Proposal for a Master’s thesis

Randomised programs with intermediate output in Isabelle/HOL

Supervisors:  Dr. Andreas Lochbihler
Professor:  Prof. David Basin

Issue date:

Prerequisites

- Knowledge in probability theory
- Familiarity with Isabelle/HOL or willingness to learn Isabelle before the start

Introduction

Randomised computations return a result only with a certain probability. To reason about them, one therefore has to consider the probability distribution over the results. These probability distributions are discrete (i.e., they are completely determined by the probability of each possible result), as an algorithm can return only countably many different results. For example, the Java snippet in Figure 1b returns 0 with probability $\frac{1}{2}$, 1 with probability $\frac{1}{4}$, 2 with probability $\frac{1}{8}$, etc.

Formally, a discrete probability distribution is given by a function $\sigma$ from outcomes $A$ to the real numbers such that $\sigma(x) \geq 0$ for all outcomes $x$ and $\sum_{x \in A} \sigma(x) = 1$. Some programs,

```java
rand = new Random();
x = 0;
while (true) {
    if (x < 3 && rand.nextBoolean())
        return x;
    x++;
}
```

(a)

```java
rand = new Random();
x = 0;
while (true) {
    if (rand.nextBoolean())
        return x;
    x++;
}
```

(b)

Figure 1: Two randomised algorithms in Java
however, do not terminate always; Figure 1a shows an example. In that case, it is better to consider a subprobability distribution over \( A \) where the sum of the probabilities over all outcomes is at most 1 (rather than equal to 1). The difference to 1 is the probability of non-termination.

On subprobabilities, we can define the relation \( \preceq \) by \( \sigma \preceq \rho \) iff \( \sigma(x) \leq \rho(x) \) for all \( x \in A \). Intuitively, \( \sigma \preceq \rho \) expresses for two subprobabilities \( \sigma \) and \( \rho \) of two algorithms \( A \) and \( B \) that \( B \) returns the outcomes of \( A \) with at least the same probability, but \( B \) may terminate more often. We say that \( B \) is more informative than \( A \). For example, this holds if we take the programs in Figures 1a and 1b for \( A \) and \( B \). This relation \( \preceq \) is an order and chain-complete [1]. This ensures that one can assign subprobabilities to programs with loops and recursive functions via least fixpoints.

When a program can print on the terminal, however, this does not suffice. For example, one program may get into an infinite loop earlier than another and therefore print less, but the text that both print is the same. Hence, we get another order \( \preceq \) on the output: \( \sigma \preceq \sigma' \) iff \( \sigma \) is a prefix of \( \sigma' \). Similar to above, a program \( B \) is more informative than \( A \) if \( B \) prints at least as much as \( A \).

If the program both produces output and is randomised, it is tempting to combine the two orders. Namely, some probability may be shifted from shorter outputs to longer ones, similar to probability being shifted from non-terminating to terminating runs. However, the resulting order is not chain-complete any more. So, fixpoints no longer exist in general [3]. The problem is that the probabilities may be split up in ever smaller parts such that the fixpoint would have to assign positive probability to uncountably many outcomes, but discrete subprobability distributions allow only countably many outcomes.

Lochbihler and Züst [2] avoid this problem by splitting and nesting the subprobability distribution. More precisely, they model sampling functions instead of subprobability distributions. Instead of defining a sampling function for the whole output, they model the result of a program by a sampling function for the first letter of the output and for a sampling function for the remainder of the output (which is split and nested recursively in the same way). This gives a possibly infinite nesting of sampling functions, all of which correspond to discrete subprobability functions. This way, they can define a chain-complete order on these nested sampling functions and deal with loops and recursive functions.

**Objectives**

The sampling functions in [2] use only coin flips. Consequently, probability distributions such as the uniform distribution over three values cannot be expressed naturally. The objective of this project is to generalise the approach with sampling functions to discrete subprobabilities and formalise it as a chain-complete order in Isabelle/HOL. The results shall be applied to model and reason about monadic programs with intermediate output and randomised choice \( p+ \).
In detail, the solution should meet the following requirements:

**Discreteness** All subprobability distributions are discrete.

**Chain completeness** The order on nested subprobability distributions is chain-complete.

**Monotonicity** The monadic operation bind and the choice operation are monotone.

**Tasks**

This project can be subdivided into the following tasks:

1. Identify the key points in the representation with nested sampling function that ensure chain completeness. Translate these insights into a new order and prove on paper that it satisfies the requirements above.
2. Formalise your order and proof of chain completeness in Isabelle/HOL.
3. Define a random choice operation \( p^+ \) and prove that it is monotone in Isabelle/HOL.
4. Prove in Isabelle/HOL that bind is monotone.
5. Find small program examples that highlight the features of your solution. Define them in Isabelle/HOL and prove meaningful properties about them.
6. (optional) Embed the representation with sampling functions from [2] into your representation. Prove that the embedding distributes over the monad and choice operations.
7. Write the final report and prepare the presentation.

**Deliverables**

The following deliverables are due at the end of the project:

**Final report** The final report should consist of an introduction; a theoretical background section; a section on the new definition and the proof of the properties; one or more sections describing the formalisation in Isabelle/HOL; and a conclusion. The report may be written in English or German. Three copies of the report must be delivered to the supervisor.

**Isabelle/HOL theories** Complete Isabelle/HOL development that runs with the latest release or a recent developer’s version.

**Presentation** At the end of the project, a presentation of 30 minutes must be given during an InfSec group seminar. It should give an overview and discuss the most important highlights of the work.
References

