Towards mechanized correctness proofs for cryptographic algorithms

Axiomatization of a probabilistic Hoare style logic.

Jerry den Hartog

Department of Computer Science, University of Twente, The Netherlands
jerry.denhartog@cs.utwente.nl

Abstract. In [5] we build a formal verification technique for game based correctness proofs of cryptographic algorithms based on a probabilistic Hoare style logic [9]. An important step towards enabling mechanized verification within this technique is an axiomatization of implication between predicates which is purely semantically defined in [9]. In this paper we provide an axiomatization and illustrate its place in the formal verification technique of [5].

1 Introduction

A typical proof to show that a cryptographic construction is secure uses a reduction from the desired security notion towards some underlying hardness assumption. The security notion is usually represented as a game, in which one proves that the attacker's chance of winning the game is negligible. From a programming language perspective, these games can be thought of as programs whose behaviour is partially known, since the program typically contains invocations to an unknown function, i.e. a function for which the body is not fixed, representing an arbitrary attacker. In this context, the cryptographic reduction is a sequence of valid program transformations.

Even though cryptographic proofs based on game reductions are powerful, the price one has to pay is high: these proofs are complex, and can easily become involved and intricate. This makes the verification difficult, with subtle errors difficult to spot. Some errors may remain uncovered long after publication, as illustrated for example by Boneh and Franklin's IBE encryption scheme [3], whose cryptographic proof has been recently patched by Galindo [7].

Recently, several papers from the cryptographic community (e.g. the work of Bellare and Rogaway [1], Halevi [8], and Shoup [13]) have recognized the need to tame the complexity of cryptographic proofs. There, the need for (development of) rigorous tools to organize cryptographic proofs in a systematic way is advocated. Besides preventing subtle easily overlooked mistakes from being introduced in the proof, this precise proof development framework would also standardize the proof writing language so that proofs can be checked easily, even perhaps using computer aided verification. The proposed frameworks [1, 8, 13] provide ad-hoc formalisms to reason about the sequences of games, providing useful program transformation rules and illustrating the techniques with several cryptographic proofs from the literature.

In [5] we introduce a framework for game-based cryptographic proofs based on Hoare logic [10] by adapting and extending earlier work on Probabilistic Hoare-logic [9]. The use of the framework is illustrated with a formalized proof of security of ElGamal [6], which reduces the semantic security of the cryptosystem to the hardness of solving the (well-known) Decisional Diffie-Hellman problem [2]. An approach [11] similar to ours introduces two systems which, translated to our approach, reason about probabilistic predicates (though restricted to counting) and about indistinguishability of Hoare tripples respectively. Main differences are that on the Hoare tripples level we only cover equivalence not yet indistinguishability while, on the other hand, our approach covers the step from program to mathematical function and combines the two systems using the well established approach of Hoare logic. For both approaches mechanization is still an open issue.

Here we address an important step towards the mechanization of proofs in the framework of [5]: axiomatization of the implication relation between predicates. The Hoare rules allow reasoning about programs, however, this needs to be combined with a system for reasoning about the probabilistic predicates used to express pre and post conditions. The main result of this paper is to provide this reasoning system in the form of a calculus for implication and equivalence of predicates. The core of the calculus, besides a conservative extension result allowing classical logical reasoning, is formed by a list of axioms and a congruence result enabling equational reasoning. Hence we also refer to the calculus as an axiomatization of the probabilistic predicate logic.

2 The Basics

We shortly recall the predicates used by the probabilistic Hoare style logic pL [9]. We introduce probabilistic states Θ and the validity relation $\models$ for predicates which gives asserts that a predicate holds in a given probabilistic state.

Expressions We define integer expressions e and Boolean expressions (or 'conditions') c by:

$$e ::= n \mid x \mid e + e \mid e - e \mid e \cdot e \mid e\, \texttt{div}\, e \mid e\, \texttt{mod}\, e \mid f(e, \dots, e)$$
$$c ::= \texttt{true} \mid \texttt{false} \mid b \mid e \texttt{=} e \mid e < e \mid c \wedge c \mid c \vee c \mid \neg c \mid c \rightarrow c$$

with x is a variable of *type* integer, b a variable of type Boolean, n an integer and f a function symbol. We assume it is clear how this can be extended with additional operators and to other types and mostly leave the type of variables implicit, assuming that all variables and values are of the correct type.

Programs Probabilistic programs statements s are defined by:

$$s ::= \texttt{skip} \mid x \texttt{ := } e \mid s \texttt{ ; } s \mid \texttt{if } c \texttt{ then } s \texttt{ else } s \texttt{ fi} \mid D(e, \dots, e; x, \dots, x) \mid s \oplus_\rho s$$

where x is a *program* variable, e an expression of the right type, ; denotes sequential composition, `if` conditional choice, D a procedure call and $\oplus_\rho$ probabilistic choice. The procedure call $D(e, e'; x, y)$ causes the body $\mathtt{B}_D$ of D to be executed with e, e' as read only and x, y as read-write arguments. The read-write arguments must be distinct variables (i.e. no aliasing). In $s \oplus_\rho s'$ program s is executed with probability ρ and s' is executed with probability $1 - \rho$. Formally a program is a pair $(s, \mathtt{B})$ consisting of a program statement and the declaration giving the procedure bodies. We will assume that the intended declaration is clear from the context and not destinguish between programs and statements.

States A *deterministic state*, $\sigma \in \mathcal{S}$, is a function that assigns a value to each program variable. A *probabilistic state*, $\theta \in \Theta$ gives the probability of being in a given deterministic state. We can think of a probabilistic state as (countable) set of labelled deterministic states or as a sum $\rho_1 \cdot \sigma_1 + \rho_2 \cdot \sigma_2 + \ldots$. Here, the probability of being in the (deterministic) state σ_i is ρ_i, $i \geq 0$. The sum of all σ_i is atmost 1, with a sum less than 1 indicating a state with 'incomplete information' (e.g. caused by non-termination). For simplicity and without loss of generality we assume that each state σ occurs at most once in θ; multiple occurrences of a single state can be merged into one single occurrence by adding the probabilities, e.g. $1 \cdot \sigma$ rather than $\frac{3}{4} \cdot \sigma + \frac{1}{4} \cdot \sigma$.

Operations on states The value $\mathcal{V}(e)(\sigma)$ of an expression e in a state σ is defined as usual. A variant $\sigma[x/e]$ of a deterministic state σ is a state which only differs from σ for the variable x where it returns $\mathcal{V}(e)(\sigma)$ the value of expression e (in state σ). A variant $\theta[x/e]$ of a probabilistic state θ is obtained by taking the variant element wise, i.e. if $\theta = \rho_1 \cdot \sigma_1 + \rho_2 \cdot \sigma_2 + \ldots$ then $\theta[x/e] = \rho_1 \cdot \sigma_1[x/e] + \rho_2 \cdot \sigma_2[x/e] + \ldots$. The (partial) operations $+$ (addition), $\rho\cdot$ (scaling) on probabilistic states (which are functions to $[0, 1]$) are the addition and scaling of functions. The operation $\oplus_\rho$ (probabilistic choice) combines these two: $\theta \oplus_\rho \theta' = \rho \cdot \theta + (1 - \rho) \cdot \theta'$. Finally the operation $c?$ (conditional choice) removes the probability for states which do not satisfy c: if $\theta = \rho_1 \cdot \sigma_1 + \rho_2 \cdot \sigma_2 + \ldots$ and c is true in σ_1, σ_3 but not in σ_2, etc. then $c?\theta = \rho_1 \cdot \sigma_1 + \rho_3 \cdot \sigma_3 + \ldots$

Deterministic and probabilistic predicates Deterministic predicates $dp \in DPred$ are first order predicate logical formulas, i.e.

$$dp ::= \mathtt{true} \mid \mathtt{false} \mid b \mid e \leq e \mid dp \wedge dp \mid dp \vee dp \mid \neg\, dp \mid dp \rightarrow dp \mid \exists i : dp \mid \forall i : dp$$

Note that the conditions (boolean expression) are also deterministic predicates. With interpretation, e.g. I, to give the value of the (program and logical, non-program) variables we can check if a predicate is satisfied, denoted $I \models_d dp$. If we have a distribution instead of the value of program variables, e.g. in a probabilistic state, we get a probability that the predicate holds. We use $\mathbb{P}(dp)$ to denote this probability and around it build a new type of expressions, the *probabilistic expressions* (e_r), with range $[0, 1]$:

$$e_r ::= \rho \mid \mathtt{r} \mid \mathbb{P}(dp) \mid e_r + e_r \mid e_r - e_r \mid e_r * e_r \mid e_r / e_r$$

where ρ is a number in $[0,1]$ and $\mathtt{r}$ a variable over this domain. A probabilistic state provides the distribution of the program variables with which we evaluate $\mathbb{P}(dp)$. For logical variables such as $\mathtt{r}$ we still have a standard interpretation which simply provides the value (i.e. a number in $[0,1]$).

Example 1. The expression $\mathtt{r} + \mathbb{P}(x > 2)$ has value $\frac{3}{4}$ for interpretation I with $I(r) = \frac{1}{4}$ and state $\frac{1}{4} \cdot [x = 1] + \frac{1}{4} \cdot [x = 2] + \frac{1}{4} \cdot [x = 3] + \frac{1}{4} \cdot [x = 4]$.

$\frac{1}{2} \cdot x$ is not a valid expression as program variables are allowed only within in the $\mathbb{P}(\cdot)$ construct.

Probabilistic predicates $p, q \in \mathit{Pred}$ are basically first order predicate formulas and the combination of such predicates using (logical and arithmetical) operators:

$$\begin{aligned} p ::= {} & \mathtt{true} \mid \mathtt{false} \mid b \mid e \leq e \mid p \wedge p \mid p \vee p \mid \neg p \mid p \rightarrow p \mid \exists j : p \mid \forall j : p \\ & \mid \rho \cdot p \mid p + p \mid p \oplus_\rho p \mid c?p \end{aligned}$$

Note that the type of the expression e can be any of the types introduced, including the probabilistic expressions.

Example 2. A common basic predicate $\mathbb{P}(x = \mathtt{1}) = \mathtt{r}$ states that probability of a deterministic predicate, in this case $x = \mathtt{1}$ holding is equal to $\mathtt{r}$.

The predicate $\forall i, j : \mathbb{P}(x = i \wedge y = j) = \mathbb{P}(x = i) \cdot \mathbb{P}(y = j)$ states that x and y are independent.

Given a probabilistic state and an interpretation of the logical variables we can check if the state satisfies a predicate, $(\theta, I) \models p$, or simply $\theta \models p$, again omitting the interpretation I from the notation. The interpretation of comparison of expressions and the logic operators are standard. The arithmetical operators are the logical counterparts of the same operations on states. We have:

$$\begin{aligned} \theta \models \rho \cdot p & \text{ when there exists } \theta'\text{: } \theta = \rho \cdot \theta', \theta \models p \\ \theta \models p + p' & \text{ when there exists } \theta_1, \theta_2\text{: } \theta = \theta_1 + \theta_2, \theta_1 \models p \text{ and } \theta_2 \models p' \\ \theta \models p \oplus_\rho p' & \text{ when there exists } \theta_1, \theta_2\text{: } \theta = \theta_1 \oplus_\rho \theta_2, \theta_1 \models p \text{ and } \theta_2 \models p' \\ \theta \models c?p & \text{ when there exists } \theta'\text{: } \theta = c?\theta', \theta \models p \end{aligned}$$

Note that if a predicate *does not use the* $\mathbb{P}(\cdot)$ *function nor the arithmetical operators*, we do not need the probabilistic state to check if the predicate is satisfied (only the interpretation of the logical variables is needed). We call such predicates $\mathbb{P}$-*free*.

Example 3. The predicate $(\mathbb{P}(x = \mathtt{1}) = \frac{1}{4}) + (\mathbb{P}(x > \mathtt{2}) = \mathtt{r})$ is true in state $\frac{1}{4} \cdot [x = 1] + \frac{1}{4} \cdot [x = 2] + \frac{1}{4} \cdot [x = 3] + \frac{1}{4} \cdot [x = 4]$ with interpretation I, $I(r) = \frac{3}{4}$, because we can split the state into $\frac{1}{4} \cdot [x = 1]$, which satisfies $\mathbb{P}(x = \mathtt{1}) = \frac{1}{4}$, and $\frac{1}{4} \cdot [x = 2] + \frac{1}{4} \cdot [x = 3] + \frac{1}{4} \cdot [x = 4]$ which satisfies $\mathbb{P}(x > \mathtt{2}) = \mathtt{r}$.

Predicate $(\mathbb{P}(x = \mathtt{1}) = \frac{1}{4}) + (\mathbb{P}(x < \mathtt{3}) = \mathtt{r})$ is false in this state and I; there is no way of splitting the state in such a way that parts satisfy both predicates.

3 A probabilistic Hoare style logic.

In this section we briefly introduce the probabilistic Hoare style logic. See [9, 5] for details. *Hoare triples*, also known as *program correctness triples*, give a precondition and a postcondition for a program. A triple is called valid, denoted $\models \{\,p\,\}\ s\ \{\,q\,\}$, if the precondition guarantees the postcondition after execution of the program.

Our derivation system for Hoare triples adapts and extends the existing Hoare logic calculus. The rules for skip, assignment, sequential composition, precondition strengthening and postcondition weakening and procedure calls are standard. The rule for conditional choice is adjusted and a new rule for probabilistic choice is added, along with some structural rules. We only present the main rules here (see e.g. [9] for a complete overview), noting that the other rules come directly from Hoare logic or from natural deduction.

$$\frac{}{\{\,p[x/e]\,\}\ x \mathrel{\texttt{:=}} e\ \{\,p\,\}}\ \text{(Assign)} \qquad \frac{\{\,c?p\,\}\ s\ \{\,q\,\} \quad \{\,\neg c?p\,\}\ s'\ \{\,q'\,\}}{\{\,p\,\}\ \texttt{if}\ c\ \texttt{then}\ s\ \texttt{else}\ s'\ \texttt{fi}\ \{\,q+q'\,\}}\ \text{(If)}$$

$$\frac{\{\,p\,\}\ s\ \{\,p'\,\} \quad \{\,p'\,\}\ s'\ \{\,q\,\}}{\{\,p\,\}\ s\ ;\ s'\ \{\,q\,\}}\ \text{(Seq)} \qquad \frac{\{\,p\,\}\ s\ \{\,q\,\} \quad \{\,p\,\}\ s'\ \{\,q'\,\}}{\{\,p\,\}\ s \oplus_\rho s'\ \{\,q \oplus_\rho q'\,\}}\ \text{(Prob)}$$

$$\frac{\{\,p\,\}\ s\ \{\,q\,\} \quad \{\,p\,\}\ s\ \{\,q'\,\}}{\{\,p\,\}\ s\ \{\,q \wedge q'\,\}}\ \text{(And)} \qquad \frac{p' \Rightarrow p \quad \{\,p\,\}\ s\ \{\,q\,\} \quad q \Rightarrow q'}{\{\,p'\,\}\ s\ \{\,q'\,\}}\ \text{(Cons)}$$

$$\frac{\{p\}\ \texttt{B}_D\ \{q\}}{\{p[{}^{\texttt{e}_1,\ldots,\texttt{e}_n,\texttt{x}_1,\ldots,\texttt{x}_n}/{}_{\texttt{v}_1,\ldots,\texttt{v}_n,\texttt{w}_1,\ldots,\texttt{w}_m}]\}\ D(\texttt{e}_1,\ldots,\texttt{e}_n;\texttt{x}_1,\ldots,\texttt{x}_n)\ \{q[{}^{\texttt{x}_1,\ldots,\texttt{x}_n}/{}_{\texttt{w}_1,\ldots,\texttt{w}_m}]\}}$$

Note the use of the implication operation $\Rightarrow$ in the (Cons) rule. This operation is formally defined in the next section which also discusses the axiomatization of this operation. After treating reasoning about (implication between) probabilistic predicates in the next section we apply the Hoare rules in the example derivation in section 5.

4 A calculus for probabilistic predicates.

The important 'rule of consequence' (Cons) in the Hoare style logic allows strengthening of the precondition and weakening of the postcondition. To apply this rule we need to determine which implications are valid. In this section we provide results allowing reasoning about equivalence of predicates and the implication between predicates. These results consist of a *conservative extension* result, allowing the use of standard first order reasoning methods, a *congruence* result allowing equational reasoning and a list of equivalences capturing the arithmetical operators, usable as *axioms* in the equational reasoning.

The key relation in our 'calculus' is equivalence $\equiv$ of predicates.

Definition 4. *We write $p \Rightarrow p'$ when for all θ, I if $(\theta, I) \models p$ then $(\theta, I) \models p'$. and $p \equiv p'$ if $p \Rightarrow p'$ and $p' \Rightarrow p$.*

The calculus is sound with respect to this semantics of $\Rightarrow$ (for proofs see the report version of this paper). Other issues such as completeness and decidability are discussed in section 6. We first present the congruence results for $\Rightarrow$.

Lemma 5 (Congruence). *If $p \Rightarrow p'$ then $op\, p \Rightarrow op\, p'$ for $op \in \{\,\exists i :, \forall i :, \rho\cdot, c?\,\}$ and $\neg\, p' \Rightarrow \neg\, p$.*

If $p \Rightarrow p'$ and $q \Rightarrow q'$ then $p\; op\; q \Rightarrow p'\; op\; q'$ for $op \in \{\,\wedge, \vee, +, \oplus_\rho\,\}$ and $p' \rightarrow q \Rightarrow p \rightarrow q'$.

As a direct consequence we also have that $\equiv$ is a congruence for all operators.

4.1 Non-probabilistic reasoning

The interpretation of the logical constructions is standard. If we consider the probabilistic and arithmetical constructions to be 'black boxes' we can do classical reasoning. To be more precise, for a fixed state θ the construct $\mathbb{P}()$ is just another function symbol in the probabilistic (i.e. real valued) expressions. When reasoning non-probabilistically we ignore the state θ. Thus the exact function represented by $\mathbb{P}()$ is not known, only that it is some function to $[0, 1]$. Similarly any arithmetical construct, e.g. $p+q$, can be seen as a black box, i.e. an unknown function which describes a boolean.

Definition 6. *We use $dp()$ to denote the* context *in which the logical variables $i_1, \ldots, i_n$ occurring in dp have been replaced by open places (denoted $\sqcup_1, \ldots, \sqcup_n$) and $dp(j_1, \ldots, j_n)$ for the predicate obtained by substituting $j_1, \ldots, j_n$ in the open places. We introduce a fresh n-ary function symbol $r_{dp()}$ to denote an unknown function to $[0, 1]$. Similarly, $p()$ denotes the* context *obtained from p and $b_{p()}$ a fresh boolean function symbol.*

Using the function symbols above we define 'black box interpretations' for expressions and probabilistic predicates. We obtain expression $BB(e)$ from expression e by replacing each occurrence of $\mathbb{P}(dp)$ by the corresponding function $r_{dp}(i_1, \ldots, i_n)$. A black box interpretation $BB(p)$ of a probabilistic predicate p is a deterministic predicate satisfying:

$$\begin{array}{ll}
BB(p) = b_{p()} & \text{or} \\
BB(p) = p & \text{for } p \;\; \mathbb{P}\text{-free, or} \\
BB(p) = BB(e) \le BB(e') & \text{for } p = \mathtt{e} \le \mathtt{e}'\text{, or} \\
BB(p) = BB(q)\; op\; BB(q') & \text{for } p = q\; op\; q',\; op \in \{\,\wedge, \vee, \rightarrow\,\}\text{, or} \\
BB(p) = op\, BB(q) & \text{for } p = op\; q,\; op \in \{\,\forall i, \exists i, \neg\,\}
\end{array}$$

Lemma 7. *If $I \models_d BB(p)$ then for any θ we have $(I, \theta) \models p$.*
If $\models_d BB(p) \rightarrow BB(q)$ then $p \Rightarrow q$.
If $\models_d BB(p) \leftrightarrow BB(q)$ then $p \equiv q$.

The intuition behind this result is as follows: The validity of $BB(p)$ is obtained irrespective of the value given by the functions $r_{dp()}$ and $b_{p()}$. By fixing θ we only select one possible function for which the predicate is true.

Example 8. (i) $i = j \;\Rightarrow\; \mathbb{P}(x = i) = \mathbb{P}(x = j)$ can be derived by noting: $BB(\mathbb{P}(x = i) = \mathbb{P}(x = j)) = (r_{X=\sqcup}(i) = r_{X=\sqcup}(j))$ and $\models_d (i = j) \rightarrow (r_{X=\sqcup}(i) = r_{X=\sqcup}(j))$.

(ii) $p + q \wedge p + q \;\equiv\; p + q$ can be derived by noting: $BB(p + q \wedge p + q) = BB(p{+}q) \wedge BB(p{+}q) = b_{p+q()}() \wedge b_{p+q()}()$ and $\models_d b_{p+q()}() \wedge b_{p+q()}() \leftrightarrow b_{p+q()}()$.

(iii) We cannot derive $(p \wedge p) + q \;\equiv p + q$ directly using lemma 7 as we cannot 'look into' the black box $(p \wedge p) + q$. However, we can first note that $p \wedge p \equiv p$ and then apply the congruence result.

Thus the non-probabilistic part of the reasoning is standard. We assume the reader is familiar with ways of formalizing such reasoning and we will be less precise in this part of the reasoning.

4.2 The axioms

To deal with the arithmetical operators we provide a list of equivalences that can be used as an axiomatic basis of equational reasoning. The equivalences provide basic properties and distributivity laws for each of the operators. We do not treat the operator $\oplus_\rho$ as $p \oplus_\rho p'$ is equivalent to $\rho \cdot p + (1 - \rho) \cdot p'$. For validity proofs of the axioms we refer to the technical report version of this paper.

In the axioms we will want to state things about validity of deterministic predicates (e.g. $dp \rightarrow dp'$). However, deterministic predicate may contain program variables which are not allowed in probabilistic predicates (outside of a $\mathbb{P}()$ construction). In this case by validity we mean that the deterministic predicate must hold, no matter which value the program variables have. We introduce the new notation $\Box(dp)$ capturing this notion of validity.

Definition 9. *Let $x_1, \ldots x_n$ be the program variables occurring in dp and let $j_1, \ldots j_n$ be fresh logical variables (i.e. not occurring in dp). Then*

$$\Box(dp) \;::=\; \forall j_1, \ldots j_n : dp[j_1/x_1, \ldots j_m/x_m]$$

Note that $\Box(dp)$ is a $\mathbb{P}$-free predicate which holds exactly when dp is fulfilled by every deterministic state σ, i.e. $(\theta, I) \models \Box(dp)$ iff $\forall \sigma : (\sigma, I) \models_d dp$.

Probabilistic axioms The following axioms capture basic properties of probabilistic states.

$$\mathbb{P}(\texttt{false}) = 0 \equiv \texttt{true} \tag{A1}$$

$$\mathbb{P}(\texttt{true}) \leq 1 \equiv \texttt{true} \tag{A2}$$

$$\mathbb{P}(dp \vee dp') = \mathbb{P}(dp) + \mathbb{P}(dp') - \mathbb{P}(dp \wedge dp') \equiv \texttt{true} \tag{A3}$$

$$\Box(dp \rightarrow dp') \Rightarrow \mathbb{P}(dp) \leq \mathbb{P}(dp') \tag{A4}$$

The first three axioms state that the given equations on chances are tautologies. They follow directly from the properties of (sub-)probability measures. Recall that incomplete states (e.g. caused by non-termination) may satisfy $\mathbb{P}(\texttt{true}) < 1$. The last axiom lifts reasoning on deterministic predicates to probabilistic predicates: If $dp \rightarrow dp'$ must hold then the probability $\mathbb{P}(dp)$ of dp cannot be more than the probability $\mathbb{P}(dp')$ of dp'. (Note that any axiom of the form $p \Rightarrow q$ could be equivalently written as $p \equiv p \wedge q$ thus still fits within the equational system.)

Example 10. (i) We obtain

$$\Box(dp \leftrightarrow dp') \;\Rightarrow\; \mathbb{P}(dp) = \mathbb{P}(dp')$$

directly from A4.

(ii) We have

$$\mathbb{P}(dp) = \mathbb{P}(\mathtt{true}) - \mathbb{P}(\neg dp) \quad \equiv \quad \mathtt{true}$$

In the derivation we streamline the notation slightly, writing $e_{r1} = e_{r2} = \ldots = e_{rn}$ rather than $\mathtt{true} \;\equiv\; (e_{r1} = e_{r2}) \;\equiv\; \ldots \;\equiv\; (e_{r1} = e_{r2}) \wedge \ldots \wedge (e_{rn-1} = e_{rn}) \;\equiv\; (e_{r1} = e_{rn})$. Using this notation gives

$$\begin{aligned}\mathbb{P}(dp')\ [\text{A3}] &= \mathbb{P}(dp \wedge \neg dp') + \mathbb{P}(dp \vee \neg dp) - \mathbb{P}(\neg dp)\\ [\text{ex. } 10(\text{i})] &= \mathbb{P}(\mathtt{false}) + \mathbb{P}(\mathtt{true}) - \mathbb{P}(\neg dp)\\ [\text{A1}] &= 0 + \mathbb{P}(\mathtt{true}) - \mathbb{P}(\neg dp)\\ &= \mathbb{P}(\mathtt{true}) - \mathbb{P}(\neg dp)\end{aligned}$$

Axioms for · The following two sets of axioms capture the behaviour of the '·'-operator. We first present basic axioms followed by distributivity laws.

$$\rho \cdot (\mathbb{P}(dp) = r) \equiv \mathbb{P}(dp) = \rho \cdot r \wedge \mathbb{P}(\mathtt{true}) \leq \rho \tag{A5}$$

$$\rho \cdot (\rho' \cdot p) \equiv (\rho \cdot \rho') \cdot p \tag{A6}$$

$$\rho \cdot p \equiv p \wedge \mathbb{P}(\mathtt{true}) \leq \rho \qquad \text{if } p \text{ is } \mathbb{P}\text{-free} \tag{A7}$$

Axiom A5 characterizes the · operator: The probability of all events is scaled. The probability of the event dp becomes $\rho \cdot r$ and no event can have a probability greater than $\rho \cdot 1$ do to the scaling. The second axiom states that first scaling with ρ' and then with ρ is the same as scaling directly with $\rho \cdot \rho'$. As a $\mathbb{P}$-free predicate does not depend on the probabilistic state, it is not influenced by scaling of this state. The scaling only affects the total probability that the state can have; after scaling it can be at most ρ.

Example 11. Applying · influences the state but not the logical variables:

$$\begin{aligned}\tfrac{1}{2} \cdot (\mathbb{P}(x = 1) = \tfrac{1}{2}) &\Rightarrow \mathbb{P}(x = 1) = \tfrac{1}{4}\\ \tfrac{1}{2} \cdot (r = \tfrac{1}{2}) &\Rightarrow r = \tfrac{1}{2}\\ \tfrac{1}{2} \cdot (\mathbb{P}(x = 1) = r \wedge r = \tfrac{1}{2}) &\Rightarrow \mathbb{P}(x = 1) = \tfrac{1}{2} r \wedge r = \tfrac{1}{2} \;\Rightarrow\; \mathbb{P}(x = 1) = \tfrac{1}{4}\end{aligned}$$

The next set of axioms capture the interplay between · and the other operators in a number of distributivity laws. The operator · distributes over the other operators in a straightforward manner. Only for + there is a complication which is explained below the rules.

$$\rho \cdot (p \; op \; p') \equiv (\rho \cdot p) \; op \; (\rho \cdot p') \tag{A8}$$
$$\rho \cdot (op\text{'}p) \equiv op\text{'}(\rho \cdot p') \tag{A9}$$
$$\rho \cdot (p + q) \equiv \exists r : \rho \cdot (p \wedge \mathbb{P}(\texttt{true}) \leq r) + \rho \cdot (q \wedge \mathbb{P}(\texttt{true}) \leq 1 - r) \tag{A10}$$

with $op \in \{\wedge, \vee, \oplus_{\rho'}\}$, $op\text{'} \in \{\exists i :, \forall i :, \rho'\cdot, c?\}$ and r a fresh variable not occurring in p or q. The scaling operator $\cdot$ distributes straightforwardly over all other operators except $+$ for which there is a complication: If θ satisfies p and θ' satisfies p' and $\theta + \theta'$ is a probabilistic state then this state satisfies $p + p'$. However, $\theta + \theta'$ may have a total mass greater than 1. In this case first adding and then scaling, as in e.g. $\frac{1}{2} \cdot (p + q)$ is not possible, however, if the states are first scaled then they can be added, as in $\frac{1}{2} \cdot p + \frac{1}{2} \cdot q$, without exceeding the maximum total probability of 1. Axiom A10 captures that $\rho \cdot (p + q)$ is the same as $\rho \cdot p + \rho \cdot q$ as long as the total probability of 1 is not exceeded.

Axioms for ? In giving the characterization of the ? operator we have the complication that part of the state has been removed. The probability of events will depend on the part of the state that has been removed.

$$\begin{aligned} c?(\mathbb{P}(dp) = r) \; &\equiv \; \mathbb{P}(\neg c) = 0 \wedge \exists r_\delta : \mathbb{P}(dp) = r - r_\delta \wedge \\ &\quad\; 0 \leq r_\delta \leq 1 - \mathbb{P}(\texttt{true}) \wedge \Box(dp \rightarrow c) \rightarrow r_\delta = 0 && \text{(A11)} \\ c?(c'?p) \; &\equiv \; (c \wedge c')?p && \text{(A12)} \\ c?p \; &\equiv \; p \wedge \mathbb{P}(\neg c) = 0 \qquad \text{if } p \text{ is } \mathbb{P}\text{-free} && \text{(A13)} \\ \texttt{true}?p \; &\equiv \; p && \text{(A14)} \\ \Box(c \leftrightarrow c') \wedge c?p &\Rightarrow c'?p && \text{(A15)} \end{aligned}$$

By removing the part of the state where c does not hold, the chance of $\neg c$ becomes 0 and the probability of dp is decreased by some amount r_δ. Clearly r_δ is at most $1 - \mathbb{P}(\texttt{true})$ which is the total amount of probability that is missing. If dp logically implies c the probability of dp cannot be decreased by removing states not satisfying c so r_δ must be 0. (Also, if dp implies $\neg c$ then all states satisfying dp will be removed giving $r_\delta = r$ but this is already implied by the fact that $\mathbb{P}(\neg c) = 0$.) The other axioms are relatively straightforward.

Example 12. (i) The total probability of $c?p$ is the probability of c in p.

$$\begin{aligned} c?(\mathbb{P}(c) = r) &\Rightarrow [A11] \; \mathbb{P}(\neg c) = 0 \wedge \exists r_\delta : \mathbb{P}(c) = r - r_\delta \wedge r_\delta = 0 \\ &\Rightarrow \mathbb{P}(\neg c) = 0 \wedge \mathbb{P}(c) = r \\ &\Rightarrow [A3] \; \mathbb{P}(c \vee \neg c) = 0 + r + 0 \\ &\Rightarrow \mathbb{P}(\texttt{true}) = r \end{aligned}$$

(ii) Similarly, if c is implied by dp then the probability of dp will not change by applying $c?$.

$$\Box(dp \rightarrow c) \wedge c?(\mathbb{P}(dp) = r) \Rightarrow [A11] \; \exists r_\delta : \mathbb{P}(dp) = r - r_\delta \wedge r_\delta = 0 \;\; \Rightarrow \;\; \mathbb{P}(dp) = r$$

The next set of axioms provide distributivity laws for ?.

$$c?(p \vee q) \equiv (c?p) \vee (c?q) \tag{A16}$$
$$c?(\exists i : p) \equiv \exists i : (c?p) \tag{A17}$$
$$c?(p + q) \equiv \exists r : c?(p \wedge \mathbb{P}(\texttt{true}) \leq r) + c?(q \wedge \mathbb{P}(\texttt{true}) \leq 1 - r) \tag{A18}$$

with i not free in c and r a fresh variable not occurring in p or q. With distributivity over $+$ we have a similar situation as for $\cdot$; $c?(p+q)$ is the same as $c?p + c?q$ as long as the total probability of 1 is not exceeded. The operator ? does not distribute over $\wedge$ (nor over $\forall i :$), $c?(p \wedge q) \not\equiv c?p \wedge c?q$, as p and q may have conflicting requirements for the part of the state which is removed by first applying the c? operator. We have to suffice with implication and a special case:

$$c?(\forall i : p) \Rightarrow \forall i : c?p \tag{A19}$$
$$c?(p \wedge q) \Rightarrow c?p \wedge c?q \tag{A20}$$
$$c?(p \wedge q) \equiv c?p \wedge c?q \qquad \text{if } p \text{ is } \mathbb{P}\text{-free} \tag{A21}$$

For a $\mathbb{P}$-free predicate the equivalence does hold as a $\mathbb{P}$-free predicate is not influenced by the ? operator. Note that a similar '$\mathbb{P}$-free-axiom' for $\forall$ already follows using (A13).

Axioms for + The following two sets of axioms capture the behaviour of the + operator. We first present basic axioms followed by distributivity laws.

$$\mathbb{P}(dp) = r + \mathbb{P}(dp') = r' \equiv r \leq \mathbb{P}(dp) \wedge r' \leq \mathbb{P}(dp') \wedge \\ \mathbb{P}(dp \wedge dp') \leq r + r' \leq \mathbb{P}(dp \vee dp') \tag{A22}$$
$$p + q \equiv q + p \tag{A23}$$
$$(p_1 + p_2) + p_3 \equiv p_1 + (p_2 + p_3) \tag{A24}$$
$$p + p' \equiv p \wedge (\texttt{true} + p') \qquad \text{if } p \text{ is } \mathbb{P}\text{-free} \tag{A25}$$

The first rule provides a characterization of the + operator: Two partial states are combined. The probability of an event cannot be less in the combined state than it already is in one of the two parts. The probability of the event $dp \wedge dp'$ is at most $r + r'$ because is probability is at most r in the left hand part and at most r' in the right hand part. Similarly the probability of $dp \vee dp'$ is at least $r + r'$ because its probability is at least r and r' in the respective parts. The second rule (commutativity) and third rule (associativity) are standard while the last rule allows moving 'non-probabilistic' properties to outside the +.

Example 13. Taking dp' equal to dp in the first rule gives

$$\begin{aligned} &\mathbb{P}(dp) = r + \mathbb{P}(dp) = r' \\ [\text{A22}] &\equiv r \leq \mathbb{P}(dp) \wedge r' \leq \mathbb{P}(dp) \wedge \mathbb{P}(dp \wedge dp) \leq r + r' \leq \mathbb{P}(dp \vee dp) \\ [\text{A4}] &\equiv r \leq \mathbb{P}(dp) \wedge r' \leq \mathbb{P}(dp) \wedge \mathbb{P}(dp) \leq r + r' \leq \mathbb{P}(dp) \\ [r, r' \geq 0] &\equiv \mathbb{P}(dp) = r + r' \end{aligned}$$

(Note that the remark $[r, r' \geq 0]$ in the last equivalence is needed only for the reverse implication.)

Finally, we provide a set of distributivity laws for $+$ which are similar to those for $c?$:

$$(p \vee p') + q \equiv (p+q) \vee (p'+q) \tag{A26}$$

$$(\exists i : p) + q \equiv \exists i : (p+q) \qquad i \text{ not free in } q \tag{A27}$$

$$(\forall i : p) + q \Rightarrow \forall i : (p+q) \qquad \text{if } i \text{ not free in } q \tag{A28}$$

$$(p \wedge p') + q \Rightarrow (p+q) \wedge (p'+q) \tag{A29}$$

$$(p \wedge p') + q \equiv (p+q) \wedge (p'+q) \qquad \text{if } p \text{ is } \mathbb{P}\text{-free} \tag{A30}$$

This completes the axiom system. In the next section we illustrate the calculus by axiomatizing the main Hoare logic derivation from the verification of ElGamal presented in [5].

5 Applying the calculus in the El-Gamal proof.

In this section we show how the calculus can be applied by treating a derivation from [5]. The proof outline in Table 1 represents this derivation which shows that the program, a transformed security game, is similar to a coin toss. The complete proof of El-Gamal security [5] uses several transformations to reach this game. We first recall some short-hand notation and results from [5].

Definition 14. *We use $I(\mathrm{e}, \mathrm{e}')$ to denote that expressions* e *and* e′ *are independent:*

$$I(\mathrm{e},\mathrm{e}') \ ::= \ \forall i,j : \mathbb{P}(\mathrm{e}=i \wedge \mathrm{e}'=j) = \mathbb{P}(\mathrm{e}=i)\cdot\mathbb{P}(\mathrm{e}'=j)$$

We use $R_{S,S'}(\mathrm{e},\mathrm{e}')$ to denote that expressions e,e' have independent uniform distributions over their respective domains S, S':

$$R_{S,S'}(\mathrm{e},\mathrm{e}') \ ::= \ \forall i,j : \mathbb{P}(\mathrm{e}=i \wedge \mathrm{e}'=j) = 1/|S| \cdot 1/|S'|$$

We assume it is clear how this can be extended to any number of expressions.

As a basic result we have that 'independent uniform distributed' variables are exactly that, independent and uniformly distributed. Also we have that if an expression is independent of the arguments of a function then it is independent of the outcome.
Lemma 15.

$$R_{S,S'}(\mathrm{e},\mathrm{e}') \equiv R_S(\mathrm{e}) \wedge R_{S'}(\mathrm{e}') \wedge I(\mathrm{e},\mathrm{e}')$$
$$I(\mathrm{e},\mathrm{e}') \Rightarrow I(\mathrm{e}, f(\mathrm{e}'))$$

The transformed security game is the program s given by the numbered lines in Table 1. For this program we derive

$$\{R_{Z_q^*3,RND,Bool}(\mathtt{v1},\mathtt{v2},\mathtt{v3},\mathtt{v4},\mathtt{v5})\} \ s \ \{\mathbb{P}(\mathtt{x1}) = \mathtt{1/2}\}$$

In otherwords, we show that given random inputs the chance of the event x_1, which represents correctly guessing which message was encoded (m_0 or m_1), is equal to half.

```
$\{R_{Z_q^*3,RND,Bool}(\mathtt{v1},\mathtt{v2},\mathtt{v3},\mathtt{v4},\mathtt{v5})\} \overset{(I1)}{\Rightarrow}$                                  (A)
$\{R_{Z_q^*3,RND,Bool}(\mathtt{v1},\mathtt{v2},\mathtt{v3}\cdot\mathtt{A0}(\mathtt{v1},\mathtt{v4}),\mathtt{v4},\mathtt{v5})$
  $\wedge\, R_{Z_q^*3,RND,Bool}(\mathtt{v1},\mathtt{v2},\mathtt{v3}\cdot\mathtt{A1}(\mathtt{v1},\mathtt{v4}),\mathtt{v4},\mathtt{v5})\}$
  $\mathbf{m0} := \mathbf{A0}(\mathbf{v1},\mathbf{v4});$                                        (1)
  $\mathbf{m1} := \mathbf{A1}(\mathbf{v1},\mathbf{v4});$                                        (2)
$p_0 \triangleq \{R_{Z_q^*3,RND,Bool}(\mathtt{v1},\mathtt{v2},\mathtt{v3}\cdot\mathtt{m0},\mathtt{v4},\mathtt{v5})$
      $\wedge R_{Z_q^*3,RND,Bool}(\mathtt{v1},\mathtt{v2},\mathtt{v3}\cdot\mathtt{m1},\mathtt{v4},\mathtt{v5})\}$
$\{R_{Z_q^*3,RND,Bool}(\mathtt{v1},\mathtt{v2},\mathtt{v3},\mathtt{v4},\mathtt{v5})\}$
  if v5 = false then                                              (3)
    $\{(\neg\mathtt{v5})?p_0\} \overset{(I2)}{\Rightarrow} \{(\neg\mathtt{v5})?R_{Z_q^*3,RND,Bool}(\mathtt{v1},\mathtt{v2},\mathtt{v3}\cdot\mathtt{m}_0,\mathtt{v4},\mathtt{v5})\}$
      tmp := v3 · m0                                              (3a)
    $p1 \triangleq \{(\neg\mathtt{v5})?R_{Z_q^*3,RND,Bool}(\mathtt{v1},\mathtt{v2},\mathtt{tmp},\mathtt{v4},\mathtt{v5})\}$
  else
    $\{(\mathtt{v5})?p_0\} \overset{(I3)}{\Rightarrow} \{(\mathtt{v5})?R_{Z_q^*3,RND,Bool}(\mathtt{v1},\mathtt{v2},\mathtt{v3}\cdot\mathtt{m}_1,\mathtt{v4},\mathtt{v5})\}$
      tmp := v3 · m1                                              (3b)
    $p2 \triangleq \{(\mathtt{v5})?R_{Z_q^*3,RND,Bool}(\mathtt{v1},\mathtt{v2},\mathtt{tmp},\mathtt{v4},\mathtt{v5})\}$
  fi
$\{p1+p2\} \overset{(I4)}{\Rightarrow} \{R_{Z_q^*3,RND,Bool}(\mathtt{v1},\mathtt{v2},\mathtt{tmp},\mathtt{v4},\mathtt{v5})\} \overset{(I5)}{\Rightarrow}$
$\{R_{Bool}(\mathtt{v5}) \wedge I(\mathtt{v5}, A2(\mathtt{v1},\mathtt{v2},\mathtt{tmp},\mathtt{v4}))\}$
  $\mathbf{b} := \mathbf{A2}(\mathbf{v1},\mathbf{v2},\mathbf{tmp},\mathbf{v4})$                                (4)
$\{R_{Bool}(\mathtt{v5}) \wedge I(\mathtt{v5},\mathtt{b})\} \overset{(I6)}{\Rightarrow} \{\mathbb{P}(\mathtt{v5}=\mathtt{b}) = 1/2\}$
  if v5 = b then                                                  (5)
    $\{(\mathtt{v5}=\mathtt{b})?(\mathbb{P}(\mathtt{v5}=\mathtt{b})=1/2)\} \overset{(I7)}{\Rightarrow} \{\mathbb{P}(\mathtt{true}) = 1/2\}$
      x1 := true                                                  (5a)
    $\{\mathbb{P}(\mathtt{x1}) = 1/2\}$
  else
    $\{(\mathtt{v5}\neq\mathtt{b})?(\mathbb{P}(\mathtt{v5}=\mathtt{b})=1/2)\} \overset{(I8)}{\Rightarrow} \{\mathbb{P}(\mathtt{false}) = 0\}$
      x1 := false                                                 (5b)
    $\{\mathbb{P}(\mathtt{x1}) = 0\}$
  fi
$\{(\mathbb{P}(\mathtt{x1}) = 1/2) + (\mathbb{P}(\mathtt{x1}) = 0)\} \overset{(I9)}{\Rightarrow}$
$\{\mathbb{P}(\mathtt{x1}) = 1/2\}$                                                        (B)
```

Table 1. Derivation of $\{R_{Z_q^*3,RND,Bool}(\mathtt{v1},\mathtt{v2},\mathtt{v3},\mathtt{v4},\mathtt{v5})\}\ s\ \{\mathbb{P}(\mathtt{x1} = \mathtt{1/2}\}$

We describe how to derive the Hoare triple in Table 1, in bottom up fashion. In the last line we use rule (Cons). To show implication ($I9$), we apply the result of example 13.

Then we apply rule (If). Implication ($I8$) follows from Axiom (A1). To show implication ($I7$), we apply the first result of example 12.

To derive implication ($I6$) we first note that from the definition of R_{Bool} we get $R_{Bool}(\mathtt{v5}) \Rightarrow \mathbb{P}(\mathtt{v5}) = \frac{1}{2} \wedge \mathbb{P}(\neg\mathtt{v5}) = \frac{1}{2} \Rightarrow \mathbb{P}(\mathtt{true}) = 1$. We then check the probability of $\mathtt{v5} = b$:

$$\begin{aligned}\mathbb{P}(\mathtt{v5} = b) &= \mathbb{P}((\mathtt{v5} \wedge b) \vee (\neg\mathtt{v5} \wedge \neg b)) \;=\; \mathbb{P}((\mathtt{v5} \wedge b)) + \mathbb{P}(\neg\mathtt{v5} \wedge \neg b) - 0\\ [I(\mathtt{v5}, b)] &= \mathbb{P}(\mathtt{v5}) \cdot \mathbb{P}(b) + \mathbb{P}(\neg\mathtt{v5}) \cdot \mathbb{P}(\neg b)\\ [R(\mathtt{v5})] &= \tfrac{1}{2} \cdot \mathbb{P}(b) + \tfrac{1}{2} \cdot \mathbb{P}(\neg b) \;=\; \tfrac{1}{2} \cdot (\mathbb{P}(b) + \mathbb{P}(\neg b)) \;=\; \tfrac{1}{2} \cdot \mathbb{P}(b \vee \neg b)\\ &= \tfrac{1}{2} \cdot \mathbb{P}(\mathtt{true}) \;=\; \tfrac{1}{2} \cdot 1 \;=\; \tfrac{1}{2}\end{aligned}$$

As the next step we use rule (Assign). Implication ($I5$) follows from Lemma 15. For implication ($I4$) we use note that (using shorthand $\rho = 1/(q^3 \cdot r \cdot 2)$) we have

$$\begin{aligned}&(\neg\mathtt{v5})?R_{Z_q^*3,RND,Bool}(\mathtt{v1}, \mathtt{v2}, \mathtt{tmp}, \mathtt{v4}, \mathtt{v5})\\ &\qquad [A11] \Rightarrow \mathbb{P}(\mathtt{v5} = \mathtt{true}) = 0\\ &\qquad [A4] \Rightarrow \mathbb{P}(\mathtt{v1} = i_1, \mathtt{v2} = i_2, \mathtt{tmp} = i_3, \mathtt{v4} = i_4, \mathtt{v5} = \mathtt{true}) = 0\\ &(\mathtt{v5})?R_{Z_q^*3,RND,Bool}(\mathtt{v1}, \mathtt{v2}, \mathtt{tmp}, \mathtt{v4}, \mathtt{v5})\\ &\qquad [\text{Def. } R] \Rightarrow (\mathtt{v5})?(\mathbb{P}(\mathtt{v1} = i_1, \mathtt{v2} = i_2, \mathtt{tmp} = i_3, \mathtt{v4} = i_4, \mathtt{v5} = \mathtt{true}) = \rho)\\ &\qquad [\text{ex. 12(ii)}] \Rightarrow \mathbb{P}(\mathtt{v1} = i_1, \mathtt{v2} = i_2, \mathtt{tmp} = i_3, \mathtt{v4} = i_4, \mathtt{v5} = \mathtt{true}) = \rho\end{aligned}$$

Combining these two facts by using the congruence lemma for + and then applying the result in example 13 gives

$$p_1 + p_2 \Rightarrow \mathbb{P}(\mathtt{v1} = i_1, \mathtt{v2} = i_2, \mathtt{tmp} = i_3, \mathtt{v4} = i_4, \mathtt{v5} = \mathtt{true}) = 0 + \rho = \rho$$

Symmetrically we also get

$$p_1 + p_2 \Rightarrow \mathbb{P}(\mathtt{v1} = i_1, \mathtt{v2} = i_2, \mathtt{tmp} = i_3, \mathtt{v4} = i_4, \mathtt{v5} = \mathtt{false}) = \rho + 0 = \rho$$

Thus

$$p_1 + p_2 \Rightarrow \forall i_5 : \mathbb{P}(\mathtt{v1} = i_1, \mathtt{v2} = i_2, \mathtt{tmp} = i_3, \mathtt{v4} = i_4, \mathtt{v5} = i_5) = \rho$$

Forall introduction for the free variables i_1, i_2, i_3, i_4 gives us implication (I4).

The following steps are straightforward from rules (If) and (Assign). Implications ($I2$) and ($I3$) are trivial.

Finally for implication ($I1$) we use the assumption that multiplication $\cdot$ has an inverse in the group. We use this assumption in the form of the following two properties:

$$\forall k, l : \exists m : k \cdot m = l \qquad \text{(Mul I)}$$
$$\forall k, l, m : (k \cdot m = l \cdot m) \;\rightarrow\; k = l \qquad \text{(Mul II)}$$

Using these assumptions we derive implication ($I1$) as follows (again using ρ as a shorthand for $1/(q^3 \cdot r \cdot 2)$)

$$\begin{array}{rl}
& R_{Z_q^*3,RND,Bool}(\mathtt{v1},\mathtt{v2},\mathtt{v3},\mathtt{v4},\mathtt{v5}) \\
& \Rightarrow \forall j : \mathbb{P}(\mathtt{v1} = i_1, \mathtt{v2} = i_2, \mathtt{v3} = j, \mathtt{v4} = i_4, \mathtt{v5} = i_5) = \rho \\
\text{[Mul I]} & \Rightarrow \forall i_3 : \exists j : j \cdot f(i_1, i_4) = i_3 \wedge \\
& \qquad\qquad \mathbb{P}(\mathtt{v1} = i_1, \mathtt{v2} = i_2, \mathtt{v3} = j, \mathtt{v4} = i_4, \mathtt{v5} = i_5) = \rho \\
\text{[Mul II]} & \Rightarrow \forall i_3 : \exists j : \Box(\mathtt{v3} = j \leftrightarrow \mathtt{v3} \cdot f(i_1, i_4) = i_3) \wedge \\
& \qquad\qquad \mathbb{P}(\mathtt{v1} = i_1, \mathtt{v2} = i_2, \mathtt{v3} = j, \mathtt{v4} = i_4, \mathtt{v5} = i_5) = \rho \\
\text{[ex.10(i)]} & \Rightarrow \forall i_3 : \exists j : \mathbb{P}(\mathtt{v1} = i_1, \mathtt{v2} = i_2, \mathtt{v3} \cdot f(i_1, i_4) = i_3, \mathtt{v4} = i_4, \mathtt{v5} = i_5) = \rho \\
& \Rightarrow \forall i_3 : \mathbb{P}(\mathtt{v1} = i_1, \mathtt{v2} = i_2, \mathtt{v3} \cdot f(\mathtt{v1}, \mathtt{v4}) = i_3, \mathtt{v4} = i_4, \mathtt{v5} = i_5) = \rho \\
& \Rightarrow R_{Z_q^*3,RND,Bool}(\mathtt{v1},\mathtt{v2},\mathtt{v3} \cdot f(\mathtt{v1},\mathtt{v4}),\mathtt{v4},\mathtt{v5})
\end{array}$$

6 Conclusions and Future work

In this paper we take an important step toward mechanizing the proofs in the methodology introduced in [5] by providing a calculus for reasoning about the validity of implication between probabilistic predicates. The usefulness of the calculus from this perspective is illustrated by showing how it can be used to replace the partly semantical reasoning of [5] in the main Hoare style derivation for the ElGamal correctness proof. The next step in the mechanization is the implementation of probabilistic predicates in a theorem proving system, such as PVS, HOL, etc. We envision three possible levels of abstraction: The first, most abstract, level is an implementation of the calculus. There is no notion of probabilistic state and reasoning consist of application of the calculus rules and use of the proof checker's built in mechanisms for reasoning about deterministic predicates. The second level introduces probabilistic states in terms of abstract functions and probabilistic properties of these functions as axioms. This allows modeling the semantics of predicates and deriving results directly from the probabilistic properties and showing the correctness of the calculus rules themselves. The final level defines probabilistic states as countable sums and uses arithmetical properties of such sums to derive results. In this way we can derive results directly from properties of the data types (such as real numbers). Of course, one can mix the levels as needed; e.g. results in a lower level can be added as axioms in a higher level. In this way we can justify results based on elemental properties while still reasoning about program at a high level of abstraction.

In addition to the implementation of probabilistic predicates in a proof checker there is the step to Hoare logic proof outlines. Checking correct application of the Hoare logic rules can be implemented in the proof checker or could be done by a pre-processor which does syntactic checks and outputs proof obligations in the form of implications to be checked in your favorite (probabilistic predicate enabled) proof checker.

We have not addressed issues such as completeness, decidability and the ability to automatically derive proofs. In the target application area we typically already have the proofs but need an intuitive way of formally expressing the properties and proofs and check them for oversights. Thus we focus on expressiveness and minimizing the step from existing proof to formalization. The framework [12] provides a well developed quantitative weakest precondition approach which allows calculating expectations. However, translating existing proofs to this setting seems to require more adaption and/or the use of meta-logical statements in the formulation of the cryptographic properties and algorithms. With reasonable restrictions it is also possible to define weakest preconditions and obtain a complete and decidable reasoning system in the probabilistic Hoare logic setting [4]. Though the size of these predicates may quickly become unmanageable for our purpose, it may be possible to use similar techniques to build decidable and complete lower levels.

Besides mechanization of the existing framework we aim at extending the techniques to different classes of cryptographic algorithms and different types of security properties. Finally, for further discussion of related work, especially in the area of verification of cryptographic protocols we refer to [5].

References

1. M. Bellare and P. Rogaway. The game-playing technique, December 2004. At `http://www.cs.ucdavis.edu/~rogaway/papers/games.html`.
2. D. Boneh. The decisional diffie-hellman problem, 1998. In Proceedings of the 3rd Algorithmic Number Theory Symposium, LNCS Vol. 1423, Springer-Verlag, 1998.
3. D. Boneh and M. K. Franklin. Identity-based encryption from the weil pairing. In *CRYPTO'01*, pages 213–229. Springer-Verlag, 2001.
4. R. Chadha, L. Cruz-Filipe, P. Mateus, and A. Sernadas. Reasoning about probabilistic sequential programs. *Theor. Comput. Sci.*, 379(1-2):142–165, 2007.
5. R. J. Corin and J. I. den Hartog. A probabilistic hoare-style logic for game-based cryptographic proofs. In M. Bugliesi, B. Preneel, and V. Sassone, editors, *ICALP 2006 track C, Venice, Italy*, volume 4052 of *Lecture Notes in Computer Science*, pages 252–263, Berlin, July 2006. Springer-Verlag.
6. T. ElGamal. A public-key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, IT-31:469–472, 1985.
7. D. Galindo. Boneh-franklin identity based encryption revisited. In *ICALP*, pages 791–802, 2005.
8. S. Halevi. A plausible approach to computer-aided cryptographic proofs, 2005. At `http://eprint.iacr.org/2005/181/`.
9. J.I. den Hartog and E.P. de Vink. Verifying probabilistic programs using a Hoare like logic. *International Journal of Foundations of Computer Science*, 13(3):315–340, 2002.
10. C.A.R. Hoare. An axiomatic basis for computer programming. *Communications of the ACM*, 12:576–580, 1969.
11. R. Impagliazzo and B. M. Kapron. Logics for reasoning about cryptographic constructions. *Journal of Computer and Systems Sciences*, 72(2), 2006.
12. C. Morgan, A. McIver, and K. Seidel. Probabilistic predicate transformers. *ACM Trans. Program. Lang. Syst.*, 18(3):325–353, 1996.
13. V. Shoup. Sequences of games: a tool for taming complexity in security proofs, May 2005. At `http://www.shoup.net/papers/games.pdf`.