theorem; Section 5 gives some concluding remarks.

The proof of the results proposed in this paper can be found in [21].

2 Preliminaries

2.1 Probabilistic Automata

Given a set X, denote by $Disc(X)$ the set of discrete probability measures over X, and by $SubDisc(X)$ the set of discrete sub-probability measures over X. We call a discrete probability measure a *Dirac* measure if it assigns measure 1 to the singleton set $\{x\}$ (denote this measure by δ_x). We also call Dirac a sub-probability measure that assigns measure 0 to all objects. In the sequel discrete sub-probability measures are used to describe progress. If the measure of X is not 1, then it means that with some non-zero probability the system does not progress. Given $\rho \in SubDisc(X)$, we denote by $Supp(\rho)$ the *support* set $\{x \in X \mid \rho(x) > 0\}$ of ρ.

Given a set X, a set $G \subseteq X$, and a measure $\rho \in Disc(X)$ such that $\rho(G) > 0$, we call the *G-conditional measure of ρ*, denoted by $\rho|G$, the probability measure $\rho' \in Disc(X)$ such that for each $x \in X$,

$$\rho'(x) = \begin{cases} \rho(x)/\rho(G) & \text{if } x \in G, \\ 0 & \text{otherwise.} \end{cases}$$

A *Probabilistic Automaton* (PA) is a tuple $(S, \bar{s}, A, D)$ where S is a set of *states*, $\bar{s} \in S$ is the *start state*, A is a set of *actions*, and $D \subseteq S \times A \times Disc(S)$ is

a *transition relation*. The set of actions A is further partitioned into three sets I, O, H of input, output and internal (hidden) actions, respectively. We call the set $E = I \cup O$ the set of external actions.

Throughout the paper we let $\mathcal{A}$, $\mathcal{B}$ range over probabilistic automata, q, r, s range over states, a, b, c range over actions, and μ range over discrete measures over states. We also denote the generic elements of a probabilistic automaton $\mathcal{A}$ by S, $\bar{s}$, A, D, and we propagate primes and indices when necessary. Thus, for example, the probabilistic automaton $\mathcal{A}'_i$ has transition relation D'_i.

An element of a transition relation D is called a *transition* or a *step*. A transition $tr = (s, a, \mu)$, also denoted by $s \xrightarrow{a} \mu$, is said to *leave* from state s, to be *labeled* by a, and to *lead* to μ, denoted by μ_{tr}. We also say that state s *enables* action a, that action a is *enabled* from s, and that (s, a, μ) is enabled from s. We denote by $D(s)$ the set of all transitions enabled by s, by $D(a)$ the set of all transitions labeled by a, and by $D(sa)$ their intersection (that is, $D(sa) = D(s) \cap D(a)$).

An *execution fragment* of a PA $\mathcal{A}$ is a sequence of alternating states and actions, $\alpha = s_0 a_1 s_1 \ldots$, starting with a state and, if the sequence is finite, ending with a state, such that, for each non final index i, there exists a transition $(s_i, a_{i+1}, \mu_{i+1})$ in D with $\mu_{i+1}(s_{i+1}) > 0$. We say that an execution fragment is *finite* if it is a finite sequence, and we denote the last state of a finite execution fragment α by $lstate(\alpha)$. We define the length $|\alpha|$ of an execution fragment α to be the number of occurrences of actions in α. An *execution* of a PA $\mathcal{A}$ is an execution fragment of $\mathcal{A}$ whose first state is $\bar{s}$. We denote by $Frags^*(\mathcal{A})$ the set of finite execution fragments of $\mathcal{A}$, by $Frags(\mathcal{A})$ the set of finite or infinite execution fragments, and by $Execs^*(\mathcal{A})$, $Execs(\mathcal{A})$ the corresponding sets of executions.

A scheduler for a PA $\mathcal{A}$ is a function $\sigma \colon Frags^*(\mathcal{A}) \to SubDisc(D)$ such that, for each finite execution fragment α and each transition $tr = (s, a, \mu)$ with $\sigma(\alpha)(tr) > 0$, $s = lstate(\alpha)$. A scheduler σ and a state s induce a probability measure $\nu_{\sigma,s}$ over execution fragments as follows. The basic measurable events are the cones of finite execution fragments, where the cone of a finite execution fragment α, denoted by C_α, is the set $\{\alpha' \in Frags(\mathcal{A}) \mid \alpha \leqslant \alpha'\}$, where $\leqslant$ is the standard prefix preorder on sequences. The probability $\nu_{\sigma,s}$ of a cone C_α is defined recursively as follows:

$$\nu_{\sigma,s}(C_\alpha) = \begin{cases} 0 & \text{if } \alpha = q \text{ for some state } q \neq s, \\ 1 & \text{if } \alpha = s, \\ \nu_{\sigma,s}(C_{\alpha'}) \sum_{tr \in D(a)} \sigma(\alpha')(tr)\mu_{tr}(q) & \text{if } \alpha = \alpha' a q. \end{cases}$$

Standard measure theoretical arguments ensure that $\nu_{\sigma,s}$ extends uniquely to the σ-field generated by cones. If $s = \bar{s}$, we call the measure $\nu_{\sigma,\bar{s}}$ a *probabilistic execution* ν_σ of $\mathcal{A}$ and we say that it is generated by σ from $\bar{s}$.

Remark 1. A typical cryptographic protocol described in the literature does not have nondeterminism, and hence there is a unique maximal probabilistic execution, where by maximal we mean that we stop only when no more transitions are enabled. However, as soon as we abstract from some probabilistic choices

nondeterminism may appear. Imagine, for example, to abstract at the Dolev-Yao level just some of the functionalities of a protocol: in this case, we have that choices of abstracted functionalities are performed nondeterministically while the choices of other functionalities are still probabilistic. Besides, there is already evidence in the recent literature [8] that sometimes it is not convenient to hide nondeterminism under the carpet.

We now turn to the notion of simulation for probabilistic automata [17], defining first what it means to lift a relation on states to a relation on measures. Let $\mathcal{R}$ be a relation from a set X to a set Y. The lifting of $\mathcal{R}$, denoted by $\mathcal{L}(\mathcal{R})$, is a relation from $Disc(X)$ to $Disc(Y)$ such that $\rho_X \ \mathcal{L}(\mathcal{R}) \ \rho_Y$ if and only if there exists a *weighting function* $w\colon X \times Y \to [0,1]$ such that (1) $w(x,y) > 0$ implies $x \ \mathcal{R} \ y$, (2) $\sum_{x \in X} w(x,y) = \rho_Y(y)$, and (3) $\sum_{y \in Y} w(x,y) = \rho_X(x)$. An alternative definition of lifting given in a more probabilistic style states that $\rho_X \ \mathcal{L}(\mathcal{R}) \ \rho_Y$ if and only if there exists a joint measure w with marginal measures ρ_X and ρ_Y such that the support of w (i.e., pairs (x,y) such that $w(x,y) > 0$) is included in $\mathcal{R}$. If we view $\mathcal{R}$ as a pseudo-metric with values in $\{0, \infty\}$, then $\mathcal{L}(\mathcal{R})$ is the Kantorovich metric on probability measures [10].

A *simulation* from a PA $\mathcal{A}_1$ to a PA $\mathcal{A}_2$ is a relation $\mathcal{R}$ from S_1 to S_2 such that $\bar{s}_1 \ \mathcal{R} \ \bar{s}_2$ and for each pair $(s_1, s_2) \in \mathcal{R}$, if $(s_1, a, \mu_1) \in D_1$, then there exists a transition $s_2 \xrightarrow{a} \mu_2$ such that $\mu_1 \ \mathcal{L}(\mathcal{R}) \ \mu_2$. We say that $\mathcal{A}_1$ is simulated by $\mathcal{A}_2$, denoted by $\mathcal{A}_1 \preceq \mathcal{A}_2$, if there exists a simulation from $\mathcal{A}_1$ to $\mathcal{A}_2$. Relation $\preceq$ is transitive and preserved by the parallel composition $\mathcal{A}||\mathcal{B}$ of PAs $\mathcal{A}$ and $\mathcal{B}$. This is the key feature that enables hierarchical and modular verification. We do not define formally composition here and we refer the interested reader to [18].

2.2 Nonces and Negligibility

A *nonce* of length k is an element of $\{0,1\}^k$ that is used at most once. An ideal way to satisfy uniqueness of nonces is to use a repository that keeps track of the nonces distributed in the past and that responds to all requests by returning a new value each time. The practical way to satisfy the uniqueness of nonces is to choose them randomly from $\{0,1\}^k$. In this way, if we choose randomly two nonces of length k, the probability that they are the same is at most 2^{-k}. This property can be extended to any polynomial number of chosen nonces, as follows: denoted by *Poly* the set of positive polynomials on $\mathbb{N}$,

Property 1. For each $c \in \mathbb{N}$ and $p \in Poly$, there exists $\bar{k} \in \mathbb{N}$ such that for each $k > \bar{k}$, given $n_1, \ldots, n_{p(k)} \in \{0,1\}^k$, if n is chosen randomly and uniformly from $\{0,1\}^k$, then $\mathbf{Pr}(n \in \{n_1, \ldots, n_{p(k)}\}) < k^{-c}$.

The phrase "for each $c \in \mathbb{N}$ and $p \in Poly$, there exists $\bar{k} \in \mathbb{N}$ such that for each $k > \bar{k}$" is commonly used in the literature on complexity theory when bounding the probability of an attack in the computational model. Essentially, this phrase means that no matter how we bound polynomially the number of actions the attacker can perform ($\forall p \in Poly$) and how we limit the polynomial probability of the attack ($\forall c \in \mathbb{N}$), we are able to find a minimum value of the

security parameter ($\bar{k} \in \mathbb{N}$) such that for each larger value k the probability of the attack is less than the imposed limit k^{-c}, that is, the probability of the attack is less than each polynomial in k.

3 State Polynomially Accurate Simulation

3.1 State Polynomially Accurate Simulation

We start by enriching the notion of lifting of a relation to account for errors.

Definition 1. *Let $\mathcal{R}$ be a relation from X to Y and let $\varepsilon \in \mathbb{R}^{\geqslant 0}$. The ε-lifting of $\mathcal{R}$, denoted by $\mathcal{L}(\mathcal{R}, \varepsilon)$, is a relation from $Disc(X)$ to $Disc(Y)$ defined as follows: for each pair ρ_X, ρ_Y of measures in $Disc(X)$ and $Disc(Y)$, respectively, $\rho_X \; \mathcal{L}(\mathcal{R}, \varepsilon) \; \rho_Y$ if and only if either $\varepsilon \geqslant 1$ or there exists an* ε-weighting function *$w_\varepsilon \colon X \times Y \to [0,1]$ such that*

1. $w_\varepsilon(x,y) > 0 \implies x \; \mathcal{R} \; y$,
2. $\sum_{y \in Y} w_\varepsilon(x,y) \leqslant \rho_X(x)$,
3. $\sum_{x \in X} w_\varepsilon(x,y) \leqslant \rho_Y(y)$,
4. $\sum_{x \in X, y \in Y} w_\varepsilon(x,y) \geqslant 1 - \varepsilon$.

Alternatively we can say that there exists a joint sub-probability measure w with marginal measures dominated by ρ_X and ρ_Y, total mass at least $1-\varepsilon$, and such that the support of w is included in $\mathcal{R}$. Also, if we view $\mathcal{R}$ as a pseudo-metric with values in $\{0,1\}$, then the supremum $1-\varepsilon$ such that $\rho_X \; \mathcal{L}(\mathcal{R}, \varepsilon) \; \rho_Y$ holds is the Kantorovich distance between ρ_X and ρ_Y. Thus, testing $\rho_1 \; \mathcal{L}(\mathcal{R}, \varepsilon) \; \rho_2$ is equivalent to solving a maximum flow problem with the additional requirement that the flow is at least $1-\varepsilon$. The original and equivalent definition given in [19] is also interesting: $\rho_X \; \mathcal{L}(\mathcal{R}, \varepsilon) \; \rho_Y$ iff either $\varepsilon \geqslant 1$ or there exist $\rho'_X, \rho''_X, \rho'_Y, \rho''_Y$ such that $\rho_X = (1-\varepsilon)\rho'_X + \varepsilon\rho''_X$, $\rho_Y = (1-\varepsilon)\rho'_Y + \varepsilon\rho''_Y$, and $\rho'_X \; \mathcal{L}(\mathcal{R}) \; \rho'_Y$.

The second ingredient for sPASs is the ε-step condition. Informally, two states s_1, s_2 satisfy the ε-step condition if each transition from s_1 can be matched by a transition from s_2 up to an ε-bounded error.

Definition 2. *Let $\mathcal{A}_1$, $\mathcal{A}_2$ be two automata and $\mathcal{R}$ be a relation from S_1 to S_2. Given $s_1 \; \mathcal{R} \; s_2$, we say that s_1 and s_2 satisfies the* ε-step condition*, denoted by $s_1 \; St(\mathcal{R}, \varepsilon) \; s_2$, if for each $(s_1, a, \mu_1) \in D_1$, there exists a transition $s_2 \xrightarrow{a} \mu_2$ such that $\mu_1 \; \mathcal{L}(\mathcal{R}, \varepsilon) \; \mu_2$.*

For the third ingredient, consider a relation $\mathcal{R}$ and two measures ρ_1, ρ_2 that are $\mathcal{L}(\mathcal{R}, \gamma)$-related. We know that there is at least one γ-weighting function that supports $\rho_1 \; \mathcal{L}(\mathcal{R}, \gamma) \; \rho_2$. Indeed, there may be several such weighting functions. Among them we would like to select one that reduces as much as possible the mass of those pairs that are not $St(\mathcal{R}, \varepsilon)$-related. We first define formally the condition above by requiring the mass of the "bad" pairs to be bounded by η.

Definition 3. *Let $\mathcal{A}_1$, $\mathcal{A}_2$ be two PAs and let $\mathcal{R}$ be a relation from S_1 to S_2. We say that $\mathcal{R}$ is an* ε-η-approximate simulation *if*

– $\bar{s}_1 \mathcal{R} \bar{s}_2$;
– *for each* γ, μ_1, μ_2 *such that* $\mu_1 \mathcal{L}(\mathcal{R}, \gamma) \mu_2$ *there exists a* γ*-weighting function* w_γ *for* $\mu_1 \mathcal{L}(\mathcal{R}, \gamma) \mu_2$ *such that* $\sum\{w_\gamma(s_1, s_2) \mid s_1 \neg St(\mathcal{R}, \varepsilon)\ s_2\} < \eta$.

Remark 2. The step condition of the ε-η-approximate simulation is not immediate. Indeed, the reader may wonder

– *why not replacing* $\mathcal{L}(\mathcal{R})$ *with* $\mathcal{L}(\mathcal{R}, \varepsilon)$ *in the step condition of an ordinary simulation?* Our objective, illustrate in Figure 1, is to take a probabilistic execution ν of a simulated automaton and produce a probabilistic execution of the simulating one that corresponds up to some error that grows with the length n of ν, but not too much. If we use a $\mathcal{L}(\mathcal{R}, \varepsilon)$ relation, then the error is bounded above by $1 - (1 - \varepsilon)^n$, that is, the error gets exponentially close to 1, which is too much;
– *what is the purpose of* γ*?* In order to avoid the error being exponentially close to 1, we propose a way to say that the error grows at most by fixed amount at each step. In our definition γ represents the error before simulating the current step, while η represents a bound on the extra error introduced by the step;
– *why* ε-η*?* A step may be simulated up to some small error either because there are sufficiently many pairs of states that simulate each other perfectly, or because all pairs of states simulate each other up to some small error, or because of a combination of the two. The parameter ε measures the second kind of error, while η measures the first kind of error. The combination of the two measures the third kind of error. Thus, for instance, if we set ε to be 0 we consider only the first kind of error, and if we set η to be 0 we consider only the second kind of error.

Observe that in the sum above the fact that $w_\gamma(s_1, s_2) > 0$ implies implicitly that $s_1 \mathcal{R} s_2$ as well as $s_1 \in Supp(\mu_1)$ and $s_2 \in Supp(\mu_2)$.

The final step to define sPASs is to enrich ε-η-approximate simulations with computational elements. Thus, we use families of PAs and of relations parameterized by a security parameter k, we consider only pairs of measures that are reached within polynomial time, and we require ε and η to be smaller than any polynomial. We use the notion of probability measure on states reachable within n steps. The formal definition, which requires some basic measure theory, states that μ is reachable within n steps if there exists a probabilistic execution, supported on executions of length at most n, whose image measure of the *lstate* function is μ.

Definition 4. *A measure* μ *on states is said to be* reachable within n steps *if there exists a probabilistic execution* ν*, supported on finite executions of length at most* n*, such that* $\mu = lstate(\nu)$.

Definition 5. *Let* $\{\mathcal{A}_k^1\}_{k\in\mathbb{N}}$ *and* $\{\mathcal{A}_k^2\}_{k\in\mathbb{N}}$ *be two families of probabilistic automata; let* $\mathcal{R}$= $\{\mathcal{R}_k\}_{k\in\mathbb{N}}$ *be a family of relations such that each* $\mathcal{R}_k$ *is a relation from* S_k^1 *to* S_k^2*; let Poly be the set of positive polynomials over* $\mathbb{N}$*. We say that* $\mathcal{R}$ *is a* state polynomially accurate simulation *from* $\{\mathcal{A}_k^1\}_{k\in\mathbb{N}}$ *to* $\{\mathcal{A}_k^2\}_{k\in\mathbb{N}}$ *if*

1. *for each* $k \in \mathbb{N}$*, it holds that* $\bar{s}_k^1 \; \mathcal{R}_k \; \bar{s}_k^2$*;*
2. *for each* $c \in \mathbb{N}$ *and* $p \in Poly$*, there exists* $\bar{k} \in \mathbb{N}$ *such that for each* $k > \bar{k}$*, for all probability measures* μ_1 *and* μ_2 *and for each* $\gamma \geqslant 0$*,*
   - *if* μ_1 *is reached within* $p(k)$ *steps in* $\mathcal{A}_k^1$ *and* $\mu_1 \; \mathcal{L}(\mathcal{R}_k, \gamma) \; \mu_2$*,*
   - *then there exists a* γ*-weighting function* w_γ *for* $\mu_1 \; \mathcal{L}(\mathcal{R}_k, \gamma) \; \mu_2$ *such that* $\sum\{w_\gamma(s_1, s_2) \mid s_1 \neg \; St(\mathcal{R}_k, k^{-c}) \; s_2\} < k^{-c}$*.*

We write $\{\mathcal{A}_k^1\}_{k\in\mathbb{N}} \lesssim_s \{\mathcal{A}_k^2\}_{k\in\mathbb{N}}$ *if there exists a state polynomially accurate simulation* $\mathcal{R}$ *from* $\{\mathcal{A}_k^1\}_{k\in\mathbb{N}}$ *to* $\{\mathcal{A}_k^2\}_{k\in\mathbb{N}}$*.*

Remark 3. In the definition of the step condition, we use the same bound k^{-c} for two different quantities: the first one is the error ε of the ε-step condition, while the second one is the overall weight η of pair of states that do not satisfy the ε-step condition. The choice of using the same value is not restrictive: if we use two different bounds, say $k^{-c'}$ and $k^{-c''}$, then we can take $c = \min(c', c'')$ and the step condition still holds. In fact, it is easy to observe that $s_1 \neg \; St(\mathcal{R}_k, k^{-c}) \; s_2$ implies $s_1 \neg \; St(\mathcal{R}_k, k^{-c'}) \; s_2$ and thus $\sum\{w_\gamma(s_1, s_2) \mid s_1 \neg \; St(\mathcal{R}_k, k^{-c'}) \; s_2\} < k^{-c''}$ implies $\sum\{w_\gamma(s_1, s_2) \mid s_1 \neg \; St(\mathcal{R}_k, k^{-c}) \; s_2\} < k^{-c}$.

Notational convention. To simplify the notation, when it is clear from the context we denote the family of automata $\{\mathcal{A}_k\}_{k\in\mathbb{N}}$ by $\mathcal{A}$. We still use $\mathcal{A}_k$ to denote the automaton of the family $\{\mathcal{A}_k\}_{k\in\mathbb{N}}$ instantiated with the specific security parameter k. Similarly, we denote the family of relations $\{\mathcal{R}_k\}_{k\in\mathbb{N}}$ by $\mathcal{R}$.

Remark 4. We do not compare explicitly sPAS with the polynomially accurate simulations of [19] here due to lack of space. For the reader familiar with [19] we observe that assuming Definition 3 it is possible to prove that for any measures $\mu_1 \; \mathcal{L}(\mathcal{R}, \gamma) \; \mu_2$, any hyper-transition $\mu_1 \longrightarrow \rho_1$ can be matched by a hyper-transition $\mu_2 \longrightarrow \rho_2$ such that $\rho_1 \; \mathcal{L}(\mathcal{R}, \gamma + \varepsilon + \eta) \; \rho_2$. This is the basic step to show that sPASs are stronger than the PASs of [19].

Remark 5. It is immediate to observe that an ordinary simulation relation $\mathcal{R}$ is a special case of a sPAS since $s_1 \; \mathcal{R} \; s_2$ implies $s_1 \; St(\mathcal{R}, k^{-c}) \; s_2$. Thus, in a complex system it is safe to use ordinary simulations and basic concurrency theory on all those components that do not rely on any computational assumption.

Remark 6. Here we are working with relations that in concurrency theory literature are known as *strong*. However, the reader familiar with weak relations may note that extending our result is not difficult: just modify the condition of the step condition imposing that $\sum\{w_\gamma(s_1, s_2) \mid s_1 \neg \; St^w(\mathcal{R}_k, k^{-c}) \; s_2\} < k^{-c}$ where $s_1 \; St^w(\mathcal{R}_k, k^{-c}) \; s_2$ if for each transition (s_1, a, μ_1), there exists a weak transition (s_2, a, μ_2) such that $\mu_1 \; \mathcal{L}(\mathcal{R}_k, k^{-c}) \; \mu_2$. In order to preserve polynomial bounds, we require that also the length of the weak transition is bounded. There are several ways to define such bounds and they are proposed in [21].

We conclude this section with the statement of the main compositionality result for sPASs, which is the goal of the definition presented in this paper.

Theorem 1. *Let* $\mathcal{A}^1$ *and* $\mathcal{A}^2$ *be two families of automata. For each context* $\mathcal{C}$ *compatible with both* $\mathcal{A}^1$ *and* $\mathcal{A}^2$*, if* $\mathcal{A}^1 \lesssim_s \mathcal{A}^2$ *then* $\mathcal{A}^1||\mathcal{C} \lesssim_s \mathcal{A}^2||\mathcal{C}$*.*

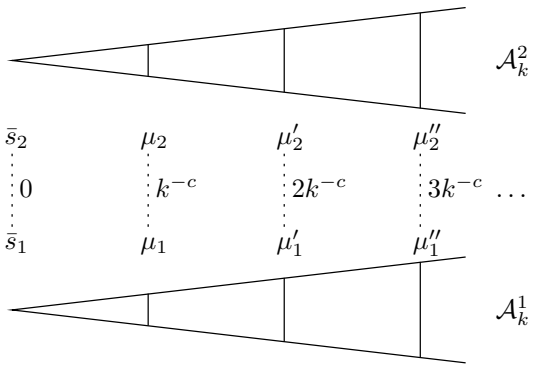

Fig. 1. Graphical representation of the Execution Correspondence Theorem

3.2 Execution Correspondence

The existence of a sPAS from $\mathcal{A}^1$ to $\mathcal{A}^2$ implies a strong correspondence between their probabilistic executions. The formal result is known in concurrency theory as the execution correspondence theorem [17], is known within cryptography in a restrictive form as the mapping lemma [13], and is the basis for reasoning about properties that relate concrete and abstract systems. Informally, the execution correspondence theorem for sPASs states that for every probabilistic execution of polynomial length of the concrete system there exists a corresponding probabilistic execution of the abstract system up to a negligible error.

Definition 6. *Given two families of automata $\mathcal{A}^1$ and $\mathcal{A}^2$ and a family $\mathcal{R}$ of relations from states of $\mathcal{A}^1$ to states of $\mathcal{A}^2$, we say that $\mathcal{A}^1$ is* execution-related by $\mathcal{R}$ *to $\mathcal{A}^2$, denoted by $\mathcal{A}^1$ $Ex(\mathcal{R})$ $\mathcal{A}^2$, if for each $c \in \mathbb{N}$, $p \in Poly$, there exists $\bar{k} \in \mathbb{N}$ such that for each $k > \bar{k}$ and each scheduler σ_1 for $\mathcal{A}_k^1$, if μ_1 is the probability measure induced by σ_1 after n steps, $n \leqslant p(k)$, then there exists a scheduler σ_2 for $\mathcal{A}_k^2$ that reaches, after n steps, a probability measure μ_2 such that μ_1 $\mathcal{L}(\mathcal{R}_k, nk^{-c})$ μ_2.*

We say alternatively that $\mathcal{R}$ is an execution relation *from $\mathcal{A}^1$ to $\mathcal{A}^2$.*

Figure 1 represents graphically the execution correspondence theorem. Given a probabilistic execution of automaton $\mathcal{A}_k^1$, let $\mu_1, \mu_1', \ldots$ be the measures reached respectively after $1, 2, \ldots$ steps. Then we can find a probabilistic execution of $\mathcal{A}_k^2$, with measures after $1, 2, \ldots$ being $\mu_2, \mu_2', \ldots$, respectively, such that the matching error grows linearly in the number of steps in the value k^{-c}, which can be made arbitrarily small by increasing the value of the security parameter k.

Theorem 2 (The Execution Correspondence). *If $\mathcal{R}$ is a state polynomially accurate simulation from $\mathcal{A}^1$ to $\mathcal{A}^2$, then $\mathcal{A}^1$ $Ex(\mathcal{R})$ $\mathcal{A}^2$.*

Proof (outline). The proof is a classical inductive argument on the number of steps: the base case follows directly from the condition on start states, while the

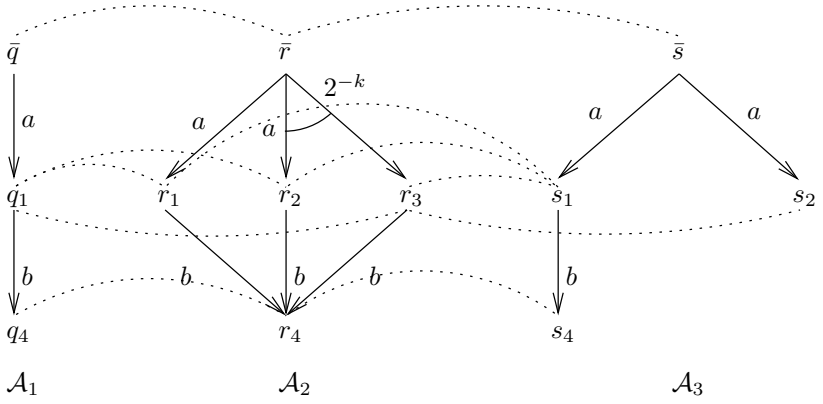

Fig. 2. Naive composition of sPAS does not lead to sPAS.

inductive step is based on the following result: if $\sum\{w_\gamma(s_1, s_2) \mid s_1 \neg\ St(\mathcal{R}, \varepsilon)\ s_2\} < \varepsilon$, and performing a step from μ_1 we reach measure φ_1, then there exists φ_2 such that $\varphi_1\ \mathcal{L}(\mathcal{R}, \varepsilon + \gamma)\ \varphi_2$ and φ_2 is reached performing a step from μ_2, that is, the following step adds an extra error that is bounded by ε (recall the discussion about the ε-η-approximate simulation). □

The execution-relationship is transitive.

Proposition 1. *Let $\mathcal{A}^1, \mathcal{A}^2, \mathcal{A}^3$ be three families of probabilistic automata and $\mathcal{R}^1, \mathcal{R}^2$ be two relations such that $\mathcal{A}^1\ Ex(\mathcal{R}^1)\ \mathcal{A}^2$ and $\mathcal{A}^2\ Ex(\mathcal{R}^2)\ \mathcal{A}^3$.*

Then, $\mathcal{A}^1\ Ex(\mathcal{R}^1 \circ \mathcal{R}^2)\ \mathcal{A}^3$.

Proof (outline). The proof follows directly from the definition of execution relation: fix $c \in \mathbb{N}$ and $p \in Poly$. $\mathcal{A}^1\ Ex(\mathcal{R}^1)\ \mathcal{A}^2$ implies that for each $c' \in \mathbb{N}$ and each probability measure μ_1 reached within $n \leqslant p(k)$ steps in $\mathcal{A}^1_k$ there exists a probability measure μ_2 reached within n steps in $\mathcal{A}^2_k$ such that $\mu_1\ \mathcal{L}(\mathcal{R}^1_k, nk^{-c'})\ \mu_2$. Similarly, $\mathcal{A}^2\ Ex(\mathcal{R}^2)\ \mathcal{A}^3$ implies that there exists a probability measure μ_3 reached within n steps in $\mathcal{A}^3_k$ such that $\mu_2\ \mathcal{L}(\mathcal{R}^2_k, nk^{-c'})\ \mu_3$. Fix c' to be $2c + 1$. Then we use the following property of ε-lifting: $\rho_1\ \mathcal{L}(\mathcal{R}_1, \varepsilon_1)\ \rho_2\ \mathcal{L}(\mathcal{R}_2, \varepsilon_2)\ \rho_3$ implies $\rho_1\ \mathcal{L}(\mathcal{R}_1 \circ \mathcal{R}_2, \varepsilon_1 + \varepsilon_2)\ \rho_3$ to conclude $\mu_1\ \mathcal{L}(\mathcal{R}^1_k \circ \mathcal{R}^2_k, 2nk^{-c'})\ \mu_3$ which implies $\mu_1\ \mathcal{L}(\mathcal{R}^1_k \circ \mathcal{R}^2_k, nk^{-c})\ \mu_3$ since $2k^{-2c-1} \leqslant k^{-c}$ for each $k > 1$. □

The proposition above is the key ingredient for hierarchical analysis since given a chain of sPAS-related automata each probabilistic execution of the first automaton corresponds to a probabilistic execution of the last automaton up to a negligible error. It would be even nicer to show that sPAS composes; however, this is not the case. Consider the three automata of Figure 2: $\mathcal{A}_1 \lesssim_s \mathcal{A}_2$ and $\mathcal{A}_2 \lesssim_s \mathcal{A}_3$ but the naive composition of sPAS relations (depicted by dotted arcs) leads to a relation that does not satisfy conditions of sPAS. In fact, q_1 is related to s_2 (through r_3) but the transition (q_1, b, δ_{q_4}) can not be matched from s_2, thus the step condition can not be satisfied for δ_{q_1} and δ_{s_2}.

$\mathcal{NG}_k^h(\mathbb{A})$

Signature:

Input: $get\_nonce(A)$, $A \in \mathbb{A}$

Output: $ret\_nonce(A, n)$, $n \in \{0,1\}^k$, $A \in \mathbb{A}$

State:

$value_A \in \{0,1\}^k \cup \{\bot\}$, initially $\bot$, $A \in \mathbb{A}$

$is\_fresh_A \in \{T, F, \bot\}$, initially $\bot$, $A \in \mathbb{A}$

$frnonces \subseteq \{0,1\}^k$, initially $\{0,1\}^k$

Transitions:

Input $get\_nonce(A)$

Effect:

$value_A := v$ where $v \in_R \{0,1\}^k$

$$is\_fresh_A := \begin{cases} T & \text{if } v \in frnonces \\ F & \text{otherwise} \end{cases}$$

$frnonces := frnonces \setminus \{v\}$

Output $ret\_nonce(A, n)$

Precondition:

$n = value_A$

Effect:

$value_A := \bot$

$is\_fresh_A := \bot$

Fig. 3. The Nonce Generator $\mathcal{NG}_k^h(\mathbb{A})$

4 Conditional Automata

Consider a public key encryption box that satisfies the *indistinguishability under chosen ciphertext attack* (IND-CCA) property. The box generates random keys and encrypts/decrypts messages. One property of IND-CCA cryptosystems is that repeated keys or repeated ciphertexts are generated with negligible probability. Consider now a more complex system that interacts with the cryptographic box. Can we assume, while analyzing the complex system, that the cryptographic box never generates any repeated key/ciphertext? Conditional automata, together with the compositionality properties of state polynomially accurate simulations, provide us with a positive answer.

We illustrate conditional automata with yet a simpler example. Consider a system that uses nonces generated by a nonce generator that simply draws random numbers of k bits, where k is the security parameter. Can we assume safely that all nonces are different? Again, the answer is positive.

To model this nonce generator, we define an automaton $\mathcal{NG}_k(\mathbb{A})$, where $\mathbb{A}$ is a set of agents, with actions to get and return nonces, respectively. In particular, each time an agent A requires a nonce, the automaton generates a number of k bits that is returned to A. Then we extend this automaton to an automaton $\mathcal{NG}_k^h(\mathbb{A})$ by adding some bookkeeping variables that simply keep track of the past and permits it to determine whether any nonce is repeated. We follow the approach based on history variables [2], for which it is known already [12] that $\mathcal{A} \preceq \mathcal{A}^h$ whenever $\mathcal{A}^h$ is obtained from $\mathcal{A}$ by adding history variables.

Figure 3 depicts $\mathcal{NG}_k^h(\mathbb{A})$. Variables $frnonces$ and is_fresh_A are history variables. The former keeps all values that are not yet chosen as nonces; the latter, one for each agent A, takes value F if the chosen nonce is repeated.

Our next step is to remove from $\mathcal{NG}_k^h(\mathbb{A})$ all those behaviors that lead to repeated nonces. We call the resulting automaton the ideal nonce generator. For the purpose, let G be the set of states of $\mathcal{NG}_k^h(\mathbb{A})$ where no nonce is repeated. We modify the transition relation of $\mathcal{NG}_k^h(\mathbb{A})$ by imposing that states outside G can not be reached by any transition. More precisely, we replace the target measure of each transition by the same measure conditional on G.

Definition 7 (Conditional Automaton). *For a PA $\mathcal{A} = (S, \bar{s}, A, D)$ and $G \subseteq S$, define the* G-conditional *of $\mathcal{A}$, denoted by $\mathcal{A}|G$, to be the PA $\mathcal{A}' = (S, \bar{s}, A, D')$ where $D' = \{(s, a, \mu|G) \mid (s, a, \mu) \in D, \mu(G) > 0\}$.*

At this point we may believe that in $\mathcal{A}|G$ no state outside G is reachable. This is indeed the case provided that the start state of $\mathcal{A}$ is in G.

Proposition 2. *Given a PA $\mathcal{A}$ and $G \subseteq S$. If $\bar{s} \in G$, then all reachable states of $\mathcal{A}|G$ are in G.*

Returning to the nonce generator example, it is immediate to verify that the automaton $\mathcal{NG}_k^h(\mathbb{A})|G$, where G is the set of states where no variable is_fresh_A has value F, is the automaton where the assignment to variable $value_A$ in action $get\_nonce(A)$ is replaced by $value_A := v$ where $v \in_R frnonces$. The last piece that is still missing from the picture is a result that allows us to conclude quickly that the ideal nonce generator is a safe abstraction of the nonce generator that can be used for further analysis.

Indeed, we turn now to our main result, namely, that the conditional automaton construction is sound for sPASs whenever we rule out states that are reachable with negligible probability on computations of polynomial length. We need some preliminary definitions.

For a set F of finite execution fragments, let $Cones(F)$ denote the set $\cup_{\alpha \in F} C_\alpha$. For a set of states B let $\Diamond(B)$ be the set of finite executions whose last state is in B, and let $\Diamond_l(B)$ be the set of executions of $\Diamond(B)$ of length at most l. Thus $Cones(\Diamond(B))$ denotes the event of reaching a state from B, while $Cones(\Diamond_l(B))$ denotes the event of reaching a state from B within l steps.

Definition 8. *Given a probabilistic automaton $\mathcal{A}$ and a set of states B, we say that B is* reachable with probability less than p in $\mathcal{A}$ *if* $\sup_\sigma\{\nu_\sigma(Cones(\Diamond(B)))\} < p$ *and that B is* reachable with probability less than p within l steps in $\mathcal{A}$ *if* $\sup_\sigma\{\nu_\sigma(Cones(\Diamond_l(B)))\} < p$.

Definition 9. *For a family $\mathcal{A}$ of probabilistic automata and a family B of states, we say that B is* polynomially reachable with negligible probability *in $\mathcal{A}$ (or alternatively that B is* negligible *in $\mathcal{A}$) if and only if for each $c \in \mathbb{N}$, each $p \in Poly$, there exists $\bar{k} \in \mathbb{N}$ such that for each $k > \bar{k}$ the probability to reach states of B_k within $p(k)$ steps in $\mathcal{A}_k$ is less than k^{-c}.*

Remark 7. The phrase "for each $c \in \mathbb{N}$, each $p \in Poly$, there exists $\bar{k} \in \mathbb{N}$ such that for each $k > \bar{k}$ the probability to reach states of B_k within $p(k)$ steps in $\mathcal{A}_k$ is less than k^{-c}" may appear to be complex; however it is a standard way to say that something is negligible (see discussion about nonces in Section 2.2).

Theorem 3 (Conditional Automaton Theorem). *Let $\mathcal{A}$ be a family of probabilistic automata and G be a family of states such that, for each $k \in \mathbb{N}$, $\bar{s}_k \in G_k$. For each $k \in \mathbb{N}$ let B_k be the set $S_k \setminus G_k$.*

Then the family B is negligible in $\mathcal{A}$ if and only if the family of identity relations is a state polynomially accurate simulation from $\mathcal{A}$ to $\mathcal{A}|G$.

Proof (outline). Negligibility of B imposes a bound on the error between each probability measure μ and $\mu|G$; the execution correspondence theorem bounds the probability to reach states in B (not reachable in $\mathcal{A}|G$) to be negligible. □

An immediate consequence of the conditional automaton theorem is indeed that the ideal nonce generator is a safe abstraction of the nonce generator.

Proposition 3. *Let B_k be the set of states of $\mathcal{NG}^h_k(\mathbb{A})$ where some variable is_fresh$_A$ is F. Let G_k be the set $S_k \setminus B_k$. Then $\mathcal{NG}^h(\mathbb{A}) \lesssim_s \mathcal{NG}^h(\mathbb{A})|G$.*

Proof (outline). Within $p(k)$ steps at most $p(k)$ nonces can be removed from *frnonces*, thus Property 1 implies that the set of states B is negligible in $\mathcal{NG}^h(\mathbb{A})$, hence $\mathcal{NG}^h(\mathbb{A}) \lesssim_s \mathcal{NG}^h(\mathbb{A})|G$. □

The exercise above can be repeated for other cryptographic primitives like encryption and digital signatures, where events like repeated keys or repeated ciphertexts are ruled out. Then, by compositionality, any complex protocol can be analyzed with the ideal versions of the underlying primitives. We do this in [21] where we reprove in a modular way the Dolev-Yao soundness result of [9].

5 Conclusion

In this paper we have introduced Conditional Automata as a tool to abstract security protocols by removing negligible events. We have provided two main theorems: the Conditional Automaton Theorem that relates the negligibility of some states to the existence of a state polynomially accurate simulation between an automaton and its G-conditional counterpart, and the Execution Correspondence Theorem that relates executions of a real protocol with executions of an idealized protocol. We have illustrated our construction via a simple example based on nonce generators; more elaborated examples are available in [21].

The conditional automaton construction turns out to be very simple, as well as the statement of the conditional automaton theorem. All the difficulties lie in setting up an appropriate underlying framework that supports the theorem and inherits the important features of concurrency theory.

We believe that one of the main advantages of our result is the ability to combine in a unique framework proofs based on cryptographic assumptions with proof techniques that have been used extensively and successfully in contexts where there are several concurrent agents. Furthermore, the current framework allows us to work in "hybrid" models where some primitives are described at a concrete level, while other primitives are abstracted a la Dolev-Yao. We are planning to apply these techniques to other more elaborate case studies, as well as investigating the relationship of polynomially accurate simulations with approximated language inclusion and metrics for probabilistic systems.

References

1. Abadi, M., Gordon, A.G.: A calculus for cryptographic protocols: the spi calculus. Information and Computation 148(1), 1–70 (1999)
2. Abadi, M., Lamport, L.: The existence of refinement mappings. Theoretical Computer Science 82(2), 253–284 (1991)
3. Abadi, M., Rogaway, P.: Reconciling two views of cryptography (the computational soundness of formal encryption). In: IFIP TCS. LNCS, vol. 2000, pp. 3–22 (2000)
4. Backes, M., Pfitzmann, B., Waidner, M.: A universally composable cryptographic library. Cryptology ePrint Archive, Report 2003/015 (2003)
5. Canetti, R.: Universally composable security: A new paradigm for cryptographic protocols. In: 42nd FOCS. pp. 136–145 (2001)
6. Canetti, R., Cheung, L., Kaynar, D., Liskov, M., Lynch, N.A., Pereira, O., Segala, R.: Using Probabilistic I/O Automata to analyze an oblivious transfer protocol. Tech. Rep. 2005/452, Cryptology ePrint Archive (2005)
7. Canetti, R., Cheung, L., Kaynar, D., Liskov, M., Lynch, N.A., Pereira, O., Segala, R.: Time-bounded task-PIOAs: A framework for analyzing security protocols. In: DISC 2006. LNCS, vol. 4167, pp. 238–253 (2006)
8. Chatzikokolakis, K., Palamidessi, C.: Making random choices invisible to the scheduler. In: CONCUR '07. LNCS, vol. 4703, pp. 42–58 (2007)
9. Cortier, V., Warinschi, B.: Computationally sound, automated proofs for security protocols. In: ESOP05. LNCS, vol. 3444, pp. 157–171 (2005)
10. Kantorovich, L.V.: On the translocation of masses. Doklady Akademii Nauk SSSR 37(7-8), 227–229 (1942)
11. Lynch, N.A., Tuttle, M.R.: Hierarchical correctness proofs for distributed algorithms. In: PODC '87. pp. 137–151 (1987)
12. Lynch, N.A., Vaandrager, F.W.: Forward and backward simulations for timing-based systems. In: REX Workshop. LNCS, vol. 600, pp. 397–446 (1991)
13. Micciancio, D., Warinschi, B.: Completeness theorems for the Abadi-Rogaway logic of encrypted expressions. Journal of Computer Security 12(1), 99–129 (2004)
14. Mitchell, J.C., Ramanathan, A., Scedrov, A., Teague, V.: A probabilistic polynomial-time process calculus for the analysis of cryptographic protocols. Theoretical Computer Science 353(1), 118–164 (2006)
15. Mitra, S., Lynch, N.A.: Approximate simulations for task-structured Probabilistic I/O Automata. In: PAuL06 (2006)
16. Pfitzmann, B., Waidner, M.: A model for asynchronous reactive systems and its application to secure message transmission. In: SP '01. pp. 184–200 (2001)
17. Segala, R.: Modeling and Verification of Randomized Distributed Real-Time Systems. Ph.D. thesis, MIT (1995)
18. Segala, R.: Probability and nondeterminism in operational models of concurrency. In: CONCUR 2006. LNCS, vol. 4137, pp. 64–78 (2006)
19. Segala, R., Turrini, A.: Approximated computationally bounded simulation relations for probabilistic automata. In: 20th CSF. pp. 140–154 (2007)
20. Shoup, V.: Sequences of games: A tool for taming complexity in security proofs. Cryptology ePrint Archive, Report 2004/332 (2004)
21. Turrini, A.: Hierarchical and Compositional Verification of Cryptographic Protocols. Ph.D. thesis, University of Verona (2009), `http://www.univr.it/main?ent=catalogoaol&id=337415&page=dettaglioPubblicazione`