Towards mechanized correctness proofs for cryptographic algorithms
Axiomatization of a probabilistic Hoare style logic.

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

Abstract. In [5] we build a formal verification technique for game based correctness proofs of cryptographic algorithms based on a probabilistic Hoare style logic [10]. An important step towards enabling mechanized verification within this technique is an axiomatization of implication between predicates which is purely semantically defined in [10]. 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 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.


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 style reasoning system deals with reasoning about programs, however, it assumes that a reasoning system for reasoning about the probabilistic predicates is available. 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 axiomatization of the probabilistic predicate logic.

In the remainder of this paper we provide some basic definitions and introduce the probabilistic logic, a probabilistic programming language and the probabilistic Hoare style logic, recalling results from [10]. Next we introduce the axiomatization of implication of predicates in an equational fashion. The role of the axiomatization in proving correctness of cryptographic algorithms using game based proofs is illustrated by using it in the correctness proof for ElGamal algorithm presented in [5]. We conclude with a discussion of the results and of the further steps needed to obtain completely mechanized proofs.

2 The Basics

We shortly recall the logical predicates used by the probabilistic Hoare style logic $pL$ (see [10]). We introduce probabilistic states $\Theta$ and the validity relation $\vdash$ for predicates which gives whether a predicate holds in a given probabilistic state.

Expressions We define integer expressions $e$ and Boolean expressions (or ‘conditions’) $c$ by:

\[
\begin{align*}
e &::= n \mid x \mid e + e \mid e - e \mid e \cdot e \mid e \text{ div } e \mid e \text{ mod } e \mid f(e, \ldots, e) \\
c &::= \text{true} \mid \text{false} \mid b \mid e = e \mid e < e \mid c \land c \mid c \lor c \mid \neg c \mid c \rightarrow c
\end{align*}
\]

where $x$ is a variable of type (or ‘has range’) integer, $b$ is a variable of type Boolean and $n$ a number. 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 (or statements) $s$ are defined by:

$$
s ::= \text{skip} \mid x := e \mid s \mid \text{if } c \text{ then } s \text{ else } s \mid D(e, \ldots, e; x, \ldots, x) \mid s +_p s
$$

where $x$ is a *program* variable, $e$ an expression of the right type, $; \sigma$ denotes sequential composition, $\text{if } c \text{ then } s \text{ else } s$ a conditional choice, $D$ a procedure call and $+_p$ probabilistic choice. The procedure call $D(e, e'; x, y)$ causes the body $B_D$ 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 $s +_p s'$ program $s$ is executed with probability $\rho$ and $s'$ is executed with probability $1 - \rho$.

**States** A deterministic state, $\sigma \in S$, 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 $\sigma_i$ is $\rho_i$, $i \geq 0$. The sum of all $\sigma_i$ is atmost 1. For simplicity and without loss of generality we assume that each state $\sigma$ occurs at most once in $\theta$; multiple occurrences of a single state can be merged into one single occurrence by adding the probabilities, e.g. $1 \cdot \sigma$ rather than $\frac{3}{4} \cdot \sigma + \frac{1}{2} \cdot \sigma$.

**Operations on states** The value $\mathcal{V}(e)(\sigma)$ of an expression $e$ in a state $\sigma$ is defined as usual. A variant $\sigma[e/x]$ of a deterministic state $\sigma$ is a state which only differs from $\sigma$ for the variable $x$ where it returns $\mathcal{V}(e)(\sigma)$ the value of expression $e$ (in state $\sigma$). A variant $\theta[e/x]$ of a probabilistic state $\theta$ is obtained by taking the variant element wise, i.e. if $\theta = \rho_1 \cdot \sigma_1 + \rho_2 \cdot \sigma_2 + \ldots$ then $\theta[e/x] = \rho_1 \cdot \sigma_1[e/x] + \rho_2 \cdot \sigma_2[e/x] + \ldots$.

The operations $+$ (addition), $\cdot$ (scaling) on probabilistic states (which are functions to $[0,1]$) are the addition and scaling of functions. The operation $+_p$ (probabilistic choice) combines these two: $\theta +_p \theta' = \rho \cdot \theta + (1 - \rho) \cdot \theta'$. Finally the operation $c? \theta$ (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 $\sigma_1, \sigma_3$ but not in $\sigma_2$, etc. then $c?\theta = \rho_1 \cdot \sigma_1 + \rho_3 \cdot \sigma_3 + \ldots$.

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

$$
dp ::= \text{true} \mid \text{false} \mid b \mid e \leq e \mid dp \land dp \mid dp \lor dp \mid \neg dp \mid dp \rightarrow dp \mid \exists i \vdash dp \mid \forall i \vdash dp
$$

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

$$
e_r ::= \rho \mid r \mid P(dp) \mid e_r + e_r \mid e_r - e_r \mid e_r \cdot e_r \mid e_r / e_r \mid f(e, \ldots, e)
$$
A probabilistic state provides the distribution of the program variables with which we evaluate \( \mathbb{P}(dp) \). For logical variables such as \( r \) we still have a standard interpretation which simply provides the value.

**Example 1.** The expression \( r + \mathbb{P}(x > 2) \) 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 the \( \mathbb{P}(\cdot) \) construct.

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

\[
p ::= \text{true} \mid \text{false} \mid b \mid e \leq e \mid p \land p \mid p \lor p \mid \neg p \mid p ightarrow p \mid \exists j : p \mid \forall j : p
\]

\[
\mid \rho \cdot p \mid p + p \mid p \oplus_p p \mid c?p
\]

with the restriction that the arithmetical operations \((\rho, +, \oplus_p, c?)\) do not occur negatively, i.e. in \( \neg p \) or \( p \rightarrow q \) the predicate \( p \) may not contain arithmetical operations. 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 \( \mathbb{P}(x = 1) = r \) states that probability of a the deterministic predicate, in this case \( x = 1 \) holding is equal to \( r \).

The predicate \( \forall i, j : (P(x = i \land y = j) = P(x = i) \cdot P(y = j) \) 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:

\[
\theta \models \rho \cdot p \text{ when there exists } \theta' : \theta = \rho \cdot \theta', \theta \models p
\]

\[
\theta \models p + p' \text{ when there exists } \theta_1, \theta_2 : \theta = \theta_1 + \theta_2, \theta_1 \models p \text{ and } \theta_2 \models p'
\]

\[
\theta \models p \oplus_p p' \text{ when there exists } \theta_1, \theta_2 : \theta = \theta_1 \oplus_p \theta_2, \theta_1 \models p \text{ and } \theta_2 \models p'
\]

\[
\theta \models c?p \text{ when there exists } \theta' : \theta = c?\theta', \theta \models p
\]

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 \( (\mathbb{P}(x = 1) = \frac{1}{4}) + (\mathbb{P}(x > 2) = r) \) 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 \( \mathbb{P}(x = 1) = \frac{1}{4} \), and \( \frac{1}{4} \cdot [x = 2] + \frac{1}{4} \cdot [x = 3] + \frac{1}{4} \cdot [x = 4] \) which satisfies \( \mathbb{P}(x > 2) = r \).

Predicate \( (\mathbb{P}(x = 1) = \frac{1}{4}) + (\mathbb{P}(x < 3) = r) \) 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 [10, 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.

\[
\{ p[x/e] \} x := e \{ p \} \quad (\text{Assign})
\]

\[
\begin{align*}
\frac{\{ p \} s \{ p' \} \quad \{ p' \} s' \{ q \}}{\{ p \} s ; s' \{ q \}} & \quad (\text{Seq}) \\
\frac{\{ p \} s \{ q \} \quad \{ p \} s' \{ q' \}}{\{ p \} s \oplus s' \{ q \oplus q' \}} & \quad (\text{Prob}) \\
\frac{\{ p \} \{ q \} \quad \{ q \}}{\{ q \}} & \quad (\text{Cons}) \\
\frac{\{ p \} B_D \{ q \} \quad \{ p \} \{ q \}}{\{ p[\alpha_1,\ldots,\alpha_n] \{ q[\beta_1,\ldots,\beta_m] \}} & \quad (\text{B})
\end{align*}
\]

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 \( p \Rightarrow p' \) if \( \forall \theta, I : (\theta, I) \models p \rightarrow (\theta, I) \models p' \) and \( p \equiv p' \) if \( p \Rightarrow p' \) and \( p' \Rightarrow p \).

We first present the congruence results for \( \Rightarrow \).

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

If \( p \Rightarrow p' \) and \( q \Rightarrow q' \) then \( p \ op \ q \Rightarrow p \ op \ q' \) for \( op \in \{ \wedge, \lor, +, \oplus \rho \} \) and \( p' \rightarrow q \Rightarrow p \rightarrow q' \).

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

For \( \cdot \) we also have the reverse, i.e.

\[
\text{if } \rho \cdot p \Rightarrow \rho \cdot p' \text{ then } p \Rightarrow p'.
\]

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 \( \theta \) the construct \( \mathbb{P}(\cdot) \) is just another function symbol in the probabilistic (i.e. real valued) expressions. When reasoning non-probabilistically we ignore the state \( \theta \). Thus the exact function represented by \( \mathbb{P}(\cdot) \) 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:

\[
\begin{align*}
BB(p) &= b_p() \quad \text{or} \\
BB(p) &= p \quad \text{for } p \ \mathbb{P}-\text{free, or} \\
BB(p) &= BB(e) \leq BB(e') \quad \text{for } p = e \leq e', \ or \\
BB(p) &= BB(q) \ op \ BB(q') \quad \text{for } p = q \ op \ q', \ op \in \{ \land, \lor, \rightarrow \}, \ or \\
BB(p) &= op \ BB(q) \quad \text{for } p = op \ q, \ op \in \{ \forall i, \exists i, \neg \}
\end{align*}
\]

Lemma 7. If \( I \models_d BB(p) \) then for any \( \theta \) we have \( (I, \theta) \models p \).

If \( I \models_d BB(p) \rightarrow BB(q) \) then \( p \Rightarrow q \).

If \( I \models_d BB(p) \leftrightarrow BB(q) \) then \( p \equiv q \).
Towards mechanized correctness proofs for cryptographic algorithms

The intuition behind this result is as follows: The validity of $BB(p)$ is obtained irrespective of the value given by the functions $r_d$ and $b_p$. By fixing $\theta$ we only select one possible function for which the predicate is true.

Example 8. (i) $i = j \Rightarrow P(x = i) = P(x = j)$ can be derived by noting: $BB(P(x = i) = P(x = j)) = (f_{x=\iota}(i) = f_{x=\iota}(j))$ and $\models (i = j) \rightarrow (f_{x=\iota}(i) = f_{x=\iota}(j))$.

(ii) $p + q \land p + q \equiv p + q$ can be derived by noting: $BB(p + q \land p + q) = BB(p + q) \land BB(p + q) = f_{p+q} \land f_{p+q}$ and $\models f_{p+q} \land f_{p+q} \leftrightarrow f_{p+q}$.

(iii) We cannot derive $(p \land p) + q \equiv p + q$ directly using lemma 7 as we cannot ‘look into’ the black box $(p \land p) + q$. However, we can first note that $p \land p \equiv p$ 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

The main remaining question is how to deal with the arithmetical operators. For each operators we provide a list of valid equivalences that can be used as an axiomatic basis of equational reasoning with these predicates. These equivalences consist of a list of basic properties and a list of distributivity laws for each of the operators. We do not treat the operator $\oplus_{p}$ as $p \oplus_{p} p' \equiv \rho \cdot p + (1 - \rho) \cdot p'$.

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, while probabilistic predicates may not. 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 $\Box$-free predicate which holds exactly when $dp$ is fulfilled by every deterministic state $\sigma$, i.e. $(\theta, I) \models \Box(dp)$ iff $\forall \sigma : (\sigma, I) \models dp$.

Below we present the axioms and explain the main characterization and some of the difficulties with distributivity.

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

$$P(\text{false}) = 0 \equiv \text{true} \quad (A1)$$

$$P(\text{true}) \leq 1 \equiv \text{true} \quad (A2)$$

$$P(dp \lor dp') = P(dp) + P(dp') - P(dp \land dp') \equiv \text{true} \quad (A3)$$

$$\Box(dp \rightarrow dp') \Rightarrow P(dp) \leq P(dp') \quad (A4)$$
The first three axioms state that the given equations on chances are tautologies. These properties are directly derived from properties of (sub-)probability measures. Recall that incomplete states (e.g., caused by non-termination) may satisfy \( P(\text{true}) < 1 \). The last axiom lifts reasoning on deterministic predicates to probabilistic predicates: If \( dp \rightarrow dp' \) must hold then the probability \( P(dp) \) of \( dp \) cannot be more than the probability \( P(dp') \) of \( dp' \). (Note that any axiom of the form \( p \Rightarrow q \) could be equivalently written as \( p \equiv p \land q \) thus still fits within the equational system.)

**Example 10.** (i) We obtain
\[
\Box(dp \leftrightarrow dp') \Rightarrow P(dp) = P(dp')
\]
directly from A4.

(ii) We have
\[
P(dp) = P(\text{true}) - P(\neg dp) \equiv \text{true}
\]
In the derivation we streamline the notation slightly, writing \( e \equiv (e_1 = e_2 = \ldots = e_n) \) rather than \( \text{true} \equiv (e_1 = e_2) \equiv \ldots \equiv (e_1 = e_2) \land \ldots \land (e_n-1 = e_n) \equiv (e_1 = e_r_n) \). Using this notation gives
\[
\begin{align*}
P(dp') \ [\text{A3}] &= P(dp \land \neg dp') + P(dp \lor \neg dp) - P(\neg dp) \\
[\text{ex. 10(i)}] &= P(\text{false}) + P(\text{true}) - P(\neg dp) \\
[\text{A1}] &= 0 + P(\text{true}) - P(\neg dp) \\
&= P(\text{true}) - P(\neg dp)
\end{align*}
\]
We will see the use of the expression ‘\( P(\text{true}) \)’ return in rules below. This expression captures the total probability of the probabilistic state. As noted above this total probability may be less than 1.

**Axioms for \( \cdot \).** The following two sets of axioms capture the behaviour of the \( \cdot \) operator. We first present the basic axioms for the \( \cdot \) operator after which we provide distributivity laws.

\[
\begin{align*}
\rho \cdot (P(dp) = r) &\equiv P(dp) = \rho \cdot r \land P(\text{true}) \leq \rho \quad (\text{A5}) \\
\rho \cdot (\rho' \cdot p) &\equiv (\rho \cdot \rho') \cdot p \quad (\text{A6}) \\
\rho \cdot p &\equiv p \land P(\text{true}) \leq \rho \quad \text{if } p \text{ is } P\text{-free} \quad (\text{A7})
\end{align*}
\]

The axiom A5 characterizes the \( \cdot \) operator: The probability of all events is scaled. Clearly the probability of the event \( dp \) becomes \( \rho \cdot r \) but also the probability of other events is scaled, which is expressed by \( P(\text{true}) \leq \rho \); no event can have a probability greater than \( \rho \cdot 1 \) after scaling.

The second axiom states that first scaling with \( \rho' \) and then with \( \rho \) is the same as scaling directly with \( \rho \cdot \rho' \).

As a \( P \)-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 (\( P(\text{true}) \)) that the state can have; after scaling it can be at most \( \rho \).
Example 11. (i) We have commutativity for $\cdot$:

\[
\rho \cdot (\rho' \cdot p) \equiv (\rho \cdot \rho') \cdot p \\
\equiv (\rho' \cdot \rho) \cdot p \\
[A6] \equiv \rho' \cdot (\rho \cdot p)
\]

(ii) The operation $\cdot$ influences the state but not the logical variables:

\[
\frac{1}{2} \cdot (P(x = 1) = \frac{1}{2}) \Rightarrow P(x = 1) = \frac{1}{4} \\
\frac{1}{2} \cdot (r = \frac{1}{2}) \Rightarrow r = \frac{1}{2} \\
\frac{1}{2} \cdot (P(x = 1) = r \land r = \frac{1}{2}) \Rightarrow P(x = 1) = \frac{1}{4} \land r = \frac{1}{2} \\
\Rightarrow P(x = 1) = \frac{1}{4}
\]

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

\[
\rho \cdot (p \circledast p') \equiv (\rho \cdot p) \circledast (\rho \cdot p') \\
\rho \cdot (op \cdot p) \equiv op \cdot (\rho \cdot p') \\
\rho \cdot (p + q) \equiv \exists r : \rho \cdot (p \land P(true) \leq r) + \rho \cdot (q \land P(true) \leq 1 - r) \quad (A10)
\]

with $op \in \{\land, \lor, \oplus\}$, $op' \in \{\exists i ; : \rho', c?\}$ and $r$ a fresh variable not occurring in $p$ or $q$. (Note that the case $op' = \rho'$ expresses commutativity of $\cdot$ which has already been derived in example 11(i).)

The scaling operator $\cdot$ distributes straightforwardly over all other operators except $+$ for which there is a complication: If $\theta$ satisfies $p$ and $\theta'$ satisfies $p'$ and $\theta + \theta'$ exists (is a probabilistic state) then this state satisfies $p + p'$. However, the combination $\theta + \theta'$ may not exist because it would have a total probability 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 a 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. Note that as a direct consequence of this rule we have $\rho \cdot (p + q) \Rightarrow \rho \cdot p + \rho \cdot q$.

**Axioms for $\propto$** In giving the characterization of the $\propto$ 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.

\[
c? (P(dp) = r) \equiv P(\neg c) = 0 \land \exists \delta : P(dp) = r - r_\delta \land 0 \leq r_\delta \leq 1 - P(true) \land \square (dp \rightarrow c) \rightarrow r_\delta = 0 \quad (A11)
\]

\[
\begin{align*}
&c? (c? \cdot p) \equiv (c \land c')?p \\
&c? p \equiv p \land P(\neg c) = 0 \quad &\text{if } p \text{ is } P\text{-free} \\
&true? p \equiv p \quad &\text{(A14)} \\
&\square (c \leftrightarrow c') \land c? p \Rightarrow c'? p
\end{align*}
\]
After removing the part of the state where \( c \) does not hold, the chance of \( \neg c \) must be 0. By removing this part of the state the probability that \( dp \) holds is also decreased by some amount, say \( r_\delta \). Clearly \( r_\delta \) is at least 0 and at most \( 1 - P(\text{true}) \) which is the total amount of probability that is missing from the state. In the special case that \( dp \) logically implies \( c \) the probability of \( dp \) cannot be decreased by removing states not satisfying \( c \) so \( r_\delta \) 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 \( P(\neg c) = 0 \).)

The last two axioms lift reasoning on conditions to reasoning on probabilistic predicates similar to what axiom A4 did for deterministic predicates.

**Example 12.** (i) We have commutativity of ?:

\[
c? (c' ? p) \equiv (c \land c') ? p \equiv (c' \land c) ? p \equiv c'? (c ? p)
\]

(ii) The total probability of \( c?p \) is the probability of \( c \) in \( p \).

\[
c?(P(c) = r) \Rightarrow [A11] \ P(\neg c) = 0 \land \exists r_\delta : P(c) = r - r_\delta \land r_\delta = 0
\]

\[
\Rightarrow P(\neg c) = 0 \land P(c) = r
\]

\[
\Rightarrow [A3] \ P(c \lor \neg c) = 0 + r + 0
\]

\[
\Rightarrow P(\text{true}) = r
\]

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

\[
\square(dp \rightarrow c) \land c?(P(dp) = r) \Rightarrow [A11] \ \exists r_\delta : P(dp) = r - r_\delta \land r_\delta = 0
\]

\[
\Rightarrow P(dp) = r
\]

The next set of axioms provide distributivity laws for ?.

\[
c?(p \lor q) \equiv (c?p) \lor (c?q) \tag{A16}
\]

\[
c? (\exists i : p) \equiv \exists i : (c?p) \tag{A17}
\]

\[
c?(p + q) \equiv \exists r : c?(p \land P(\text{true}) \leq r) + c?(q \land P(\text{true}) \leq 1 - r) \tag{A18}
\]

with \( i \) not free in \( c \) and \( r \) a fresh variable not occurring in \( p \) or \( q \). With distributivity over + we have the same complication as for \( \cdot \); the axiom above capture that \( c?(p + q) \) is the same as \( c?p + c?q \) as long as the total probability of 1 is not exceeded. Similar to the case with \( \cdot \) we have that \( c?(p + q) \Rightarrow c?p + c?q \) follows directly.

The operator \( ? \) does not distribute over \( \land \) (nor over \( \forall i : \)), \( c?(p \land q) \neq c?p \land c?q \), 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:

\[
c?(\forall i : p) \Rightarrow \forall i : c?p \tag{A19}
\]

\[
c?(p \land q) \Rightarrow c?p \land c?q \tag{A20}
\]

\[
c?(p \land q) \equiv c?p \land c?q \quad \text{if } p \text{ is } P\text{-free} \tag{A21}
\]
Towards mechanized correctness proofs for cryptographic algorithms

For a $\mathbb{P}$-free predicate the equivalence does hold as a $\mathbb{P}$-free predicate is not influenced by the $?$ operator. Note that by using axiom (A13) we already have:

$$c \? \left( \forall i : p \right) \equiv \left( \forall i : p \right) \mathbb{P} \left( \neg c \right) = 0 \equiv \forall i : (p \wedge \mathbb{P}(\neg c) = 0) \equiv \forall i : (c?p)$$

thus a similar ‘$\mathbb{P}$-free-axiom’ for $\forall$ is not needed.

### Axioms for +

The following two sets of axioms capture the behaviour of the $+$ operator. We first present the basic axioms for the $+$ operator after which we provide distributivity laws.

$$\mathbb{P}(dp) = r + \mathbb{P}(dp') = r' \equiv r \leq \mathbb{P}(dp) \wedge r' \leq \mathbb{P}(dp') \wedge$$

$$\mathbb{P}(dp \wedge dp') \leq r + r' \leq \mathbb{P}(dp \vee dp') \quad (A22)$$

$$p + q \equiv q + p \quad (A23)$$

$$(p_1 + p_2) + p_3 \equiv p_1 + (p_2 + p_3) \quad (A24)$$

$$p + p' \equiv p \wedge (\text{true} + p') \quad \text{if } p \text{ is } \mathbb{P}\text{-free} \quad (A25)$$

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 $dp$ may be higher than $r$ if it also has a probability of occurring in the right hand side. 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 $r$ in the left hand part and at least $r'$ in the right hand part. The second rule (commutativity) and third rule (associativity) are straightforward while the last rule allows moving ‘non-probabilistic’ properties to outside the $+$. 

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

$$\mathbb{P}(dp) = r + \mathbb{P}(dp) = r'$$

$$[A22] \equiv r \leq \mathbb{P}(dp) \wedge r' \leq \mathbb{P}(dp) \wedge \mathbb{P}(dp \wedge dp) \leq r + r' \leq \mathbb{P}(dp \vee dp)$$

$$[A4] \equiv r \leq \mathbb{P}(dp) \wedge r' \leq \mathbb{P}(dp) \wedge \mathbb{P}(dp) \leq r + r' \leq \mathbb{P}(dp)$$

$$[r, r' \geq 0] \equiv \mathbb{P}(dp) = r + r'$$

Two parts are added so the probability of a given event is the sum of the probabilities in both parts. (Note that the remark $[r, r' \geq 0]$ in the last equivalence is needed only for the reverse implication.)

Below we provide a set of distributivity laws for $+$. Recall that we already have given rules for distributivity of $\cdot$ (axiom A10) and $?$ (axiom A18) over $\vee$.

$$\left( p \vee p' \right) + q \equiv \left( p + q \right) \vee \left( p' + q \right) \quad (A26)$$

$$\left( \exists i : p \right) + q \equiv \exists i : (p + q) \quad \text{i not free in } q \quad (A27)$$

The operator $+$ does not distribute over $\wedge$ and $\forall$. Similar to the case for $?$ we have that $\left( p \wedge p' \right) + q$ is stronger than $\left( p + q \right) \wedge \left( p' + q \right)$. In $\left( p + q \right) \wedge \left( p' + q \right)$ the
state can be split in different ways to satisfy $p + q$ and $p' + q$ thus there is no guarantee that there exists a part of the state which satisfies both $p$ and $p'$. We have to suffice with implication and a special case.

\[
(\forall i : p) + q \Rightarrow \forall i : (p + q) \quad \text{if } i \text{ not free in } q \quad (A28)
\]

\[
(p \land p') + q \Rightarrow (p + q) \land (p' + q) \quad (A29)
\]

\[
(p \land p') + q \equiv (p + q) \land (p' + q) \quad \text{if } p \text{ is } \mathbb{P}-\text{free} \quad (A30)
\]

Again a similar ‘$\mathbb{P}$-free-axiom’ for $\forall$ is not needed as, using axiom A25 we already get, for $i$ not free in $q$: $(\forall i : p) + q \equiv (\forall i : p) \land (\text{true} + q) \equiv \forall i : (p \land (\text{true} + q)) \equiv \forall i : (p + q)$

This completes the axiom system. In the next section we illustrate the calculus by axiomatizing a 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 reasoning rules for the logic can be applied in the setting of a Hoare-style logic by treating an example presented in [5]. The proof outline in Table 1 represents the main derivation using the rules of the probabilistic Hoare style logic. The complete proof of El-Gamal security uses several transformations to reach the program (game) below. We refer to [5] for details.

We first recall some short-hand notation and results from [5].

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

\[
I(e, e') := \forall i, j : \mathbb{P}(e = i \land e' = j) = \mathbb{P}(e = i) \cdot \mathbb{P}(e' = j)
\]

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

\[
R_{S,S'}(e, e') := \forall i, j : \mathbb{P}(e = i \land e' = j) = 1/|S| \cdot 1/|S'|
\]

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.**

\[
R_{S,S'}(e, e') \equiv R_S(e) \land R_{S'}(e') \land I(e, e')
\]

\[
I(e, e') \Rightarrow I(e, f(e'))
\]
Below we will illustrate the calculus using a main derivation in \[5\] which shows that, after a number of transformations, the security game is similar to a coin toss. The transformed security game is the program \(s\) given by the numbered lines in Table 1. For this procedure we derive

\[
\{ R_{Z^3, RNDBool}(v1, v2, v3, v4, v5) \} \mathbf{s} \{ P(x1) = 1/2 \}
\]

In other words, 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.

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 second result of example 12.

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

\[
P(v5 = b) = P((v5 \land b) \lor (\neg v5 \land \neg b)) = P(v5 \land b) + P(\neg v5 \land \neg b) - 0
\]

\[
[I(v5, b)] = P(v5) \cdot P(b) + P(\neg v5) \cdot P(\neg b)
\]

\[
[R(v5)] = \frac{1}{2} \cdot P(b) + \frac{1}{2} \cdot P(\neg b) = \frac{1}{2} \cdot (P(b) + P(\neg b)) = \frac{1}{2} \cdot P(b \lor \neg b)
\]

\[
= \frac{1}{2} \cdot P(\text{true}) = \frac{1}{2} \cdot 1 = \frac{1}{2}
\]

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

\[
(\neg v5)?R_{Z^3, RNDBool}(v1, v2, tmp, v4, v5)
\]

\[
[A11] \Rightarrow P(v5 = \text{true}) = 0
\]

\[
[A44] \Rightarrow P(v1 = i_1, v2 = i_2, tmp = i_3, v4 = i_4, v5 = \text{true}) = 0
\]

\[
(v5)?R_{Z^3, RNDBool}(v1, v2, tmp, v4, v5)
\]

\[
[\text{Def. } R] \Rightarrow (v5)?(P(v1 = i_1, v2 = i_2, tmp = i_3, v4 = i_4, v5 = \text{true}) = \rho)
\]

\[
[\text{ex. } 12(iii)] \Rightarrow P(v1 = i_1, v2 = i_2, tmp = i_3, v4 = i_4, v5 = \text{true}) = \rho
\]

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

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

Symmetrically we also get

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

Thus

\[
p_1 + p_2 \Rightarrow \forall i_5 : P(v1 = i_1, v2 = i_2, tmp = i_3, v4 = i_4, v5 = i_5) = \rho
\]
\[
\{R_Z^2, RND, Bool(v_1, v_2, v_3, v_4, v_5)\} \quad (11)
\]
\[
\{R_Z^2, RND, Bool(v_1, v_2, v_3 \cdot A_0(v_1, v_4), v_4, v_5) \}
\land R_Z^2, RND, Bool(v_1, v_2, v_3 \cdot A_1(v_1, v_4), v_4, v_5) \}
\]
\[
m_0 := A_0(v_1, v_4);
\]
\[
m_1 := A_1(v_1, v_4);
\]
\[
\begin{align*}
p_0 \triangleq & \{R_Z^2, RND, Bool(v_1, v_2, v_3 \cdot m_0, v_4, v_5) \\
& \land R_Z^2, RND, Bool(v_1, v_2, v_3 \cdot m_1, v_4, v_5) \} \\
\{R_Z^2, RND, Bool(v_1, v_2, v_3, v_4, v_5) \}
\end{align*}
\]
if \(v_5 = \text{false}\) then
\[
\begin{align*}
\{(-v_5)?p_0\} & \quad (12) \quad \{(v5)?R_Z^2, RND, Bool(v_1, v_2, v_3 \cdot m_0, v_4, v_5) \}
\end{align*}
\]
\[
\text{tmp} := v_3 \cdot m_0
\]
\[
p_1 \triangleq \{(v5)?R_Z^2, RND, Bool(v_1, v_2, \text{tmp}, v_4, v_5) \}
\]
else
\[
\begin{align*}
\{(v5)?p_0\} & \quad (13) \quad \{(v5)?R_Z^2, RND, Bool(v_1, v_2, v_3 \cdot m_1, v_4, v_5) \}
\end{align*}
\]
\[
\text{tmp} := v_3 \cdot m_1
\]
\[
p_2 \triangleq \{(v5)?R_Z^2, RND, Bool(v_1, v_2, \text{tmp}, v_4, v_5) \}
\]
fi
\[
\{p_1 + p_2\} \quad (14) \quad \{R_Z^2, RND, Bool(v_1, v_2, \text{tmp}, v_4, v_5) \} \quad (15)
\]
\[
\{R_{\text{Bool}}(v_5) \land I(v_5, A_2(v_1, v_2, \text{tmp}, v_4)) \}
\]
\[
b := A_2(v_1, v_2, \text{tmp}, v_4)
\]
\[
\{R_{\text{Bool}}(v_5) \land I(v_5, b)\} \quad (16) \quad \{P(v_5 = b) = 1/2\}
\]
if \(v_5 = b\) then
\[
\begin{align*}
\{(v_5 = b)?(P(v_5 = b) = 1/2)\} & \quad (17) \quad \{P(\text{true}) = 1/2\}
\end{align*}
\]
\[
x_1 := \text{true}
\]
\[
\{P(x_1) = 1/2\}
\]
else
\[
\begin{align*}
\{(v_5 \neq b)?(P(v_5 = b) = 1/2)\} & \quad (18) \quad \{P(\text{false}) = 0\}
\end{align*}
\]
\[
x_1 := \text{false}
\]
\[
\{P(x_1) = 0\}
\]
fi
\[
\{(P(x_1) = 1/2) + (P(x_1) = 0)\} \quad (19)
\]
\[
\{P(x_1) = 1/2\}
\]

**Table 1.** Derivation of \(\{R_Z^2, RND, Bool(v_1, v_2, v_3, v_4, v_5)\} \Rightarrow \{P(x_1) = 1/2\}\)
Towards mechanized correctness proofs for cryptographic algorithms

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:

$$\forall k, l : \exists m : km = l$$  \hspace{1cm} (Mul I)

$$\forall k, l, m : (km = lm) \rightarrow k = l$$  \hspace{1cm} (Mul II)

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

$$R_{Z^3,RND,\text{Bool}}(v_1, v_2, v_3, v_4, v_5)$$

$$\Rightarrow \forall j : P(v_1 = i_1, v_2 = i_2, v_3 = j, v_4 = i_4, v_5 = i_5) = \rho$$

[Mul I] $$\Rightarrow \forall i_3 : \exists j : j \cdot f(i_1, i_4) = i_3 \land$$

$$P(v_1 = i_1, v_2 = i_2, v_3 = j, v_4 = i_4, v_5 = i_5) = \rho$$

[Mul II] $$\Rightarrow \forall i_3 : \exists j : \neg (v_3 = j) \land v_3 \cdot f(i_1, i_4) = i_3 \land$$

$$P(v_1 = i_1, v_2 = i_2, v_3 = j, v_4 = i_4, v_5 = i_5) = \rho$$

[ex.10(i)] $$\Rightarrow \forall i_3 : \exists j : P(v_1 = i_1, v_2 = i_2, v_3 = j, v_4 = i_4, v_5 = i_5) = \rho$$

$$\Rightarrow \forall i_3 : P(v_1 = i_1, v_2 = i_2, v_3 \cdot f(v_1, v_4) = i_3, v_4 = i_4, v_5 = i_5) = \rho$$

$$\Rightarrow R_{Z^3,RND,\text{Bool}}(v_1, v_2, v_3 \cdot f(v_1, v_4), v_4, v_5)$$

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 main ingredients of the calculus are the preservation of classical logical reasoning, a congruence result and a list of axioms capturing the basic behaviour and distributivity properties of the arithmetical operators which are specific to the probabilistic Hoare logic.

The next step in the mechanization would be the implementation of probabilistic predicates in a theorem proving system, such as PVS, HOL, etc. We envision a three possible levels of abstraction: The first, most abstract, level is an implementation of the calculus; There is no notion of probabilistic state and the reasoning consist purely of syntactical manipulations, application of the calculus rules and use of the proofcheckers built in mechanisms for reasoning about deterministic predicates. The second level introduces probabilistic states in terms of abstract functions and the probabilistic properties of these functions as axioms. This allows modeling the semantics of predicates and deriving results directly from the probabilistic properties of the states. At this level of abstraction one can
also reason about and derive 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 theoretical aspects such as completeness or 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. Note that with reasonable restrictions it is also possible to define weakest preconditions and obtain a complete reasoning system in the probabilistic Hoare logic setting [4] though the size of these predicates may quickly become unmanageable for our purpose.

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

A Soundness of the Axioms

In this section we illustrate how the soundness of the axioms can be derived by showing the proofs for some of the key axioms. We treat a basic probabilistic axioms and the characterizations of the different operators. First we fix some notation:

**Definition 16.** We use $f[S]$ to denote the sum of $f$ over all elements specified by $S$:

- For $f$ a function to the reals and $S$ a $f$-countable\(^1\) subset of its domain:
  $$f[S] := \sum_{s \in S} f(s)$$
- For a probabilistic state $\theta$ and a deterministic predicate $dp$ we define
  $$\theta[dp] := \theta[\{ \sigma \mid \sigma \models dp \}]$$
- For the special case $dp = \text{true}$ we write $|\theta|$, called the probability mass of $\theta$:
  $$|\theta| := \theta[\text{true}]$$

To show correctness of axiom A3 we first note that $f[S \cup S'] = f[S] + f[S'] - f[S \cap S']$. Using this fact we get:

\[(A3) \quad \mathbb{P}(dp \lor dp') = \mathbb{P}(dp) + \mathbb{P}(dp') - \mathbb{P}(dp \land dp') \equiv \text{true} \]

**Proof.** $\square$ Clear.

- If $\theta \models dp \lor dp'$ then
  $$\theta[dp \lor dp'] = \theta[\{ \sigma \mid \sigma \models dp \} \cup \{ \sigma \mid \sigma \models dp' \}] = \theta[\{ \sigma \mid \sigma \models dp \}] + \theta[\{ \sigma \mid \sigma \models dp' \}] - \theta[\{ \sigma \mid \sigma \models dp \} \cap \{ \sigma \mid \sigma \models dp' \}] = \theta[dp] + \theta[dp'] - \theta[dp \land dp']$$
  thus $\theta \models \mathbb{P}(dp \lor dp') = \mathbb{P}(dp) + \mathbb{P}(dp') - \mathbb{P}(dp \land dp')$.

\(^1\) i.e. $f$ is non-zero on at most countably many elements of $S$. 

---

To show correctness of axiom A5 we first note that \( \rho \cdot (\theta[S]) = (\rho \cdot \theta)[S] \) and \( \rho \cdot \theta \) is a state if \( \rho \cdot [\theta] \leq 1 \). Using these facts we get:

\[
(A5) \quad \rho \cdot (\mathbb{P}(dp) = r) = \mathbb{P}(dp) = \rho \cdot r \land \mathbb{P}(\text{true}) \leq \rho
\]

**Proof.** If \( \theta \models \rho \cdot (\mathbb{P}(dp) = r) \) then \( \theta \models \rho \cdot \theta', \theta' \models \mathbb{P}(dp) = r \) for some \( \theta' \in \Theta \).
Thus \( \theta[dp] = \rho \cdot \theta'[dp] = \rho \cdot I(r) \) and \( \theta[\text{true}] = \theta[S] = \rho \cdot \theta'[S] \leq \rho \cdot 1 = \rho \).
Thus \( \theta \models \mathbb{P}(dp) = \rho \cdot r \) and \( \theta \models \mathbb{P}(\text{true}) \leq \rho \).

If \( \theta \models \mathbb{P}(dp) = \rho \cdot r \land \mathbb{P}(\text{true}) \leq \rho \) then \( |\theta| \leq \rho \) thus \( \theta' \models \frac{1}{\rho} \cdot \theta \) is a state in \( \Theta \).
We have \( \theta = \rho \cdot \theta' \) and \( \theta' \models \mathbb{P}(dp) = r \).
Thus \( \theta \models \rho \cdot (\mathbb{P}(dp) = r) \).

To show correctness of axiom A11 we first note that \( \theta = c?\theta + \neg c?\theta \).
Using this fact we get:

\[
(A11) \quad c?(\mathbb{P}(dp) = r) = \mathbb{P}(\neg c) = 0 \land \exists x : \mathbb{P}(dp) = r - x_s \land 0 \leq x_s \leq 1 - \mathbb{P}(\text{true}) \land \Box(dp \rightarrow c) \rightarrow r_s = 0
\]

**Proof.** If \( \theta \models c?(\mathbb{P}(dp) = r) \) then \( \theta \models c?\theta', \theta' \models \mathbb{P}(dp) = r \) for some \( \theta' \in \Theta \).
Thus \( \theta[\neg c] = c?\theta' [\neg c] = 0 \) as \( c?\theta'(\sigma) = 0 \) when \( \sigma \neq c \).
Put \( \theta'' := c?\theta' \) then \( \theta'' = \theta + \theta'' \), i.e. \( \theta = \theta' - \theta'' \).
Also, put \( I(x_s) := \theta''[dp] \) then \( \theta'[dp] = \theta'[dp] - \theta''[dp] = I(r) - I(x_s) \) and \( I(x_s) \geq 0 \) and \( I(x_s) = \theta''[dp] \leq \theta''[S] = \theta'[S] - \theta[S] \leq 1 - \theta[S] \).
Finally, if \( \Box(dp \rightarrow c) \) then \( I(x_s) = \theta''[dp] \leq \theta''[c] = 0 \).

If \( \theta \models \mathbb{P}(\neg c) = 0 \land \exists x : \mathbb{P}(dp) = r - x_s \land 0 \leq x_s \leq 1 - \mathbb{P}(\text{true}) \land \Box(dp \rightarrow c) \rightarrow r_s = 0 \) then \( \theta = c?\theta \) as \( \theta[\neg c] = 0 \).
If \( \Box(dp \rightarrow c) \) then put \( \theta' := \theta \).
We have \( \theta = c?\theta' \) and \( \theta'[dp] = I(r) - I(x_s) = I(r) - 0 = I(r) \).
Thus \( \theta = c?\theta', \theta' \models \mathbb{P}(dp) = r \).
Otherwise there exists \( \sigma : \sigma \models dp, \sigma \neq c \).
Put \( \theta' := \theta + I(x_s) \cdot \sigma \) then \( c?\theta' = \theta \\ and \( \theta'[dp] = \theta[dp] + I(x_s) = (I(r) - I(x_s)) + I(x_s) = I(r) \).
Thus again \( \theta = c?\theta', \theta' \models \mathbb{P}(dp) = r \).

Finally, to show correctness of axiom A22 we let the operation ? work on deterministic predicates and states rather than only on conditions and states: We have \( dp?\theta(\sigma) \) equals \( \theta(\sigma) \) if \( \sigma \models dp \) and 0 otherwise. Using this notation we get:

\[
(A22) \quad \mathbb{P}(dp) = r + \mathbb{P}(dp') = r' \Rightarrow r \leq \mathbb{P}(dp) \land r' \leq \mathbb{P}(dp') \land \mathbb{P}(dp \land dp') \leq r + r' \leq \mathbb{P}(dp \lor dp')
\]

**Proof.** If \( \theta \models \mathbb{P}(dp) = r + \mathbb{P}(dp') = r' \) then \( \theta = \theta_1 + \theta_2, \theta_1 \models \mathbb{P}(dp) = r, \theta_2 \models \mathbb{P}(dp) = r' \) for some \( \theta_1, \theta_2 \in \Theta \).
Thus \( \theta[dp] \geq \theta_1[dp] = I(r) \), i.e. \( \theta \models r \leq \mathbb{P}(dp) \), also \( \theta[dp'] \geq \theta_2[dp'] = I(r') \), i.e. \( \theta \models r' \leq \mathbb{P}(dp') \).
Finally,

\[
\theta[dp \land dp'] = \theta_1[dp \land dp'] + \theta_2[dp \land dp'] \leq \theta_1[dp] + \theta_2[dp']
\]

\[
\theta_1[dp] + \theta_2[dp'] \leq \theta_1[dp \lor dp'] + \theta_2[dp \lor dp'] = \theta[dp \lor dp']
\]

thus \( \theta \models \mathbb{P}(dp \land dp') \leq r + r' \leq \mathbb{P}(dp \lor dp') \).
If \( \theta = r \leq P(dp) \land r' \leq P(dp') \land P(dp \land dp') \leq r + r' \leq P(dp \lor dp') \) then we split \( \theta \) into \( \theta_1 + \theta_2 \) such that \( \theta_1 \models P(dp) = r \) and \( \theta_2 \models P(dp') = r' \). To be able to do this we divide \( \theta \) into four parts depending on whether \( dp \) and/or \( dp' \) are satisfied:

\[
\begin{align*}
\theta_{nn} &:= (-dp \land -dp')?\theta \\
\theta_{ny} &:= (-dp \land -dp')?\theta \\
\theta_{yn} &:= (dp \land -dp')?\theta \\
\theta_{yy} &:= (dp \land dp')?\theta
\end{align*}
\]

Note that the value of \( P(dp) \) in \( \theta \) equals \( |\theta_{yn}| + |\theta_{yy}| \). For each of these parts we determine how much they should contribute to \( \theta_1 \) and \( \theta_2 \). As \( \theta_{nn} \) does not effect the probability of \( dp \) nor that of \( dp' \) it can go anywhere. Below we will put it in \( \theta_1 \). We call the contribution of the other three sections to \( \theta_1 \alpha, \beta \) and \( \gamma \) respectively and the contributions to \( \theta_2 \bar{\alpha}, \bar{\beta} \) and \( \bar{\gamma} \) respectively.

\[
\begin{align*}
\gamma &:= \min\{|\theta_{yy}|, I(x), |\theta_{yn}| - (I(x') - |\theta_{ny}|)\} \\
\beta &:= I(x) - \gamma \\
\alpha &:= |\theta_{ny}| - \bar{\gamma} \\
\theta_1 &:= \theta_{nn} + (\alpha/|\theta_{ny}|) \cdot \theta_{ny} \\
\theta_2 &:= (\bar{\alpha}/|\theta_{ny}|) \cdot \theta_{ny} + (\bar{\beta}/|\theta_{ny}|) \cdot \theta_{yy} \\

\end{align*}
\]

Note that the coefficient of \( \theta_{ny} \) has been chosen to get the contribution to the mass of \( \theta_1 \) equal to \( \alpha = |(\alpha/|\theta_{ny}|) \cdot \theta_{ny}| \) (and similarly for the other coefficients).

As \( \theta_1 + \theta_2 \) must equal \( \theta \) we have that \( \alpha + \bar{\alpha} = |\theta_{ny}| \), and similarly for \( \beta \) and \( \gamma \). Also, because the probability of \( dp \) should be \( I(r) \) in \( \theta_1 \) we have that \( \beta + \gamma = I(r) \) (as \( \theta_{nn} \) and \( \theta_{ny} \) do not contribute to the probability of \( dp \) and similarly, because \( \theta_2 \) should give probability \( I(r') \) to \( dp' \) we have \( \bar{\alpha} + \bar{\gamma} = I(r') \).

Looking at these restrictions we see that fixing \( \gamma \) fixes all parameters. Above we have chosen to make \( \gamma \) as large as possible. There are three factors restricting the choice of \( \gamma \): (1) We cannot put more than there is \( (\gamma \leq |\theta_{yy}|) \). (2) The value of \( P(dp) \) must not exceed \( I(r) \), so certainly \( \gamma \leq I(r) \). (3) The value of \( P(dp') \) must be \( I(r') \) in \( \theta_2 \). As this probability can only come from \( \theta_{yn} \) and \( \theta_{yy} \) we need that \( \bar{\gamma} \) is at least \( I(r') - |\theta_{yn}| \), i.e. \( \gamma \leq |\theta_{yy}| - (I(r') - |\theta_{ny}|) \).

To show that \( \theta_1 \) and \( \theta_2 \) are well defined states it is sufficient to show that \( \gamma, \bar{\gamma}, \beta, \bar{\beta}, \alpha \) and \( \alpha \) are non-negative.

- \( (\gamma \geq 0) \) Clearly \( |\theta_{yy}| \geq 0 \) and \( I(r) \geq 0 \). Also, as \( \theta \models P(dp') \geq r' \), \( I(r') \leq \theta(dp') = \theta_{ny} + \theta_{yy} \), thus \( |\theta_{yy}| - (I(r') - |\theta_{ny}|) \geq 0 \).

- \( (\bar{\gamma} \geq 0) \) As \( \gamma \leq |\theta_{yy}| \) we have \( \bar{\gamma} \geq 0 \).

- \( (\beta \geq 0) \) As \( \gamma \leq |\theta_{yy}| \) we have \( I(r) - \gamma \geq 0 \).

- \( (\bar{\beta} \geq 0) \) We have \( \beta = |\theta_{yn}| - I(r) + \gamma \). If \( \gamma = |\theta_{yy}| \) then \( \beta = |\theta_{yn}| + |\theta_{yy}| - I(r) \geq 0 \) as \( P(dp) \geq r \). If \( \gamma = I(r) \), then \( \beta = |\theta_{yn}| \geq 0 \). If \( \gamma = |\theta_{yy}| - (I(r') - |\theta_{ny}|) \) then \( \beta = |\theta_{yn}| + |\theta_{yy}| + (I(r) + I(r')) \geq 0 \) as \( P(dp \lor dp') \geq r + r' \).

- \( (\alpha \geq 0) \) We have \( \alpha = |\theta_{ny}| - (I(r') - (|\theta_{yy}| - \gamma)) \geq |\theta_{ny}| - (I(r') - (|\theta_{yy}| - (I(r) - \gamma))) = 0 \).

- \( (\bar{\alpha} \geq 0) \) We have \( \bar{\alpha} = I(r') - |\theta_{yy}| + \gamma \). If \( \gamma = |\theta_{yy}| \) then \( \bar{\alpha} = I(r') \geq 0 \). If \( \gamma = I(r) \) then \( \bar{\alpha} = I(r') + I(r) - |\theta_{yy}| > 0 \) as \( r + r' \geq P(dp \land dp') \). If \( \gamma = |\theta_{yy}| - (I(r') - |\theta_{ny}|) \) then \( \bar{\alpha} = |\theta_{ny}| \geq 0 \).
By construction $\theta_1 + \theta_2 = \theta$ and $\theta_1 \models P(dp) = r$ and $\theta_2 \models P(dp') = r'$. Thus $\theta \models P(dp) = r + P(dp') = r'$.

This completes the most involved proofs for the axioms. Correctness of the other axioms can be shown in a similar or simpler fashion.